САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

А. С. Сеннов, А. А. Шварц

ГЕОИНФОРМАЦИОННЫЕ СИСТЕМЫ В ГИДРОГЕОЛОГИИ

Учебное пособие

Санкт-Петербург

2005

УДК 519.72+556.3
ББК 26.326
С31

Р е ц е н з е н т ст. науч. сотр. *Е. В. Шалина*
(Центр экологической безопасности РАН)

Печатается по постановлению
Редакционно-издательского совета
Санкт-Петербургского государственного университета

С е н н о в А. С., Ш в а р ц А. А.
С31 **Геоинформационные системы в гидрогеологии:** Учеб. пособие. – СПб., 2005. – 64 с.

Рассмотрен широкий круг вопросов, касающихся создания и использования баз данных и геоинформационных систем в гидрогеологии. В качестве программ-приложений рассмотрена работа с системой управления базами данных Access и географической информационной системой MapInfo. Приведены примеры обработки гидрогеологической информации с помощью вышеуказанных программных продуктов.

Для студентов-гидрогеологов и специалистов в области гидрогеологии и охраны окружающей среды.

ББК 26.326

ВВЕДЕНИЕ

Картографирование всегда было основным методом как изучения, так и наглядного отображения гидрогеологических условий территории. С его помощью решались такие традиционные задачи, как:

- выявление характера распространения, условий формирования и геологической роли подземных вод (ПВ);
- определение закономерностей изменения их ресурсов, состава и свойств в естественных условиях и при техногенном воздействии;
- оценка и прогноз геолого-гидрогеологических условий в результате антропогенной деятельности и т.д.

В настоящее время решение таких задач практически невозможно без использования информационных технологий, среди которых превалируют географические информационные системы (ГИС). К вышеуказанным задачам добавилась задача создания информационных ресурсов для принятия правильных решений в области природопользования.

Основными разделами информации для картографирования, согласно [1], следует считать состояние ПВ, основные типы водных объектов, гидравлические типы ПВ, взаимодействие с поверхностными водами, геохимические типы режима ПВ, а также экзогенные, в том числе мерзлотные, процессы, вызванные деятельностью ПВ.

Гидрогеологические карты подразделяют по объектам картографирования, назначению, содержанию и характеру решаемых задач; тем не менее автоматизация построения распространяется на все типы карт. Ее целью является коренное совершенствование технологий картографирования [1]. Так, традиционные карты при всем их разнообразии можно объединить в следующие группы – карты гидрогеологических условий, карты состояния ПВ и карты гидрогеологического районирования. Электронная карта, в дополнение к этому, может обеспечить автоматизацию трудоемких операций по составлению карт, автоматизированный анализ пространственной изменчивости параметров, снятие ограничения на широкое использование моделей, применение экспертных систем и др. Следовательно, ГИС должна обеспечивать:

1) управление эксплуатацией системы;
2) сбор данных;
3) ввод и хранение данных;
4) подготовку данных;
5) построение карт, разрезов, различных моделей;
6) печать карт и сопровождающих материалов.

Программно-аппаратный комплекс, позволяющий решать задачи автоматического картографирования, должен формироваться с учетом конкретного технического задания, на основе имеющихся технических средств и программного обеспечения.

Таким образом, термин ГИС обозначает как минимум набор аппаратуры, программного обеспечения, географических данных, предназначенный для ввода, хранения, обновления, обработки, анализа и визуализации всех видов географически привязанной информации [2].

Сегодня ГИС-технологии активно развиваются также в области глобальных сетевых технологий [3]. В результате они способны:

- использоваться несколькими пользователями одновременно;
- хранить данные не на одной машине, а на нескольких, что позволяет резко увеличить максимальный объем данных и, кроме того, применять для анализа данные из нескольких источников одновременно;
- находиться на сколь угодно большом расстоянии от пользователя.

Это выражается также в интеграции ГИС с GPS (Global Position Searching, Инструментальное определение координат на местности). Вся активность в области информационных технологий находится под влиянием Интернета и ГИС в том числе. Объединение двух технологий, появившихся практически одновременно, привело к тому, что ГИС обретают принципиально новые возможности. Это позволит использовать ГИС следующего поколения в принципиально ином качестве – из инструмента пространственного анализа ГИС превратится в инструмент управления пространственно-распределенными проектами. Быстрая интеграция ГИС в Интернет привела также к тому, что на рынке представлено множество продуктов для Web-картографирования. В целом можно сказать, что индустрия ГИС активно впитывает новые веяния, изменяется, эволюционирует и развивается, обеспечивая своих пользователей все новыми и новыми возможностями.

Так как ГИС-системы являются средством для работы с данными, эффективная работа с ними невозможна без понимания принципов работы с системами управления базами данных.

1. БАЗЫ ДАННЫХ В ГИДРОГЕОЛОГИИ

1.1. Модели данных

Современные компьютеры могут хранить самые разнообразные формы информации: записи, документы, графику, звуко- и видеозаписи, научные данные, новые форматы данных. Человечество добилось заметных успехов в хранении, получении, управлении, анализе и визуализации данных. Такого рода задачи принято называть управлением данными, а программы для их решения – системами управления базами данных (СУБД).

Несмотря на впечатляющие достижения современных СУБД, следует отметить, что обработка информации, главным образом с целью учета ресурсов, имеет очень давнюю историю. Однако ранее она производилась вручную, и только в начале XIX в. были разработаны технологии автоматической ее обработки с помощью перфокарт.

В 50-х годах XX в. появились первые компьютеры, и информацию стали переносить на магнитные носители. Первоначально способ ее хранения (использование файловой системы) не позволял наблюдать за текущими ее изменениями; но в конце 1960-х годов стали применять системы баз данных (БД) с оперативным доступом к актуальной информации. Такие БД хранились на магнитных дисках, которые обеспечивали доступ к любому элементу данных за доли секунды. Программное обеспечение управления данными давало возможность программам считывать несколько записей, изменять их и затем возвращать новые значения. Поиск данных проводился либо просто по номеру записи, либо ассоциативно по ключу.

Однако быстро выяснилось, что этот способ хранения данных неудобен. Приложениям часто требуется связать две или более записей, в том числе включающих разнородную информацию. Так, информация по режимной сети гидрогеологических скважин может содержать данные как о самих скважинах, о рядах наблюдений за химическим составом или уровнем, так и по ОФР, проводимым в этих скважинах. Конечному пользователю может понадобиться и полная информация об одной конкретной скважине, и более однородная по многим – например, для построения карты гидроизогипс необходима информация по уровням.

Подобные проблемы решались путем применения *иерархической* модели данных, которая предполагала группировку одних записей под

другими. Но по мере развития программ-приложений выяснилось, что различным пользователям желательно иметь разные представления данных, выражаемые в виде разнообразных иерархий, что приводило к необходимости хранения избыточных данных.

Эту проблему пытались решить с помощью сетевой модели данных, в которой каждая запись хранится в одном экземпляре и связывается с набором других записей посредством связи. Например, все данные по конкретной скважине связываются именно с ней. Программа может попросить СУБД их перебрать. Такая модель была широко распространена к началу 1980-х годов, но по ряду причин была неудобна, в основном для разработчиков, а не для конечных пользователей. В качестве альтернативы выступила *реляционная*, наиболее распространенная сегодня модель БД.

Одним из самых естественных способов представления данных является двухмерная таблица. Поскольку каждая сетевая структура может быть разложена в совокупность древовидных структур, то и любое представление данных может быть сведено к двухмерным плоским файлам. Связи между данными также могут быть представлены в виде двухмерных таблиц. Каждая такая таблица обладает следующими свойствами:

- любой ее элемент есть оригинальный элемент данных, повторения отсутствуют;
- все столбцы в таблице однородные, т.е. элементы столбца имеют одинаковую природу;
- столбцам присвоены уникальные имена;
- в таблице не должно быть двух одинаковых строк;
- порядок расположения строк и столбцов в таблице безразличен.

Таблица такого рода называется отношением. БД, построенная с помощью отношений, называется реляционной моделью. Принципиальное ее отличие от сетевых и иерархических состоит в том, что вторые и третьи используют связь по структуре, а первая – по значению. Именно поэтому реляционная технология значительно упрощает задачу проектирования БД.

На смену реляционной модели может прийти объектная, построенная на основе прямых адресных отсылок, хотя в настоящее время для большинства рядовых пользователей она интересна скорее теоретически. Возможно, что при ее развитии выявить данные об объекте будет так же просто, как сейчас получить данные о файле щелчком правой кнопки мыши на нем в окне программы «Проводник».

Преимущества объектной модели данных сводятся к возможности:

- описывать в рамках единого информационного поля объекты, имеющие разнородную внутреннюю структуру и состав элементов;
- установления сложных многоуровневых отношений между информационными объектами;
- вложения объектов друг в друга, выделения общих свойств объектов на верхних уровнях и учет индивидуальных качеств и свойств объектов на нижних уровнях иерархии;
- хранения в едином банке данных структурированной информации и неформализованных данных.

В настоящем пособии пойдет речь о реляционных БД.

1.2. Архитектура БД

Локальные БД часто называются одноярусными, в такой БД все изменения (редактирование, вставка или удаление записей) происходят незамедлительно. Программа имеет с БД прямое соединение.

В двухярусной БД приложение-клиент обращается к серверу через драйверы БД. Управление соединением обеспечивает сервер. За корректность информации, записываемой в БД, отвечает клиент. От него в значительной степени зависит сохранение целостности БД.

В многоярусной архитектуре клиент/сервер приложение-клиент обращается к одному или нескольким прикладным серверам, которые, в свою очередь, обращаются к серверу БД. Программы среднего яруса называются прикладными серверами, поскольку они обслуживают приложения-клиенты. Один прикладной сервер может работать только с данными, обрабатывая запросы клиентов и передавая их БД. Другой сервер может отвечать только за вопросы безопасности (секретности).

Приложения-клиенты запускаются на локальных машинах, прикладные серверы – как правило, на серверах, а сама БД может размещаться на третьем сервере. Многоярусная архитектура позволяет делать приложения-клиенты очень маленькими, поскольку большую часть работы выполняют прикладные серверы. Это дает возможность создавать приложения, называемые минимальными клиентами (thin-client applications).

Другая причина использования многоярусной архитектуры – необходимость управления программными ресурсами. Приложения-клиенты могут быть написаны менее опытными программистами, поскольку эти приложения взаимодействуют с прикладными серверами,

которые, в свою очередь, управляют доступом к самой БД. Прикладной сервер может быть написан более опытным программистом, знающим правила работы с ней. Можно сказать, что прикладные серверы пишутся программистами, в обязанности которых входит защита данных от возможного повреждения некорректно написанными приложениями-клиентами.

Термином «архитектура клиент/сервер» обозначают модель, в которой доступ к БД подразделения или к корпоративной осуществляется с персонального компьютера или рабочей станции. Однако особенных аппаратных средств для архитектуры «клиент/сервер» не требуется. Это программная архитектура, в которой один набор программных компонентов (клиенты) с помощью сообщений просит другой набор программных компонентов (серверы) выполнить различные действия. Серверы выполняют затребованные действия и возвращают полученные результаты клиентам (также с помощью сообщений). И клиенты, и серверы посылают свои сообщения не по адресам, а по именам. В частности, клиенты посылают свои запросы именованным сервисам, а не конкретным машинам, рассчитывая при этом, что в результате разрешения имен будет определен необходимый физический сервер. В таком случае говорят о трехуровневой архитектуре, в которой приложение разделено на части, осуществляющие:

1) управление пользовательским интерфейсом;

2) выполнение правил обработки;

3) выполнение функций хранения и выборки данных.

Одно из следствий использования трехуровневой архитектуры состоит в том, что появляется возможность жестко разделить правила для процессов и правила для данных. Первые реализуются исключительно на среднем уровне, а вторые – чаще всего на уровне управления данными. Одно из самых значительных преимуществ трехуровневой архитектуры заключается в том, что она позволяет приложению (среднему уровню) использовать несколько разных серверов данных без применения каких-либо шлюзовых продуктов для стыковки этих серверов между собой.

Отметим, что при использовании серверных СУБД выполнение запросов производится самим сервером, клиентские приложения получают от него только результаты самого запроса и не требуют передачи лишней информации, что существенно снижает сетевой трафик при обработке запросов.

1.3. Программы-приложения для работы с БД

В зависимости от архитектуры БД программы-приложения обычно разделяют на две группы – настольные и распределенные системы. Первые ориентированы на работу на локальном компьютере одним пользователем. Они часто имеют в своем составе средства разработки, ориентированные на работу с данными и позволяющие создать более или менее удобный пользовательский интерфейс. Обработка данных осуществляется в пользовательском, иначе – клиентском приложении.

В настоящее время известно более двух десятков форматов данных настольных СУБД, однако наиболее популярными являются dBase, Paradox, FoxPro и Access.

Первая промышленная версия СУБД dBase (компания Ashton-Tate, приобретенная позже компанией Borland) появилась в начале 1980-х годов. Благодаря простоте при применении и нетребовательности к ресурсам компьютера она приобрела большую популярность. Хранение данных в dBase основано на принципе «одна таблица – один файл» (*.dbf). Формат данных dBase является открытым, что позволило ряду других производителей заимствовать его для создания собственных СУБД (FoxBase, Clipper).

Современная версия программы (Visual dBase) принадлежит компании dBase Inc. и включает средства создания форм, отчетов и приложений, визуального построения запросов, генерации исполняемых файлов и дистрибутивов, публикации данных в Internet и создания Web-клиентов.

Paradox был разработан компанией Ansa Software в 1985 г. Этот продукт впоследствии также был приобретен компанией Borland. С июля 1996 г. он принадлежит компании Corel и является составной частью Corel Office Professional. Хранение его данных в dBase также основано на принципе «одна таблица – один файл» (*.db *.md *.px). В отличие от программы dBase, формат данных Paradox закрытый. Это имеет свои плюсы и минусы. Положительным обстоятельством является то, что появляется возможность защиты таблиц, отдельных полей и правил ссылочной целостности паролем, которая невозможна при использовании «открытых» форматов.

Современная версия программы включает все возможности, перечисленные для dBase, и некоторые другие, например драйверы для доступа к данным серверных СУБД.

FoxPro – продукт развития от настольной СУБД FoxBase фирмы Fox Software, которая разрабатывалась в конце 1980-х годов; позже продукт был куплен компанией Microsoft. Использует формат dBase (*.dbf). Начиная с версии 3, выпущенной в 1995 г., продукт называется Visual FoxPro. Особенностью этой СУБД является интеграция с технологиями Microsoft, вследствие чего, например, появилась возможность создания распределенных приложений. Visual Fox Pro также включает средства доступа к данным серверных СУБД, таких, как Microsoft SQL Server и Oracle, а также средство визуального моделирования компонентов и объектов. Так же как dBase и Paradox, Visual FoxPro из настольной СУБД постепенно превращается в средство разработки приложений в архитектуре «клиент/сервер».

Первая настольная реляционная СУБД MS Access (для 16-разрядной версии Windows) появилась в начале 1990-х годов. Ее популярность заметно возросла после включения в состав пакета Microsoft Office.

Современная версия этой СУБД ориентирована в первую очередь на пользователей Microsoft Office и допускает разработку БД без использования программирования. В частности, потому вся БД, включая таблицы, индексы, правила ссылочной целостности, бизнес-правила, список пользователей, формы и отчеты, хранится в одном файле (*.mdb). В состав современных версий Access входят средства создания форм, отчетов и приложений, при этом отчеты могут быть экспортированы в формат Microsoft Word или Microsoft Excel, средства публикации отчетов – в Internet, средства доступа – к данным серверных СУБД и создания клиентских. Таким образом, Access можно использовать как настольную СУБД, так и в качестве клиента Microsoft SQL Server, позволяющего осуществлять его администрирование.

Наиболее популярными серверными СУБД являются Oracle, Microsoft SQL Server, Informix, Sybase, DB2. В отличие от настольных, они характеризуются такими особенностями, как реализация для нескольких платформ (UNIX – Solaris, Linux, HP/UX Windows 2000, естественно, кроме Microsoft SQL Server), резервное копирование данных, поддержка репликаций, т.е. копирование информации из одной базы в несколько других, параллельная обработка данных в многопроцессорных системах, распределенные запросы, поддержка собственных и чужих средств разработки и генерации отчетов, поддержка доступа к данным с помощью Internet.

Первой коммерческой реляционной СУБД, поддерживающей архитектуру «клиент-сервер», была Oracle 5 (1985 г.). Версии для персональных компьютеров появились в середине 90-х годов XX в., Oracle также была первой компанией, создавшей СУБД, которая могла применять средства параллельных вычислений (Oracle Parallel Server). Сейчас этот продукт является несомненным лидером в данном классе программного обеспечения. Потому многие компании производят драйверы для доступа к Oracle. Нередко используются готовые решения от самой Oracle Corporation, объединенные под общим названием Oracle Applications.

Первая версия Microsoft SQL Server была разработана для платформы OS/2 в 1988 г. компаниями Microsoft и Sybase. В дальнейшем данная СУБД разрабатывалась для OS серии Windows NT. В настоящее время является второй по популярности СУБД.

Серверные продукты компании Sybase также базируются на ранних версиях Microsoft SQL Server, но с 1994 г. компания разрабатывает свои продукты независимо как для Windows NT, так и для UNIX/Linux. Поддерживает распределенную и параллельную обработку запросов, в том числе к БД других производителей, обработку запросов в многопроцессорных системах. Ведущий продукт фирмы Informix – Informix Dynamic Server, последняя версия которого называется Informix Dynamic Server.2000 (выпущена в сентябре 1999 г.). Данный продукт поддерживает платформы UNIX и Microsoft Windows NT и обеспечивает эффективную работу как на одно-, так и на многопроцессорных системах, поддерживает средства для параллельной обработки данных.

Семейство СУБД DB2 – продукт фирмы IBM. Отметим, что при переносе DB2 на другие платформы фирма старается максимально использовать уникальные функциональные возможности конкретной платформы. Среди них AIX, HP-UX, Linux, OS/2.

Таким образом, существующие возможности наиболее популярных серверных СУБД отражают современные тенденции развития информационных систем, такие, как использование многопроцессорных систем и распределенной обработки данных, создание распределенных систем, в том числе с помощью технологий Internet, применение средств быстрой разработки приложений, создание систем поддержки принятия решений с использованием аналитической обработки данных, а также все более повышающиеся требования к надежности и отказоустойчивости информационных систем.

1.4. Основы проектирования реляционных БД

Основным объектом реляционной БД являются таблицы, каждая из которых имеет свое уникальное имя. Таблица состоит из записей (строка) и полей (столбец), на пересечении которых находятся атрибуты записей. Имена таблиц, так же как и полей, лучше записывать латинскими символами, используя при необходимости знак «_» вместо пробела.

Зачем нужно несколько таблиц? К одной из причин относится возможность избежать дублирования информации. Так, вспомним список temperature.xls, который приведен в качестве примера в п. 1.3. Названия региона и метеостанций в соответствующем поле повторялись многократно. А ведь в этом поле записывались текстовые переменные, длина которых по умолчанию составляет 256 символов. И так столько раз, сколько записей. Можно было ограничить эту длину через пункт меню Данные/Проверка, но в любом случае потребуется запомнить очень много повторяющихся символов. Любой регион Северо-Запада можно представить двузначной цифрой, т.е. всего двумя символами. То же можно сказать про метеостанции. Правда, придется отдельно составить таблицу расшифровки номеров, но все равно это существенно экономит дисковое пространство. Уменьшается также нагрузка на оперативную память, ибо в ней будут обрабатываться меньшие объемы информации.

На содержимое таблиц допустимо накладывать следующие ограничения:

- требование уникальности содержимого каждой ячейки какого-либо столбца и/или совокупности ячеек в строке, относящейся к нескольким столбцам;
- запрет для какого-либо столбца (столбцов) иметь незаполненные ячейки.

Ограничение в виде требования уникальности тесно связано с понятием ключа таблицы. Ключом таблицы называется столбец или комбинация столбцов, содержимое ячеек которого (которых) используется для прямого доступа (быстрого определения местоположения) к строкам таблицы. Различают ключи первичный (единственный для каждой таблицы) и вторичные. Первичный ключ уникален и однозначно идентифицирует строку таблицы. Столбец строки, определенный в качестве первичного ключа, не может содержать «пустое» значение в какой-либо ячейке. Вторичный ключ устанавливает местоположение в общем

случае не одной строки таблицы, а нескольких подобных (в любом случае, это ускоряет доступ к ним, хотя не в такой степени, как первичный ключ).

Ключи используются внутренними механизмами СУБД для оптимизации затрат на доступ к строкам таблиц (путем, например, их физического упорядочения по значениям ключей или построения двоичного дерева поиска, воспроизводящего что-то вроде идеологии «вилки» при артиллерийской стрельбе).

СУБД позволяют пользователю создавать запросы, формы и отчеты. Так же, как и в MS Excel, есть возможность автоматизировать работу путем создания макросов на VBA. Помимо этого, имеется и механизм автоматической записи макросов, т.е. серии макрокоманд, выполняющих конкретные операции (например, открытие форм или печать отчетов). Макросы применяют для автоматизации часто выполняемых задач.

2. ПРАКТИЧЕСКАЯ РАБОТА В MS ACCESS

2.1. Объекты MS Access

Попробуем создать простую БД из нашего списка, хранящегося в файле Karelian.xls. Для этого необходимо его импортировать в MS Access. Но прежде чем это сделать, следует создать новую БД.

Запустим MS Access и установим переключатель в позицию *Новая база данных*. В отличие от остальных компонентов пакета MS Office, СУБД сразу же предлагает определить место для сохранения файла. Операции типа *Сохранить как* в дальнейшем не потребуются. Чтобы узнать, где сохраняется файл, следует обратиться к пункту меню *Файл* | *Свойства базы данных*, вкладка *Общие*.

После того, как файл был сохранен, появляется окно БД. Закрытие его означает закрытие файла БД. Это окно должно всегда быть открыто во время работы с БД Access. Все остальные лишние окна лучше закрывать сразу по окончании работы с ними.

В окне БД видны вкладки, которые позволяют работать с различными объектами СУБД. Основными являются таблицы, запросы, формы и отчеты. Они расположены вдоль левой стороны окна БД в MS Access 2000/2002 и по верхней кромке в MS Access 97.

2.2. Работа с таблицами

Теперь импортируем таблицу MS Excel. Обратимся к пункту меню *Файл* | *Внешние данные* | *Импорт*. В появившемся окне *Импорт*, мало отличающемся от обычного окна открытия файла, установим тип файла Microsoft Excel, найдем нужную папку и, пометив файл Karelian.xls, нажмем кнопку *Импорт*. При этом запустится то, что по аналогии можно было бы назвать Мастером импорта (рис. 1). Впрочем, это название нигде не появляется.

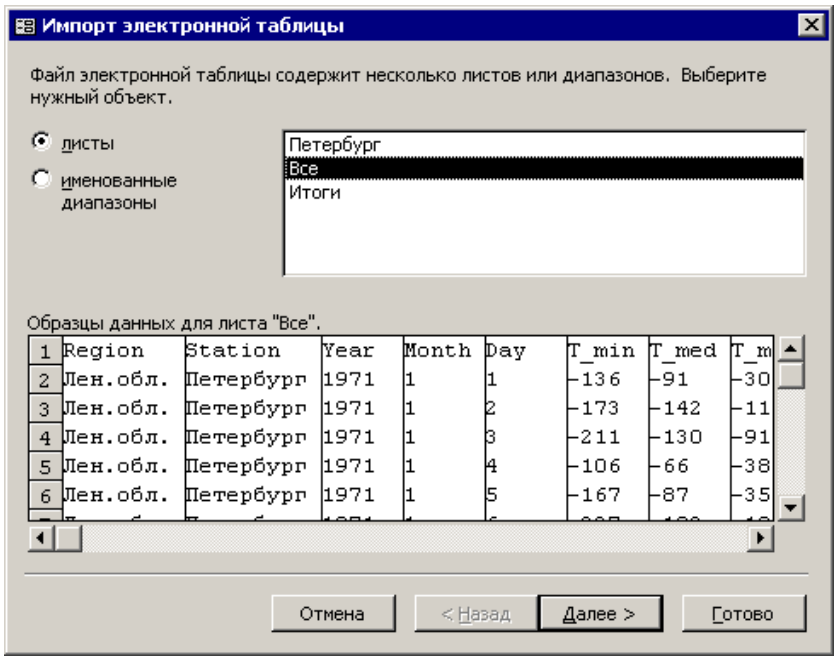

Рис. 1. Импорт электронной таблицы – шаг 1.

На первом шаге следует определить, откуда мы будем импортировать данные – с именованного диапазона или с рабочего листа. Выберем лист Data и нажмем кнопку *Далее*, так как именованные диапазоны не были созданы.

На втором шаге надо подтвердить, что первая строка списка содержит заголовки полей записей. Правильнее всегда именно так и составлять списки. Проверив, что флажок установлен, нажмем кнопку *Далее*. Иногда возможно появление сообщения о том, что некоторые поля имеют неправильные заголовки. Это скорее всего означает, что либо какое-то поле списка не было озаглавлено, либо где-то (в нашем случае, скорее всего, правее списка) были оставлены записи в ячейках.

На следующем шаге следует указать, что данные будут сохранены в новой таблице, после чего появляется возможность описать каждое поле, т.е. поменять имя, пропустить (не импортировать) поле, индексировать, чтобы в дальнейшем было проще искать информацию в этом поле (сейчас лучше оставить его неиндексированным).

После нажатия на кнопку *Далее* появляется следующее окно (рис. 2).

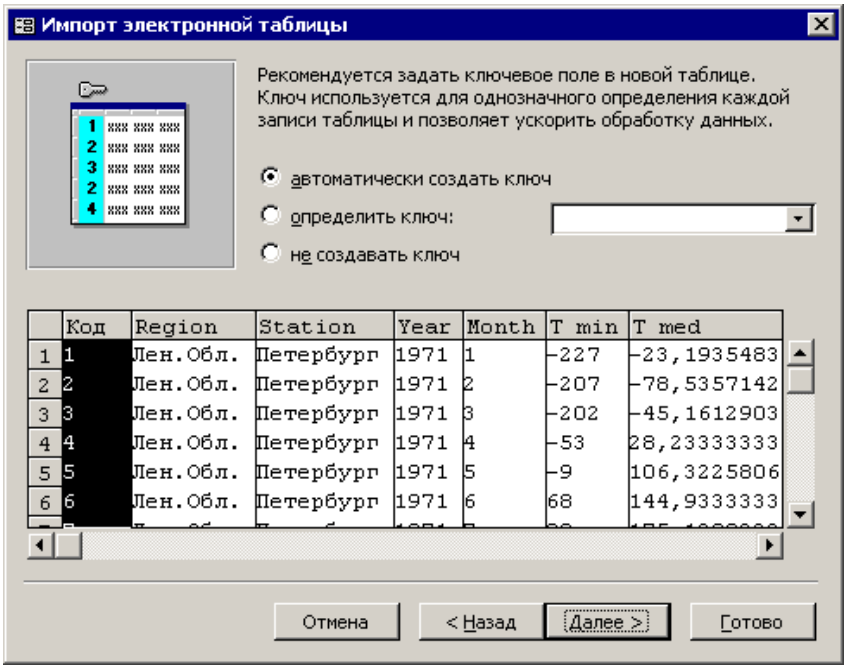

Рис. 2. Импорт электронной таблицы – определение ключевых полей.

Ключевое поле используется для того, чтобы организовывать те самые связи между таблицами, о которых выше уже упоминалось. Оно похоже на столбец типа № п/п, и на первых порах не следует вникать в особенности его использования – просто оно должно присутствовать в таблице, желательно в первом столбце. Для обеспечения однозначности выбора записи значения в ключевом поле не должны повторяться. В нашем списке присутствуют номера водопунктов, которые не повторяются, так что следует назначить этот столбец ключевым.

На следующем шаге появляется возможность задать имя таблицы, по умолчанию предлагается использовать имя рабочего листа, в нашем случае это «data». Последнее диалоговое окно выводит надпись о том, что импорт успешно завершен. Впрочем, иногда MS Access сообщает, что были ошибки импорта. Список ошибок размещается в специальной таблице. Открыв ее, можно проверить, какие именно поля и какие записи не были однозначно интерпретированы. Причем следует помнить, что номера записей будут отличаться на единицу, так как в СУБД первая запись – это первая строка, а в электронных таблицах – вторая.

Итак, в окне БД появилась новая таблица (если вкладка таблицы активна!). Откроем ее двойным щелчком мыши. Таблица мало чем отличается от соответствующей в MS Excel. Правда, одно отличие есть – ни справа, ни снизу нет пустых столбцов и строк. Чтобы изменить тип данных в полях, следует перейти в режим конструктора. Проще всего сделать это, нажав на кнопку  на панели инструментов.

Внутри конструктора таблиц видны три столбца – *Имя поля*, *Тип данных* (в этом поле) и *Описание* (поля). *Имя поля* – это заголовок столбца, *Тип данных* – примерно то же, что формат ячейки в MS Excel, а *Описание поля* – комментарий, который можно написать для себя, чтобы не забыть, какие данные собираемся хранить в этом поле.

После импорта данных следует проверить, правильно ли MS Access понял наши данные. Для этого в режиме конструктора следует пройти по столбцу *Тип данных* и проверить тип данных, видимый на вкладке *Общие*. Так, для поля № водопункта видно, что тип данных числовой, но с плавающей точкой (рис. 3). Очевидно, что для данного поля было бы правильнее установить тип данных *Целое* (2 байта) или *Длинное целое* (4 байта). Это легко сделать, щелкнув мышью в окне *Размер поля*, где можно выбрать нужный тип данных. На рис. 3 данные вполне могут быть типа *Одинарное с плавающей точкой* (4 байта),

но на первых порах лучше поставить *Двойное с плавающей точкой* (8 байт) просто потому, что это стандартный для MS Access тип данных. Стандартным является и тип *Длинное целое*. Используя стандартные типы, можно позволить на первом этапе помнить всего о трех возможных разновидностях данных в нашей базе – текст, целое число и действительное число.

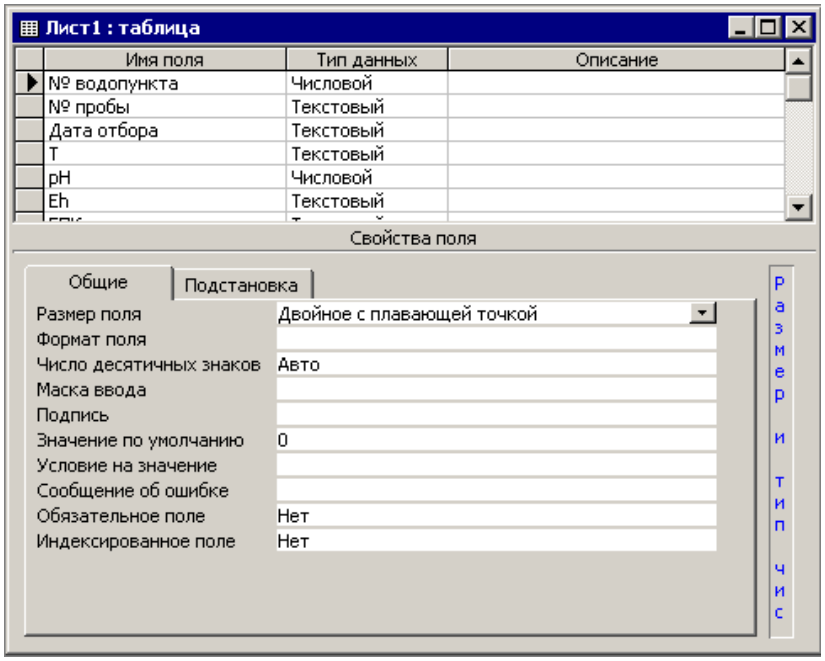

Рис. 3. Импортированная таблица в режиме конструктора.

Два слова про поле *Счетчик*. В конструкторе таблиц видно, что оно индексировано и совпадения в нем не допускаются. Это означает, что через поле *Счетчик* будет осуществляться поиск данных в таблице, и, следовательно, каждая запись должна иметь уникальный номер, который *Счетчик* автоматически присваивает для новой записи. Если запись будет удалена, ее освободившийся номер все равно нельзя будет использовать. Если это вас не устраивает, сделайте ключевое поле типа *Длинное целое*. Но тогда придется вводить номер самостоятельно

и при этом следить за его уникальностью. На наш взгляд, на первых порах лучше использовать *Счетчик*, не обращая внимания на отсутствие некоторых номеров после удаления записей – на самом деле это ничему не мешает. Но в том случае, когда требуется использование конкретного номера (например, номер водопункта), необходимо использовать *Длинное целое*.

Перейдем в режим таблицы. Некоторые операции, которые выполнялись на рабочем листе MS Excel, похожим образом исполняются и здесь. На панели инструментов *Таблица* в режиме таблицы есть серия командных кнопок для выполнения таких операций – *Сортировка по возрастанию*, *Сортировка по убыванию*, *Фильтр по выделенному*, *Изменить фильтр*, *Применить фильтр*, *Найти запись*, *Добавить запись*, *Удалить запись*, *Перейти в окно базы данных* и *Добавить новый объект*, по умолчанию – это *Автоформа*. Сортировка по одному полю выполняется точно так же, как в списках MS Excel. Кнопки *Фильтр по выделенному* и *Изменить фильтр* практически реализуют все возможности автофильтра. Так, щелкнув в ячейку с атрибутом Родник, а затем по кнопке с нарисованной на ней воронкой с молнией (*Фильтр по выделенному*), мы задали критерий фильтрации – показывать только записи по родникам.

Кнопка *Изменить фильтр* действует так же, как режим *Автофильтр*, но элемент управления полем со списком появляется лишь в активном заголовке поля. Просто воронка (*Применить фильтр*) приводит в действие инструмент *Фильтрация данных* в соответствии с установленными критериями (критерием). Бинокль (*Найти запись*) полностью соответствует пункту меню *Правка/Найти*. Далее следуют кнопки *Добавить запись* и *Удалить запись* в режиме таблицы, *Перейти в окно базы данных*. Последняя кнопка, *Новый объект*, позволяет перейти к конструированию новых таблиц, запросов и т.д. По умолчанию выводится *Автоформа*. Она немного похожа на форму из списков MS Excel, но вместо полосы прокрутки справа от полей ввода здесь в нижней части расположен специфический для БД элемент управления. Используя его, можно перейти прямо к нужной записи, введя ее номер в поле, или добавить новую, щелкнув крайнюю правую кнопку. С помощью *Автоформы* нельзя искать записи по критерию – можно только просматривать, добавлять или изменять их. Впрочем, этим инструментом пользуются редко, так как в MS Access есть специальные объекты-формы, которые пользователь может создавать для решения непосредственно своих задач.

Чтобы ознакомиться с другими возможностями MS Access по работе с данными, превратим наш список в БД.

2.3. Создание новых таблиц

Попытаемся устранить дублирующуюся информацию. На вкладке *Таблицы* окна БД видно, что есть три способа построения таблицы – в режиме конструктора, с помощью специального мастера и путем ввода данных. Так как нельзя изучить сразу все, в рамках настоящего пособия ограничимся изучением режима конструктора. Таблица водопунктов должна иметь примерно такой вид:

Код_водопункта	Водопункт
1	Скважина эксплуатационная
2	Скважина картировочная
3	Колодец
4	Родник

При этом значения в первом столбце не повторяются, так как каждая запись должна иметь свой уникальный номер. Для данного поля удобно использовать специальный тип данных, который называется *Счетчик*. Как было указано ранее, он совпадает с типом *Длинное целое* и отличается от него тем, что при введении новой записи значение *Счетчика* автоматически увеличивается на единицу. Это поле удобно применять для организации связи между таблицами, тогда его надо сделать ключевым. По традиции такие поля часто называют, начиная с латинских букв id, или код. Используя английские названия, можно было бы озаглавить ключевое поле id_point, а текстовое поле с расшифровкой – point. В настоящем пособии будем использовать для заголовков кириллицу (имея в виду, что это не рекомендуется для реальных БД), поэтому назовем ключевое поле код_водопункта, а текстовое поле с расшифровкой названий – водопункт. Таким образом, структура новой таблицы в окне конструктора должна выглядеть примерно так, как показано на рис. 4.

Поставив курсор правее, в столбец *Тип данных*, в нижней половине окна конструктора можно изменить размер поля. Для поля с названием региона длина в 50 символов вполне достаточна. Можно было, видимо, сделать его и меньше, но задумываться на эту тему не стоит – ведь надо ввести название всего 1 раз! А вот определить первое поле как ключевое желательно прямо сейчас, для чего следует щелкнуть

левой кнопкой мыши на сером прямоугольнике левее надписи «код_водопункта» и в появившемся контекстном меню выбрать позицию *Ключевое поле*. На сером прямоугольнике возникнет изображение ключика.

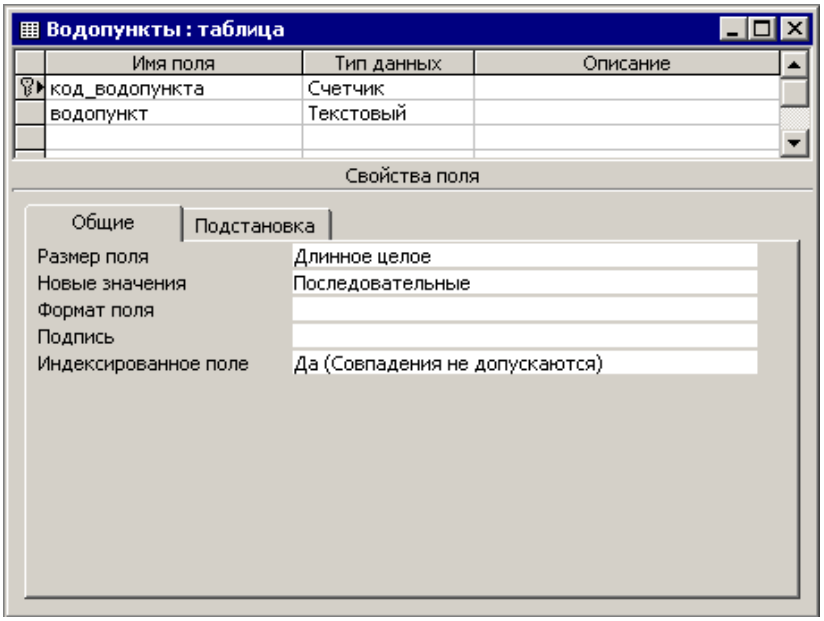

Рис. 4. Структура новой таблицы.

Теперь пора перейти в режим *Таблица* и заполнить ее. После щелчка по кнопке с изображением таблицы будет предложено как-нибудь ее назвать. Подходящим названием будет, например, «Водопункты».

Еще раз отметим следующее: хотя русифицированная версия MS Office позволяет присваивать таблицам и полям заголовки на русском языке, опыт показывает, что удобнее использовать названия на английском языке. Особенно это важно при работе в локальных и глобальных сетях – использование символов кириллицы в заголовках иногда приводит к ошибкам в работе.

Теперь заменим длинные текстовые названия водопунктов в таблице «Гидрогеология» на их коды. Сначала заменим названия на коды, используя обычный прием *Правка/Заменить*. Затем изменим тип данных этого поля с текстового на *Длинное целое*. Причина, по которой был выбран последний, состоит в том, что поле код_водопункта в таблице водопунктов имеет тип *Счетчик*, который, в свою очередь, является разновидностью *Длинного целого*, а организовать связь между двумя таблицами можно только через поля, имеющие одинаковый тип данных.

Щелкнув по вкладке *Подстановка*, введем параметры:

- тип элемента управления – поле со списком, это позволяет организовать подстановку значений из другой таблицы;
- тип источника строк – таблица или запрос;
- источник строк – имя таблицы, поле которой мы хотим присоединить к данному полю;
- присоединенный столбец – первый, т.е. ключевой;
- число столбцов – два, так как собственно названия регионов записаны во втором столбце;
- заглавия столбцов – если *Да*, в момент подстановки будет видно название столбца из связанной (главной) таблицы. По умолчанию установлено *Нет;*
- ширина столбцов – как правило, нам не нужно видеть номер записи в главной таблице, достаточно названия. В этом случае надо написать 0 для первого поля. Для второго поля (названий) лучше не писать ничего, тогда MS Access подберет ширину окна по длине названия;
- число строк списка – в приведенной таблице (см. рис. 4) их всего четыре, так что можно ничего не менять;
- ширина списка – авто, так как мы уже решили, что не будем регулировать ширину полей;
- ограничиться списком – да, конечно, необходимо, чтобы в связанной таблице «Гидрогеология» были только те номера водопунктов, которые есть в главной таблице.

Обратите внимание на то, что в главной таблице левее ключевого поля появился дополнительный столбец с черными крестиками в рамках. Щелкнув по ним, можно увидеть список записей, связанных с данным полем, в подчиненной таблице.

Аналогичным образом импортируем таблицу «Химия» и установим связь между ней и таблицей «Гидрогеология» по полю *Номер во-*

допункта. Отметим, что при этом таблица «Химия» будет подчиненной, и номера водопунктов в нем могут быть только теми, которые есть в главной таблице. В противном случае появится сообщение о нарушении целостности данных (рис. 5).

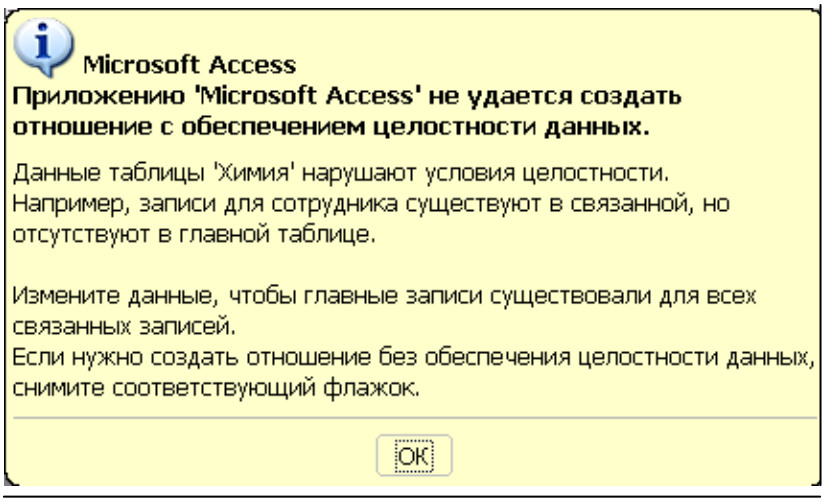

Рис. 5. Нарушение целостности данных.

Так как повторяющиеся текстовые значения в полях записей принято заменять на цифровые коды, правильнее создать таблицу «Категории качества» и выполнить подстановку кодов качества в таблицу «Химия» так же, как она проводилась для поля *Вид водопункта* таблицы «Гидрогеология».

2.4. Схема данных

Для того чтобы можно было в одном запросе обращаться одновременно к нескольким таблицам, необходимо определить связи между ними. Какими они должны быть?

Теоретически связи могут быть трех типов – один к одному, один ко многим и многие ко многим. Связи первого типа, например, реализованы в любой записи таблицы «Гидрогеология». Все поля содержат информацию об одном водопункте. А связь между конкретными водопунктами и наблюдениями за химическим составом в таблице «Хи-

мия», видимо, один ко многим – на одном водопункте было произведено много химических анализов. Связь типа многие ко многим в нашей БД существует между таблицами «Гидрогеология» и «Категории качества». Так, одна категория качества может быть у многих водопунктов, но на одном водопункте могут быть зафиксированы различные категории качества в разных пробах. Связи такого типа в реляционных БД устанавливаются через промежуточные таблицы, в данном случае через таблицу «Химия».

Чтобы определить связи, обратимся к пункту меню *Сервис / Схема* данных. Выберем все три таблицы и нажмем на кнопки *Добавить*, а потом *Закрыть*. В окне схемы данных расставим макеты таблиц в таком порядке – слева «Водопункты», правее «Гидрогеология», еще правее «Химия». Для того чтобы установить связь типа один ко многим между первыми двумя таблицами, щелкнем мышью на поле «код_водопункта» в таблице «Водопункты» и, не отпуская кнопку мыши, наведем ее на поле «вид_водопункта» таблицы «Гидрогеология». После того как мы отпустим кнопку мыши, на экране появится диалоговое окно (рис. 6).

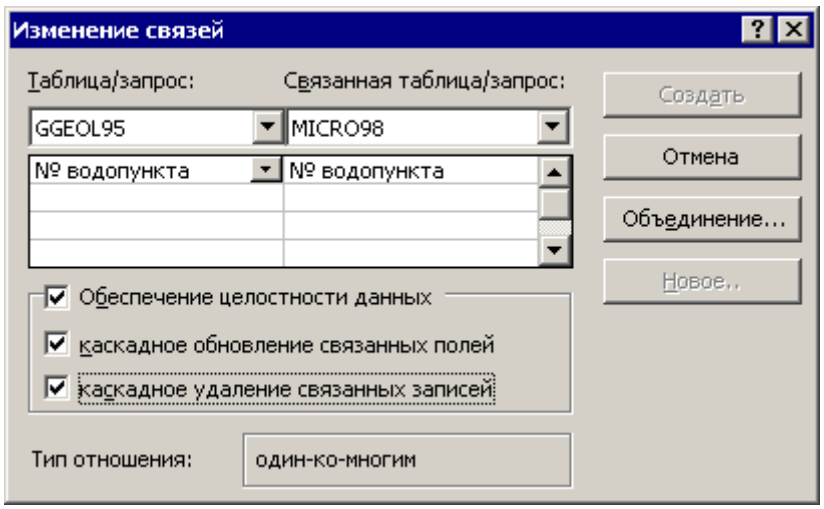

Рис. 6. Установка связи между таблицами.

В нем главная таблица всегда размещена слева. Установим флажок *Обеспечение целостности данных* (MS Access будет следить за

тем, чтобы в поле «Водопункты» таблицы станций были только такие номера, какие есть в поле «код_пункта») и флажки каскадного обновления и удаления связанных записей. Тогда при изменении поля в главной таблице (например, «скв. карт.» на «скважина картировочная») или удалении поля изменятся или будут удалены все связанные поля в таблице «Гидрогеология». Нажмем кнопку *Создать*. В окне *Схема данных* будет нарисована связь типа один ко многим. Аналогичным образом объединим ячейки «код_водопункта» таблиц «Гидрогеология» и «Химия». На рис. 7 показан полученный вид схемы данных.

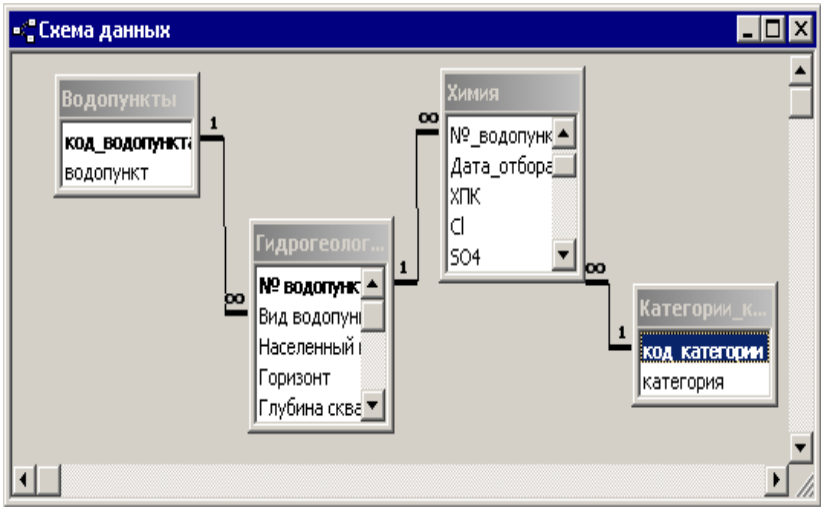

Рис. 7. Окно ***Схема данных***.

На этом этапе часто возникают ошибки и, как правило, они происходят от невнимательности. Нельзя установить связь между полями разного типа – поле формата *Длинное целое* можно связать только с таким же или со *Счетчиком*. Если в главной таблице нет, например, четвертого номера, а в подчиненной он есть, связь не установится. Часто источником ошибок становится значение поля по умолчанию, которое конструктор таблиц считает равным нулю.

На этом формирование БД можно считать законченным. Теперь можно составлять запросы к ней.

2.5. Запросы на выборку

Попробуем составить запрос на выборку. Например, отберем данные по водопунктам с высшей категорией качества. Заранее надо решить, какую информацию мы хотим увидеть в результате выполнения запроса. Пусть такую – название населенного пункта и его координаты.

Дважды щелкнув по строке *Создание запроса в режиме конструктора* и выбрав таблицы «Химия» и «Гидрогеология», попадем в *Конструктор запросов*. Верхняя его половина напоминает схему данных, в ней могут быть отражены не все таблицы, а только те, которые были выбраны для данного запроса. Все таблицы должны быть связаны, иначе запрос не будет работать корректно. Иногда надо проверить связи и удалить лишние, ориентируясь на составленную схему данных – такое возможно в случае совпадения заголовков полей. В нижней половине *Конструктора запросов* нужно установить сначала имя таблицы (вторая строка), потом имя поля (первая строка). Можно и просто перенести нужные поля из верхней половины конструктора в нужную ячейку первой строки. При необходимости ниже записываются условия отбора. Так, например, в столбце «Категория качества» в данном случае надо записать единицу, соответствующую высшей категории.

Составив запрос, нажмем на кнопку с изображением темно-красного восклицательного знака на панели инструментов *Конструктор запросов*. Запрос выполнится. Нажав на голубой треугольник, снова вернемся в режим конструктора. При попытке закрыть окно конструктора получим сообщение о том, что можно сохранить запрос и присвоить ему имя. В этом случае выгода по сравнению с фильтрацией очевидна: запрос будет присутствовать в БД и после внесения любых дополнений или изменений всегда может быть вызван снова прямо со вкладки запросов.

Другим существенным преимуществом запроса является возможность задать запрос с параметром *Условие отбора*. Например, нам нужно ввести класс качества с клавиатуры. Тогда в поле *Условие отбора* нужно записать следующее выражение: Like [?].

Выполнение запроса начинается с появления диалогового окна, в которое можно ввести условие отбора, т.е. номер требуемого класса качества.

Сделаем некоторые пояснения. В общем случае строка, содержащая оператор Like, выглядит примерно так: Like [???]&"*". Ее присутствие в строке условий поля вызывает появление диалогового окна *Введите значение параметра*. Оно означает, что вместо трех символов вопроса следует ввести три первых буквы, например какого-то поселка, района и т.д. Символы амперсанд (&) и звездочка в двойных кавычках ("*") означают, что к ним следует добавить любое окончание. Если бы после закрывающей квадратной скобки ничего не было, запрос искал бы точное соответствие этим двум символам в указанном поле. Отметим, что символы знака вопроса (?) не являются обязательными. В частности, вместо них можно было бы подставить три первых буквы названия того поселка, данные по которому просматриваем чаще всего.

Внутри запроса можно установить сортировку полей, например, *X* и *Y*. Так как первое поле *X* расположено левее второго поля *Y*, записи выводятся отсортированными вначале по координате *x*, затем по *y*. Именно таким образом в запросах осуществляется сортировка по нескольким полям одновременно – поле, находящееся левее, имеет более высокий приоритет.

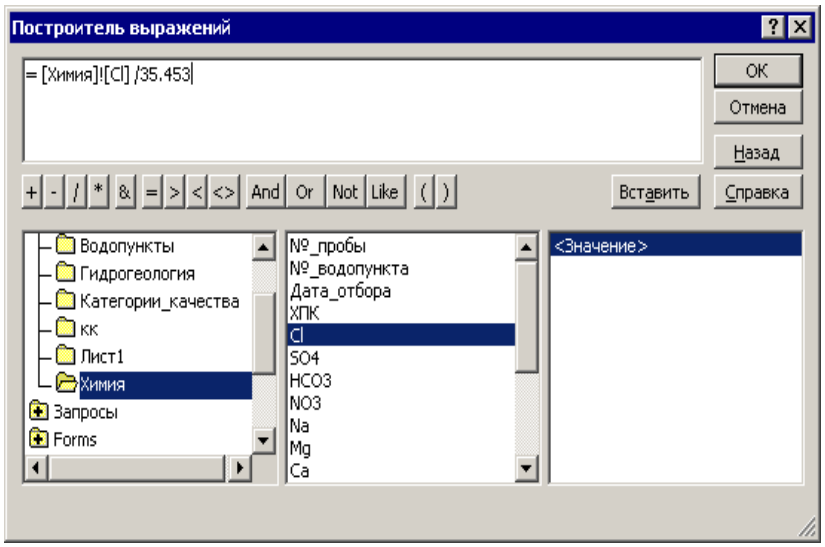

Рис. 8. Окно ***Построитель выражений***.

В БД не принято хранить информацию, которую можно получить из существующих полей путем каких-либо вычислений. Например, если необходимо получить значения элементов в виде мг/экв, а в таблице они представлены в виде мг/л, в запрос можно добавить вычисляемое поле. Для того чтобы его создать, запишем произвольное имя в верхней строке запроса в первом пустом столбце, например: count: и, поставив курсор клавиатуры после двоеточия, нажмем кнопку *Построитель выражений*  на панели инструментов *Конструктор запросов*. В его верхнем левом окне записываем выражение, используя кнопки и возможность подставить имена полей из окна ниже (рис. 8).

Так же, как и в электронных таблицах, имена полей в выражениях отделяются от имен таблиц (листов) символом !. Иногда нужно подправить автоматически генерируемые вставки, не имеющие смысла. Так, вставку «Выражение», нажав на кнопку OK, вводим из окна построителя в верхнюю строку *Конструктора запросов*.

2.6. Запросы на добавление и обновление данных

Если единовременно приходится добавлять только несколько записей, это можно сделать в режиме таблицы или формы, в частности автоформы. Но предположим, что получен большой объем записей и уже в электронной форме. Конечно, для решения подобных задач имеется специальный инструмент – запросы на добавление данных.

Для составления подобного запроса надо ясно представлять, из какой таблицы и в какую будут добавляться данные. Кроме того, теперь уже известно, что данные должны иметь одинаковые форматы, информация должна быть однотипной. Например, если в присланных данных по метеостанциям в поле *Станция* записано название, а не номер, нам придется это исправить.

Когда готовы данные для добавления, вначале готовится запрос на выборку данных из исходной (исходных) таблицы, т.е. из содержащей новые данные. Полезно делать именно так, чтобы убедиться в том, что на этом этапе не допущена ошибка. Убедившись, что выборка данных проходит правильно, можно преобразовать этот запрос в запрос на добавление. Для этого, не выходя из *Конструктора запросов*, надо войти в пункт меню *Запрос* и заменить установленный по умолчанию прежний запрос на выборку на запрос на добавление (зеленый крестик с восклицательным знаком). После этого на экране появится

диалоговое окно выбора той таблицы, в которую будут добавляться данные.

Нельзя забывать о том, что запрос на добавление должен выполняться один раз. Нажав на красный восклицательный знак, мы увидим предупреждение о том, что в таблицу будет добавлено столько-то записей. Повторное выполнение запроса приведет к добавлению тех же записей. Сконструированные и проверенные запросы на выборку, как правило, сохраняют в БД для многократного использования. Сохранение запроса на добавление имеет смысл только в том случае, если мы будем использовать данные для добавления из той же таблицы, удаляя из нее старые данные после выполнения запроса. Новые данные для таблицы могут быть получены, например, путем импорта из другого файла.

Здесь уместно добавить следующее. MS Access позволяет работать с внешними файлами, как с собственными таблицами. В этом случае внешние данные не импортируются, а связываются с БД. С этой целью используют опции меню *Файл/Внешние данные/Связь с таблицами*. Например, организовав связь с таблицей Excel в окне БД, у связанной таблицы увидим характерный голубой *X,* который указывает на то, что данный объект является не внутренним, а внешним по отношению к нашей БД и хранится на диске в формате *xls*. Однако в запросах его можно использовать так же, как объекты самой базы, в том числе и в запросах на добавление.

Из сказанного следует простой вывод – оператор, который вводит новые данные, может делать это в привычном для себя формате, например электронных таблиц. Нужно только заранее согласовать структуру списка со структурой таблицы БД.

Другим видом запроса является запрос на обновление или изменение данных. Предположим, необходимо заменить название поселка. Вначале составим запрос на выборку, и установим критерий отбора записей по названию поселка. Затем преобразуем его в запрос на обновление (в режиме *Конструктора запросов* выбираем пункты меню *Запрос/Обновление*), и введем в поле *Обновление* новое название поселка.

Как и запрос на добавление, запрос на обновление выполняется один раз.

2.7. Простые формы доступа к данным

Используя формы доступа к данным, можно осуществить первоначальный *ввод* данных в таблицы, просмотр и редактирование записей в привычном для пользователя виде, напоминающем обычный документ, при этом выполнение многих операций упрощается, а присутствие на экране только нужной информации помогает не отвлекаться от сути операций.

При правильной организации данных с помощью одной формы можно вводить данные в несколько взаимосвязанных таблиц, реализуя тем самым принцип однократного ввода данных. Иногда удобно создать несколько форм для одной таблицы. Вместе с тем, используя возможности форм, можно организовать более удобный и наглядный *вывод* информации на экран.

Существует несколько способов создания форм. Так, создание Автоформы непосредственно из режима *Таблица* рассматривалось в п. 1.2.2. Можно воспользоваться кнопкой *Создать* на панели инструментов *Окна базы данных* при открытой вкладке *Формы*. При этом на экране появится диалоговое окно *Новая форма*. В окне справа выбираем вид будущей автоформы, таблицу/запрос – источник данных, щелкаем на OK и получаем форму выбранного вида. Для дальнейшего использования ее необходимо сохранить, присвоив имя.

Другим упрощенным способом создания формы является использование *Мастера форм*, который, например, можно запустить и прямо из *Окна базы данных* (можно и из окна новой формы), выбрав строку *Создание формы с помощью мастера* на вкладке форм. Построение формы выполняется в несколько шагов.

На первом шаге необходимо выбрать источник данных (таблицу или запрос). При этом в нижней половине диалогового окна *Создание формы* выводится список полей выбранной таблицы. Щелчок по кнопке с двумя стрелками отбирает все поля сразу, кнопка с одной стрелкой позволяет выбирать поля поштучно. На следующем шаге можно выбрать внешний вид формы с точки зрения расположения полей таблицы (запроса). Возможны следующие варианты внешнего вида формы:

- в один столбец – в окне формы будет выведена одна запись, атрибуты которой располагаются в один столбец;
- ленточный – в окне формы несколько записей, атрибуты каждой расположены в строку, образуя таблицу;

- табличный – то же, что и ленточный, только оформление больше соответствует таблице без графических эффектов;
- выровненный – в окне формы одна запись, но ее атрибуты расположены в несколько строк;
- сводная таблица или диаграмма – в режиме сводной таблицы имеется возможность просматривать исходные данные, упорядочивая поля в областях фильтра, строк, столбцов и данных. В режиме сводной диаграммы можно представлять эту информацию графически.

В третьем окне при необходимости можно выбрать стиль оформления, в четвертом – присвоить форме название. После нажатия кнопки *Готово* в соответствии с выбранными параметрами будет создана форма (рис. 9).

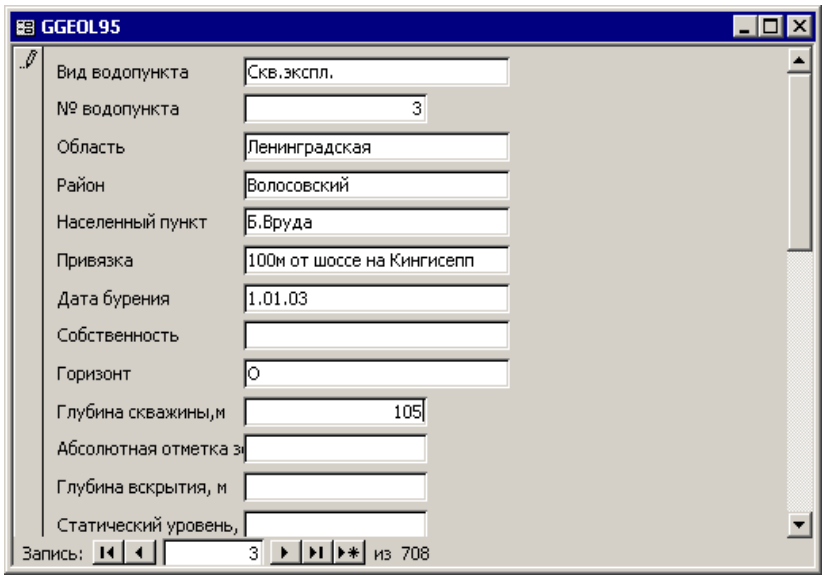

Рис. 9. Простая форма.

Отметим появление специфического элемента управления БД, который расположен слева внизу формы. В нем отображаются номер активной записи и общее число доступных записей. Кнопки слева и справа позволяют переходить на одну запись вправо/влево или в конец/начало таблицы. Можно и просто ввести номер записи в текстовое поле, и форма перейдет на эту запись после нажатия клавиши «Ввод».

Кнопка с изображением стрелки и звездочки (ее копия имеется также на панели инструментов) позволяет ввести новую запись. Для удаления записи используем пункт меню *Правка/Удалить* или соответствующую кнопку панели инструментов.

Наиболее общим способом является создание формы в режиме конструктора. Для этого в окне БД, на вкладке форм выберем строку *Создание формы в режиме конструктора*. На экране появится окно, содержащее макет будущей формы.

Первое, что необходимо сделать – установить связь с источником данных, т.е. с таблицей. Для этого надо вызвать на экран *Окно свойств* (если его на экране нет). Это можно сделать либо через меню *Вид*/ *Свойства*, либо из контекстного меню, которое появляется после щелчка правой кнопкой мыши на строке заголовка формы. На вкладке *Данные окна свойств* в строке *Источник записей* выбираем из списка имеющихся объектов нужную таблицу (рис. 10).

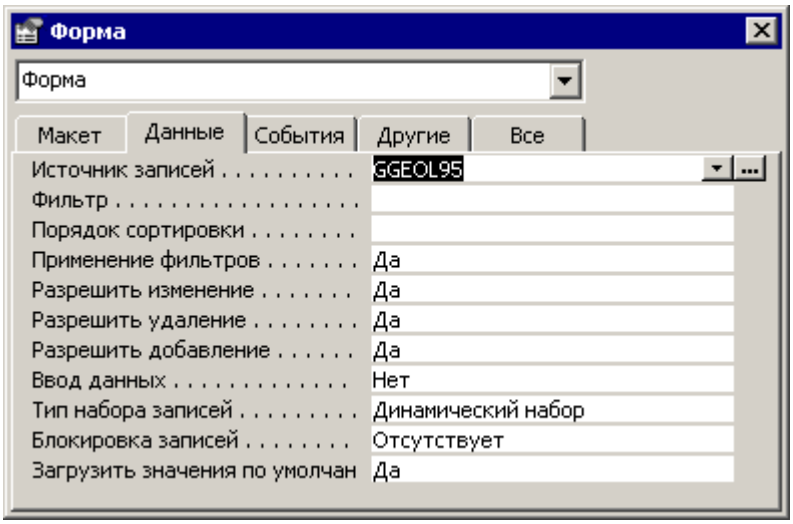

Рис. 10. Окно свойств.

После этого на экране появляется *Список полей* таблицы, присоединенной к форме. Если этого не произошло, вызвать список полей можно через меню *Вид/Список полей* или соответствующую кнопку панели инструментов.

Далее следует расположить поля на макете формы, это, в частности, можно сделать перетаскиванием мышью. Следует обращать внимание на то, что на макете формы появляются как сами поля, так и надписи к ним. По умолчанию при выделении, например перед перетаскиванием поля, выделяется само поле вместе с надписью к нему. Для того чтобы передвинуть поле относительно надписи, нужно выделить его щелчком и перетаскивать мышью, курсор которой в виде руки с вытянутым указательным пальцем направлен на черный квадрат в левом верхнем углу выделения. При перетаскивании надписи вместе с полем курсор имеет вид руки с растопыренными пальцами. Отметим, что в том случае, если в исходной таблице присутствуют поля подстановок, при перетаскивании его мышью на поле формы в последней создается поле со списком. В дальнейшем при работе с формой, например при вводе новой записи, можно использовать данный список.

Для того чтобы выровнять положение полей на форме, можно привязать их к углам сетки. Для этого в режиме конструирования используют пункт меню *Формат/Привязать к сетке*. Флажок в соответствующей строке должен быть установлен, и в этом случае верхний левый угол поля перемещается по узлам сетки. Если необходимо в данном режиме перемещать поле произвольно, следует удерживать клавишу <Ctrl>.

Для изменения шага сетки в режиме конструирования формы в окне свойств формы на вкладке *Макет* следует поменять Число делений по *X* или Число делений по *Y*.

В процессе проектирования формы в конструкторе в любой момент можно перейти в режим формы, чтобы оценить результаты работы. Для этого надо щелкнуть мышкой по левой кнопке на панели инструментов. Продолжить работу можно, щелкнув по той же кнопке, на которой в режиме формы изображен треугольник – переход в режим *Конструктора форм*.

При закрытии формы появляется диалоговое окно сохранения формы; сохранив форму, в дальнейшем получаем возможность вызывать ее прямо из вкладки форм окна БД.

Два слова о настройке свойств формы на вкладке *Данные* (рис. 10). Свойства *Разрешить изменение*, *Разрешить добавление*, *Разрешить удаление* записей в особых комментариях не нуждаются – эти операции можно либо разрешить, либо запретить. Отметим, на наш взгляд, немного невнятную формулировку *Ввод данных*: если это свойство запрещено, к связанной с формой таблице имеется полный

доступ, конечно, в том случае, если разрешены все упомянутые чуть выше три свойства. Если в строке *Ввод данных* установлено *Да*, в связанную с формой таблицу можно только добавлять новые данные, при этом ранее введенные записи не отображаются.

Иногда удобнее показывать на форме сразу несколько записей. Для этого используют ленточную либо табличную форму, в целом отличающуюся только оформлением. Так, создадим ленточную форму на основе имеющейся простой формы. На вкладке форм *Окна базы данных* скопируем имеющуюся форму и вставим ее под другим названием. В окне свойств перейдем на вкладку *Макет* и в строке *Режим по умолчанию* вместо *Одиночная форма* выберем *Ленточные формы*.

Отметим, что при большом количестве полей таблицы такие формы неудобны. При конструировании ленточной формы не следует преувеличивать высоту каждой строки. В режиме конструктора отображается всего одна такая полоса, и ее следует сделать достаточно узкой. Кроме того, теряют смысл подписи полей в каждой строке, логичнее было бы разместить их один раз над таблицей. Для этого используют заголовок формы. Он добавляется вместе с примечанием формы в режиме конструктора с помощью пунктов меню *Вид/Заголовок/Примечание формы*. И заголовок, и примечание формы являются необязательными областями, высоту любой из них в режиме конструктора можно уменьшить до нуля, при этом в режиме формы такая область вообще не будет отображаться. Заголовок формы располагается между линейкой и областью данных формы, примечание – ниже области данных. Надпись из области данных можно вырезать и вставить в область заголовка, расположив ее над соответствующим полем.

2.8. Составные формы

Элементы управления на форме могут быть свободными или присоединенными. Присоединенный элемент управления используется для ввода или отображения содержимого поля из базовой таблицы, запроса или инструкции SQL. Имя поля, к которому присоединен элемент, задается в свойстве *Данные*. Элемент управления, не подключенный к полю в базовой таблице, запросе или инструкции SQL, называется свободным. Свободные элементы управления используют, например, для вывода подсказок или декоративных рисунков.

Присоединенный элемент управления можно создать в форме или отчете, связанном с источником записей в виде таблицы, запроса или

SQL-инструкции, относящихся к БД. Допускается создание элемента управления, присоединенного к полю логического типа.

Для размещения на форме элементов управления обычно используют Панель элементов. Элементы управления С*писок* и *Поле со списком* применяются для вывода на экран строк из списков. При этом принято считать, что первый используется для представления относительно короткого списка, а размер второго не ограничен. В действительности на формах заметно чаще применяют *Поле со списком*, так как при прочих равных оно занимает меньше места. Вместе с тем преимуществом *Списка* является возможность выбрать сразу несколько значений; в свою очередь, преимущество *Поля со списком* – возможность ввода данных в поле непосредственно с клавиатуры. Для того чтобы такие элементы корректно работали, необходимо правильно установить их свойства в окне свойств.

Помимо элементов управления, в Access можно добавлять на одну форму другую, иначе, создавать составные формы. Как правило, в них представлены таблицы БД, связанные между собой отношением один к одному или один ко многим. Различают подчиненные и главные формы.

Подчиненная форма – это одна форма, находящаяся внутри другой, которая, в свою очередь, называется главной. Иногда такую конструкцию называют также иерархической формой или используют термины «родительская» и «дочерняя» формы.

Если источником данных для главной и подчиненной форм служат связанные таблицы, при переходе по записям в главной форме в подчиненной выводятся связанные записи. Подчиненные формы часто используют для вывода данных из таблиц или запросов, связанных отношением один ко многим. Например, для БД по гидрогеологическим скважинам можно создать форму с подчиненной формой для вывода данных из таблицы химических анализов. Данные в главной таблице находятся на стороне «один», в подчиненной – на стороне «многие»: по каждой скважине могут быть проведены несколько химических анализов. Например, когда в главной форме отображается водопункт 7, в подчиненную форму можно вывести только 2 анализа по нему (рис. 11).

При использовании формы с подчиненной формой для ввода новых записей текущая запись в главной форме сохраняется при входе в подчиненную форму. Это гарантирует, что записи из таблицы на стороне «многие» будут иметь связанную запись в таблице на стороне

«один», и автоматически сохраняет каждую запись, добавляемую в подчиненную форму.

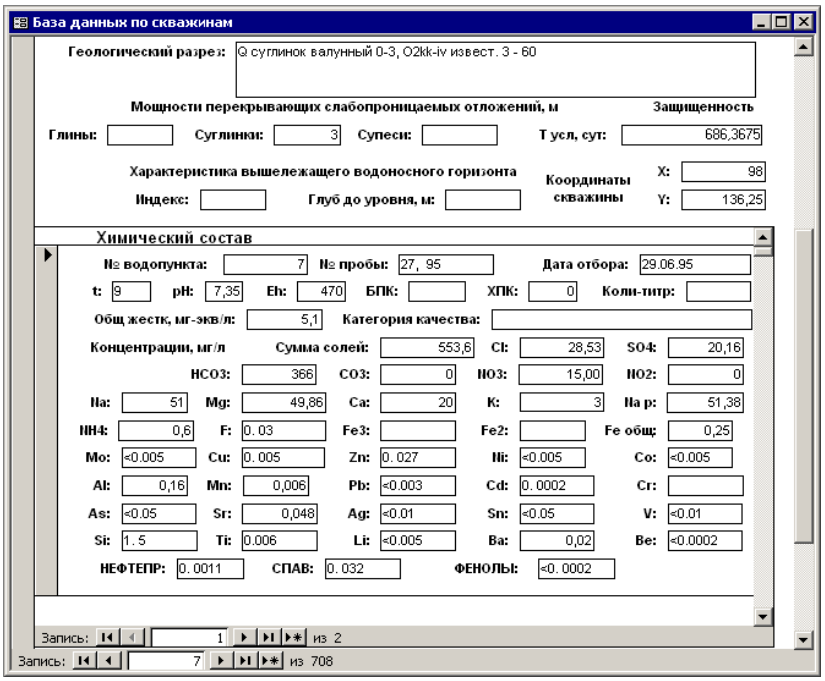

Рис. 11. Составная форма.

В связи с созданием подчиненных форм отметим, что, кроме простой формы, возможно создание также ленточных форм и форм-таблиц. Главная форма может иметь любое число подчиненных форм, если каждая из них помещается в главную форму.

Быстрее всего создать составную форму с помощью *Мастера*. Выбрав на вкладке форм *Окна базы данных/Создание формы в режиме конструктора*, в появившемся диалоговом окне *Создание форм* выберем последовательно нужные таблицы и отберем необходимые для отображения поля, перенося их из левого в правое поле нижней половины окна.

Нажав на клавишу <Далее>, во втором окне создания форм установим переключатель в положение *Подчиненные формы*. Затем в сле-

дующих окнах установим вид подчиненной таблицы (ленточный или табличный), а также стиль оформления. Отметим, что вместо подчиненной формы во втором окне создания формы можно было бы выбрать связанные формы. В этом случае на главной форме была бы размещена кнопка, нажатие на которую вызывает появление подчиненной. Составные формы очень удобны, в частности, при вводе данных в различные таблицы из одного графического окна.

2.9. Диаграммы в формах

Для того чтобы добавить на форму диаграмму, следует открыть эту форму в режиме конструктора, выбрать команду *Диаграмма* в меню *Вставка* и нарисовать на форме местоположение диаграммы. После запуска *Мастера* следует ответить на несколько очевидных вопросов.

При необходимости часто добавлять диаграмму полезно добавить соответствующую кнопку на панель инструментов. Для этого нужно щелкнуть по выбранной панели инструментов правой кнопкой мыши и выбрать команду *Настройка* в контекстном меню. Далее, перейдя на вкладку *Команды*, выбрать *Элементы управления* в списке *Категории* и перетащить кнопку *Добавить диаграмму* на выбранную панель инструментов.

Например, стоит задача построить круговую диаграмму, отражающую макрокомпонентный состав проб таблицы химических анализов. Создадим пустую форму, выберем пункты меню *Вставка/Диаграмма*, отметим крестиком положение диаграммы на форме и в диалоговом окне *Создание диаграммы* определим местоположение полей на макете. Далее включим эту форму, как подчиненную в форму, связанную с таблицей водопунктов (рис. 12).

Затем необходимо нарисовать диаграмму, отображающую цифры, выведенные во второй столбец. В данном случае подходит кольцевая диаграмма, так как ряд данных один. Для того чтобы нарисовать диаграмму, обобщающую данные по всем записям таблицы, достаточно разместить диаграмму в области *Примечание формы*. При этом содержимое строк *Подчиненные поля* должно быть удалено.

В Microsoft Access диаграмма создается заново при каждом ее выводе на печать или просмотре, а также при каждом переходе в режим формы из конструктора. Таким образом, изменения, внесенные в диаграмму, могут быть отменены из-за появившихся противоречий с ис-

точником данных или значением свойства *Источник строк.* При изменении диаграммы необходимо помнить, что изменения данных источника приоритетны над всеми другими. Тем не менее, если в свойстве *Источник строк* задается вычисляемое поле, то форматирование определяется результатами вычислений, а не данными источника.

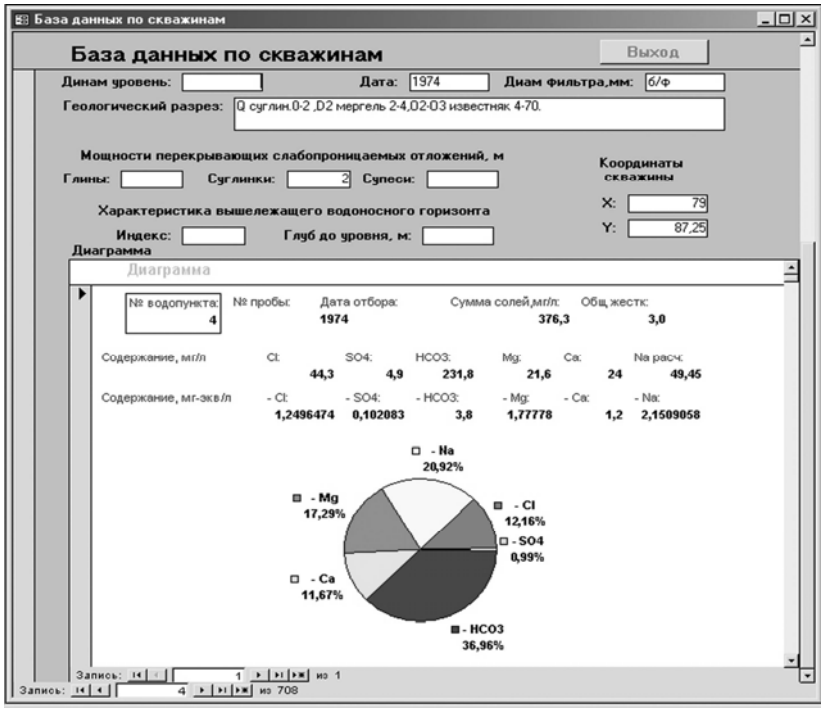

Рис. 12. Диаграмма в подчиненной форме.

2.10. Кнопочные формы

Продумав все операции, которые необходимо выполнять в текущей БД, и создав формы для их выполнения, можно добавить в БД кнопочную форму, которая управляет выводом на экран остальных форм. Для этого выбираем пункт меню *Сервис/Служебные программы/Диспетчер кнопочных форм* и, установив в диалоговом окне строку

Новая страница кнопочной формы, нажимаем кнопку *Изменить*. Появляется окно *Изменение страницы кнопочной формы*, в котором, в свою очередь, следует нажать на кнопку *Создать*. Появляется диалоговое окно *Изменение элемента кнопочной формы*, в котором можно присвоить такие параметры, как название кнопки и выполняемая по ее нажатии операция (рис. 13).

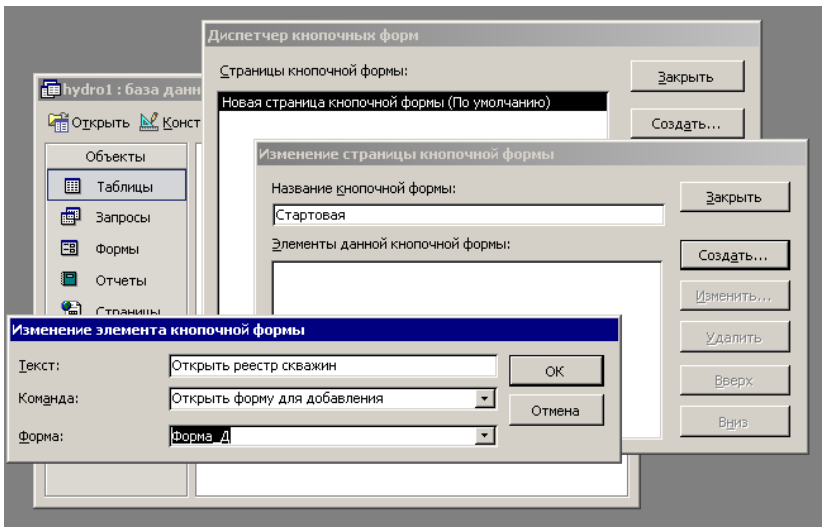

Рис. 13. Создание кнопочной формы.

Для создания кнопочной формы, которая открывает другие кнопочные формы, следует выбрать в поле *Команда* команду *Перейти к кнопочной форме* и указать кнопочную форму, которую надо открыть. Как правило, у команд в поле *Команда* открывается поле со списком, в котором можно выбрать нужный элемент. Для того чтобы изменить или удалить созданную кнопку, выберем ее имя в списке *Элементы данной кнопочной формы* и нажмем кнопку *Изменить* или *Удалить*. Если требуется изменить порядок элементов кнопочной формы, воспользуемся кнопками *Вверх* или *Вниз*. Закончив создание кнопочной формы, нажмем кнопку *Закрыть*. Для удаления кнопки нажимаем кнопку *Удалить*. Если необходимо удалить саму кнопочную форму, необходимо выбрать ее в окне *Диспетчера кнопочных форм* и нажать

кнопку *Удалить*. Пример кнопочной формы с несколькими элементами представлен на рис. 14.

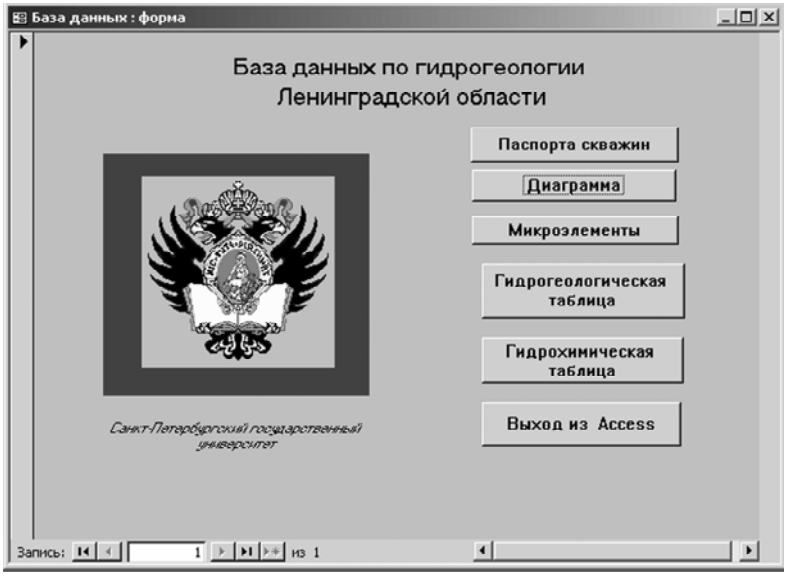

Рис. 14. Пример кнопочной формы.

Набор кнопочных форм позволяет работать с БД примерно так же, как при использовании меню. При необходимости можно выполнить установки, при которых стартовая форма автоматически загрузится при открытии БД.

3. ГИС В ГИДРОГЕОЛОГИИ

3.1. Основы применения ГИС-технологий

Окончательное представление информации из гидрогеологических БД формируется в виде цифровых карт, образующих единую основу для позиционирования объектов и набора тематических слоев данных, совокупность которых образует информационное содержание ГИС. Термин ГИС обозначает набор аппаратуры, программного обеспечения, географических данных, персонала. Такой набор предназначен для эффективного ввода, хранения, обновления, обработки, анали-

за и визуализации всех видов географически привязанной информации.

Аппаратные средства включают компьютеры, на которых работают ГИС. Некоторые ГИС (например, ARC/INFO) функционируют на мощных серверах, обслуживающих клиентские машины в локальных сетях и в Internet. Кроме того, необходимы различные периферийные устройства – дигитайзеры для оцифровки карт, лазерные принтеры, плоттеры для печати карт и т.д.

Программное обеспечение позволяет вводить, сохранять, анализировать и отображать географическую информацию. Ключевыми его компонентами являются средства для ввода и обработки данных; СУБД; программные средства, обеспечивающие поддержку запросов, географический анализ и визуализацию информации; графический интерфейс пользователя.

Таким образом, ГИС – это БД с записями, привязанными к географическим объектам, координаты которых также хранятся в БД. Географические данные обычно отображаются в виде электронной карты, привязанные к ней данные – в виде таблиц. Таблицы связываются между собой по ключевым полям, для них могут быть определены индексы, отношения и т.п., подобно тому, как это описано в п. 2.3. Кроме этого, в ГИС описательная информация связывается с пространственными данными. Отличие ГИС от стандартных СУБД типа FoxPro, dBASE или MS Access состоит как раз в том, что БД ГИС позволяют работать с пространственными данными.

Пространственные данные в ГИС представляются в двух основных формах – векторной и растровой. Растровая графика, или точечное изображение, описывает рисунок, используя цветные точки, или пиксели (от англ. picture element), расположенные на экране монитора, при этом количество разных оттенков может доходить до 16 млн. Любое изображение создается цветом каждой точки сетки. Таким образом, в целом оно представляет собой нечто вроде мозаики. При большом увеличении плавная линия становится похожей на лестницу со ступеньками, но при высоком разрешении, большом числе точек и цветов это незаметно для глаза.

Векторная графика описывает изображение, используя графические примитивы, например прямые, кривые либо ломаные линии. Им ставятся в соответствие параметры, характеризующие цвета и некоторые другие параметры. Например, изображение озера на карте описывается точками, через которые проходит линия, очерчивающая берего-

вую линию. Замкнутая линия представляет собой полигон. Цвет озера задается цветом области внутри полигона.

Векторное изображение может отличаться очень высокой точностью передачи линий и сложных геометрических форм, но его неудобно использовать для отображения оттенков или текстуры. Тем не менее векторные изображения имеют ряд преимуществ по сравнению с растровыми. Векторная графика является масштабируемой, что означает возможность увеличения или уменьшения рисунков без каких-либо искажений. В векторных изображениях часто раздельно представлены координаты векторов (точек) и характеристики их отображения, что позволяет легко получать разнообразные визуальные образы для одной геометрической формы.

Форматы растровых изображений достаточно универсальны и поддерживаются многими приложениями. В частности, со сканера или цифровой камеры снимается именно растровое изображение. Программы для работы с такими изображениями обычно ориентированы именно на обработку изображений, введенных в компьютер одним из этих способов. Кратко опишем основные растровые форматы.

Формат BMP (bitmap, битовое отображение) – стандартный формат графических файлов в ОС Windows, предусматривающий 1, 4, 8 и 24 бита на точку. Поддерживается всеми программами-приложениями. Формат «запоминает» информацию о всех пикселах, поэтому отличается значительным размером.

Формат GIF (Graphic Interchange Format, формат графического обмена) был разработан компанией CompuServe специально для передачи изображений в глобальных сетях. Изображение в формате GIF может содержать до 256 цветов, но их число можно уменьшить для получения меньшего размера файла. Этот формат хорошо сжимает контрастные и несильно заполненные изображения с малым числом цветов. Например, чертежи, рисунки – то, что обычно выполняется средствами векторной графики. На размер файла формата GIF сильно влияет число цветов на изображении, плавные переходы могут передаваться полосами.

Формат JPG (или JPEG), наоборот, предназначен для работы с изображениями, имеющими плавные цветовые переходы. Удобен для фотографий, пейзажей, художественных изображений. Название он получил от аббревиатуры английских слов Joint Photographic Experts Group, т.е. Объединенная группа экспертов в области фотографии. Характеризуется компактностью файлов и более быстрой передачей, чем

GIF, но имеет такие недостатки, как медленное декодирование и иногда потеря деталей изображения. Для него свойственна возможность устанавливать сжатие, проверяя потерю качества. Таким образом, можно выбирать оптимальное соотношение между размером файла и его качеством, детальностью прорисовки.

В полиграфических целях обычно используются несжатые форматы TIFF, PCD, EPS и DCS. Так же как формат BMP, они отличаются заметно бóльшими размерами, чем JPG или GIF.

Форматы векторных файлов не универсальны, обычно каждое приложение, поддерживающее векторную графику, имеет собственный формат. Однако векторная графика широко применяется в ГИС и картографии. Данные форматы более компактны и удобны для работы. Но современные ГИС-пакеты часто используют смешанные векторно-растровые технологии. Некоторые задачи, например обработку данных спутникового зондирования или фотографий земной поверхности, а также интерполяцию точечных изображений на поверхность, проще решать с помощью растровых форматов.

3.2. Примеры ГИС

Классифицировать ГИС-программы можно по разному – по территориальному охвату (общенациональные и региональные); по целям (многоцелевые, специализированные, в том числе информационно-справочные, инвентаризационные, для нужд планирования, управления); по тематической ориентации (общегеографические, отраслевые, в том числе водных ресурсов, использования земель, лесопользования, туризма, рекреации и др.). Часто такие технологии подразделяют по функциональным возможностям – крупные, ориентированные на мощные сетевые серверы, способные обрабатывать значительные объемы информации и работающие под управлением различных версий ОС UNIX, и настольные, обладающие меньшими возможностями. Большинство таких настольных систем являются «урезанными» версиями крупных, переписанными под ОС Windows, например ARC/INFO, которая разрабатывается Институтом исследования природных систем США ESRI (Environment System Research Institute). Программные средства ARC/INFO работают на миникомпьютерах и рабочих станциях различного типа под управлением операционной системы UNIX, на персональных компьютерах типа Apple Macintosh и PC. Структура ARC/INFO 3.4 включает несколько интегрированных

модулей, предназначенных для работы как с векторной географической информацией (точки, ломаные линии, замкнутые контуры), так и с атрибутивной (текстовой), логически связанной с географической. ARC/INFO поддерживает полную реляционную БД и язык программирования для обработки табличной информации, интерактивного ввода и редактирования пространственных и табличных данных, вывода их на экран дисплея и на другие графические устройства, связь с другими ГИС и СУБД (например, dBASE, Oracle). Векторные форматы данных, в которые осуществляются экспорт/импорт информации, включают в себя IGES (Initial Graphics Exchange Specification, исходный стандарт обмена графическими данными), DXF (AutoCAD Drawing Exchange Format, обменный формат чертежей AutoCAD) и т.п.

Другой широко применяемой системой является GeoDraw/ GeoGraph. Она разрабатывается с 1992 г. в Центре геоинформационных исследований Института географии РАН (ЦГИ ИГ РАН). Данная ГИС поддерживает клиент-серверные приложения, имеет двуязычный русско-английский интерфейс, работает с БД через ODBC. Для написания приложений всем пользователям бесплатно предоставляется GeoConstructor, выполненный в стандарте VBX (OCX) и позволяющий создавать ГИС-приложения в современных средах визуального программирования (Visual Basic, Visual C++, Delphi, dBase). Система может работать с разными форматами как пространственных, так и атрибутивных данных, включая SQL-сервера (например, с Oracle). Ее копии работают во многих европейских и северо-американских странах, чем может похвастать далеко не каждая ГИС российского производства. Информация о ней доступна на сайте http://geocnt.geonet.ru/.

ГИС MapInfo – динамично развивающаяся система, приобретающая все бóльшую популярность. Предусмотренные в ней возможности по созданию тематических карт позволяют формировать запросы к БД и получать отображение результатов на карте. Имеется возможность составлять отчеты, которые представляют собой набор карт и легенд к ним, таблиц из БД и т.п. Появляются и другие пакеты, работающие с форматами MapInfo. Например, Vertical Mapper, позволяющий работать с трехмерными данными. Отображение информации в форматах MapInfo предусмотрено в электронных таблицах MS Excel. В ГИС MapInfo картографическая информация в пределах слоя-таблицы организуется в виде отдельных графических примитивов – объектов. ГИС MapInfo предоставляет разработчикам язык MapBasic для написания собственных модулей.

3.3. Представление гидрогеологической информации

Пространственные данные в ГИС представляются в двух основных формах – векторной и растровой. Растровые данные – это отдельные точки, которыми манипулируют компьютерные программы, как по отдельности, так и группами. Применяются они в основном там, где графическая информация должна быть просмотрена, но не нуждается в модификации или анализе. Векторные данные хранятся в виде точек и линий, связанных геометрически и математически, используются в системах для предоставления информации, нуждающейся в анализе. Например, при обработке снимков, полученных из космоса, придется работать с растровыми форматами, а карту границ водоносных горизонтов разумнее хранить в векторном формате. Причем следует понимать, какой способ отображения, площадной или точечный, больше соответствует характеру отображаемой информации. По нашему мнению, для гидрогеологических ГИС наиболее правильным подходом является создание тематических точечных карт по различным параметрам, так как даже в пределах отдельных участков одного водоносного горизонта любое обобщение с выделением каких-либо полей неизбежно ведет к искажению природной ситуации. Обобщение параметров на площади при необходимости можно осуществлять как с помощью инструментов анализа, имеющихся в составе самой ГИС, так и во внешних информационных системах.

Точечный подход имеет ряд преимуществ:

1) объективность получаемой картины, так как данные по составу воды, геологическому разрезу, фильтрационным параметрам горизонта, уровенному режиму и т.д. изначально получены в скважине, т.е. в точке;

2) оперативность внесения изменений в связи с получением новой информации (меняются атрибуты у уже имеющегося объекта или добавляется новый объект);

3) возможность визуальных обобщений на карте точечных объектов, которая позволяет наглядно демонстрировать преобладающие диапазоны эколого-гидрогеологических параметров, а при необходимости просмотреть первичные данные по аномальным точкам;

4) карты точечных объектов можно использовать совместно с электронными картами и снимками, оцифрованными с различных масштабов в пределах точности координатной привязки скважин.

Для создания в интерактивном режиме тематических точечных карт по интересующим нас параметрам важно иметь правильно составленную структуру БД. Структура БД, или таблиц атрибутов, в ГИС играет роль легенды для традиционных тематических гидрогеологических карт. При этом важно правильно составить не только структуру БД, но и определить количество и формат вводимой и вычисляемой информации. БД должен позволять оперативно создавать тематические эколого-гидрогеологические карты по запросу сразу после введения вновь полученной информации по химическому составу, условиям эксплуатации подземных вод и т.д. Пользователь должен иметь возможность знакомиться как с первичным материалом, так и с результатами его обработки.

Типы таблиц атрибутов в гидрогеологии, согласно Требованиям Министерства природных ресурсов РФ [1], в соответствии с различными видами работ, могут быть следующими:

- аэровизуальные наблюдения и дешифрирование аэрокосмоснимков;
- анализ воды;
- бурение;
- геофизические работы;
- маршрутные наблюдения;
- опытно-фильтрационные работы;
- изучение режима подземных вод;
- отбор проб воды;
- отбор проб грунтов и горных пород;
- инженерно-геологическое изучение горных пород;
- эколого-геологическое изучение недр.

3.4. Ввод графической и атрибутивной информации

Ввод графической информации осуществляется с бумажного носителя дигитайзером, интерактивным векторным цифрованием по растровой подложке либо подключением внешних БД.

При использовании дигитайзера бумажная карта закрепляется на планшете, и координаты объектов на ней передаются в компьютер оператором, выбирающим точки произвольно. Предполагается, что для некоторых (контрольных) точек координаты в требуемой системе известны. Таким образом, оцифровка проводится на бумажной основе.

При применении сканера бумажное изображение переводится в растровое, и оцифровка проводится на экране монитора.

В последнее время, благодаря развитию программного обеспечения, получил развитие способ интерактивного цифрования по растровой подложке. При этом оператор обводит объект курсором, а программа-векторизатор сама определяет границы такого объекта.

Практически во всех ГИС предусмотрена возможность подключения внешних БД стандартных форматов, которые могут содержать как графическую, так и атрибутивную информацию. Процесс координатной привязки атрибутивной информации называется *геокодированием*. Как правило, координаты записи присваиваются по результатам сравнения данных из таблицы и данных из другой таблицы, которая уже имеет координаты *x* и *y* для ее объектов.

Современные СУБД, которые используются в составе программного обеспечения ГИС, различаются по типам поддерживаемых моделей данных (иерархические, сетевые и реляционные), но особо широкое применение получили СУБД реляционного типа. Такие СУБД позволяют представить данные о пространственных объектах (точках, линиях и полигонах) и их характеристиках (атрибутах) в виде отношения или таблицы, строки которой – индексированные записи – соответствуют набору значений атрибутов объекта, а колонки (столбцы) обычно устанавливают тип атрибута, его размер и имя атрибута. В число атрибутов не входят геометрические, описывающие их геометрию и топологию. Векторные записи координат объектов упорядочиваются и организуются с использованием особых средств. Связь между геометрическим описанием объектов и их семантикой в реляционной таблице устанавливается через уникальные номера-идентификаторы.

3.5. Управление данными и геоанализ

Удобство манипулирования данными в БД существенно зависит от языковых средств СУБД. Широкие возможности предоставляются пользователю СУБД, в которых реализован язык обработки запросов SQL и его расширения, адаптированные к описанию пространственных запросов к БД ГИС и содержащие конструкции, включающие пространственные переменные и условия.

Одним из главных мотивов, определяющих необходимость использования технологии БД при создании ГИС в настоящее время, является поддержка современными СУБД сетевых возможностей хранения и использования технологий локальных сетей (LAN) и удален

ных сетей в так называемых распределенных БД. Тем самым достигаются оптимальное применение вычислительных ресурсов и возможность коллективного доступа пользователей к распределенным БД.

Блок анализа данных, являясь одним из трех крупных модулей ГИС (ввода, обработки и вывода), составляет ядро ГИС-технологий, все остальные операции которых являются сервисными, обеспечивающими возможность выполнения системой ее основных аналитических и моделирующих функций. Содержание аналитического блока современных программных средств формировалось и формируется в процессе реализации конкретных ГИС-проектов, в результате чего образовался относительно постоянный набор операций, наличие или отсутствие которых в составе конкретного продукта указывает на его качество. Приведем примерный список таких операций: переструктуризация данных, изменение проекций и систем координат, операции вычислительной геометрии, наложение разнотипных слоев данных, аналитические, графо-аналитические и моделирующие инструменты.

Результаты обработки и анализа данных в конечном счете должны быть визуализированы и выведены на печать. Аппаратные и программные средства ГИС имеют широкие возможности генерации табличных графических и картографических выходных данных. Среди них средства машинной графики, конвертеры, позволяющие без потерь преобразовывать данные из одних форматов в другие, графопостроители, графические дисплеи с высоким разрешением и т.д.

4. ПРАКТИЧЕСКАЯ РАБОТА В ГИС MAPINFO

4.1. Структура данных

Как большинство программ для работы с электронными картами, ГИС MapInfo работает со слоями. Каждый слой называется таблицей. Несколько таблиц, относящихся к одной карте, называются рабочим набором. После запуска программы на экране возникает диалоговое окно с предложением открыть либо таблицу, либо рабочий набор, либо последний рабочий набор, либо просто восстановить последний сеанс работы. Для того чтобы построить таблицу заново, все это можно просто отменить.

В таблицах содержится вся информация – графическая, текстовая и др. Каждая таблица является группой файлов-компонентов, каждый из которых содержит информацию одного типа: графические объекты,

БД или индексы. Основной смысл работы по заполнению электронной карты информацией состоит в том, чтобы скомбинировать некоторые данные с картами, которые мы либо создали сами, либо получили от какого-либо разработчика. Таким образом, любая работа начинается с открытия одной или нескольких таблиц. Для добавления нового слоя надо создать новую таблицу. Так как ГИС в некотором смысле является СУБД, перед началом работы с новой таблицей ее необходимо сохранить на жестком диске. Но в отличие, например, от БД MS Access, таблица сохраняется в нескольких файлах. Перечислим их.

- Имя.tab. Этот файл содержит описание структуры данных таблицы. Он представляет собой небольшой текстовый файл, описывающий формат того файла, который содержит данные.
- Имя.dat. Этот файл содержит табличные данные. Отметим, что имеется возможность работать с файлами dBASE/FoxBASE, ASCII с разделителями, Lotus 1-2-3, MS Excel или MS Access, в том числе и через ODBC. В этом случае таблица MapInfo будет состоять из файла с расширением tab и либо файла данных, либо файла электронной таблицы. Таблицы, содержащие растровые изображения, хранят данные в файлах-компонентах форматов BMP, TIFF или GIF.
- Имя.map. Данные могут включать в себя также графические объекты, которые описываются в этом файле, например, из DXF-файлов. В п. 4 вы узнаете о том, как присваивать координаты *x*- и *y*-записям, чтобы отображать их на карте. Если записям соответствуют координаты *x* и *y*, то таблица содержит графические объекты.
- Имя.id. Этот файл включает список указателей (индексов) на графические объекты, позволяющий MapInfo быстро находить объекты на карте.
- Имя.ind. Таблица может также еще раз содержать индексный файл, который дает возможность проводить поиск объектов на карте. Как и в любой БД, если есть необходимость искать некоторые значения в полях таблиц, соответствующие поля таблицы должны быть проиндексированы.

Итак, чтобы работать с данными, необходимо открыть содержащие их файлы или таблицы. Чтобы открыть таблицу, выполним команду *Файл/Открыть таблицу*. Появится диалоговое окно *Открыть таблицу*. Отметим, что в списке перечисляются только файлы с расширением tab, именно их и следует открывать.

MapInfo автоматически открывает таблицу, показывая окно *Карта*. Если таблица не содержит графических объектов, MapInfo откроет

для нее окно *Список*. Возможно представление данных и в виде графика.

Если необходимо открыть внешнюю таблицу, выбираем один из возможных типов файлов. При этом в окне *Имя файла* будут перечислены только файлы с соответствующим расширением. Например, если выбрана строка *Microsoft Excel* [**.xls*], то MapInfo покажет только файлы формата XLS. После открытия файла будет создана таблица для его данных, потому в дальнейшем формат этих файлов не нужно указывать. Если будет предпринята попытка открыть файл данных в его исходном формате, на экране появится сообщение о том, что таблица уже определена и можно отменить ее импорт. Впрочем, можно построить ее заново, нажав на кнопку *OK*.

MapInfo может показывать растровые изображения. Их обычно используют как подложку под векторные слои карты. Растровое изображение нужно зарегистрировать в MapInfo, т.е. привязать его к координатам, и тогда система сможет корректно отобразить его. Открытие растрового изображения в диалоговом окне *Открыть таблицу* автоматически сопровождается появлением диалогового окна *Регистрация изображения*, в котором можно задать систему координат. В результате регистрации создается TAB-файл, имя.tab, который можно открывать как таблицу. Растровые изображения, входящие в комплект поставки MapInfo, уже зарегистрированы.

Любая таблица может быть показана в нескольких окнах различных типов. Вы можете вывести таблицу одновременно в окне *Карта* (границы стран) и окне *Список* (названия стран, их население и т.д.). В соответствии с технологией синхронного представления данных MapInfo любые изменения данных в одном из окон приводят к их автоматическому изменению в остальных окнах.

Техника работы с окнами традиционна для среды Windows. Активно может быть только одно окно. Указав мышью на другое окно, делаем его активным. В зависимости от того, какой тип окна активен, меняется состав строки заголовков меню. Например, если активно окно *Карта*, то появляется меню *Карта*, а если перейти в окно *Список* – *Список*.

Для того чтобы данные можно было отображать на карте, они должны содержать координаты. Процесс присваивания координат записям данных называется *геокодированием*. MapInfo присваивает записи координат по результатам сравнения данных из таблицы и дан-

ных из другой таблицы, которая уже имеет координаты *x* и *y* для ее объектов.

Кроме растровых, MapInfo размещает на слоях четыре основных вида объектов, для работы с которыми используют панель инструментов *Пенал* (рис. 15). Отметим, что вместо более привычного пункта меню *Вид* для вывода на экран панелей инструментов здесь используют пункт меню *Настройки/Инструментальные панели.*

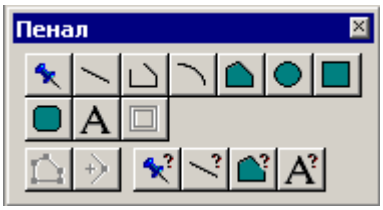

Рис. 15. Панель инструментов *Пенал.*

Этими объектами являются:

- области или полигоны (имеющие площадь): замкнутые многоугольники, эллипсы и прямоугольники, представляющие регионы, территории, округа, городские районы, зоны бедствий или коммерческих интересов и т.д.;
- точечные объекты, т.е. объекты, не отображаемые в масштабе карты: это могут быть метеостанции, поселки, адреса клиентов и т.д.;
- прямые линии и дуги;
- ломаные линии или полилинии.

Освоив средства ГИС, в частности MapInfo, можно рисовать на карте и надписывать области, вычислять расстояния, например от базы до точки наблюдения, сосчитать количество таких точек, находящихся на заданном расстоянии от базы, и т.д. С помощью операций выбора можно выделять информацию из наборов данных и получать ответы на вопросы такого типа: «В каких скважинах единичный дебит превышает 10 л/с», «Какие скважины расположены на расстоянии не более 100 км от базы?», «В каких скважинах, расположенных на расстоянии не более 100 км от базы, единичный дебит превышает 10 л/с?». Можно выделять области (административные территории, водные бассейны, водоносные горизонты) разными цветами или штриховкой в зависимости, например, от сложности геологического строения или числа точек наблюдения на единицу площади.

4.2. Создание карты

Попробуем создать несложную электронную карту, используя в качестве основы растровое изображение части Ленинградской обл.

На первом этапе загрузим в MapInfo карту в виде растрового рисунка (Len_obl.jpg). Для этого следует обратиться к пункту меню *Файл/Новая таблица* и в появившемся диалоговом окне установить флажок *Показать картой*. Нажав кнопку *Создать*, переходим в диалоговое окно *Создать структуру таблицы* (рис. 16).

Выявим структуру таблицы хотя бы из одного столбца. После этого надо определить проекцию. Щелкнув кнопку *Проекция*, переходим в диалоговое окно *Выбор проекции* (рис. 17).

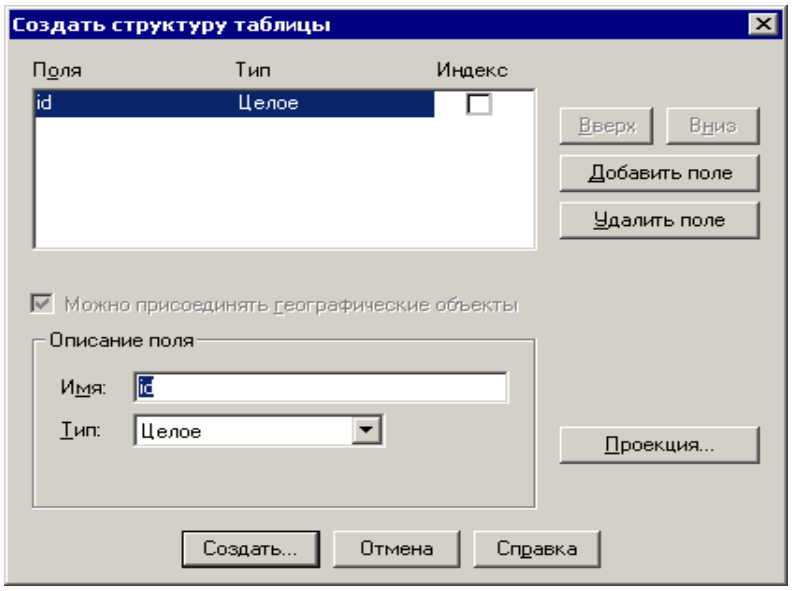

Рис. 16. Диалоговое окно *Создать структуру таблицы*.

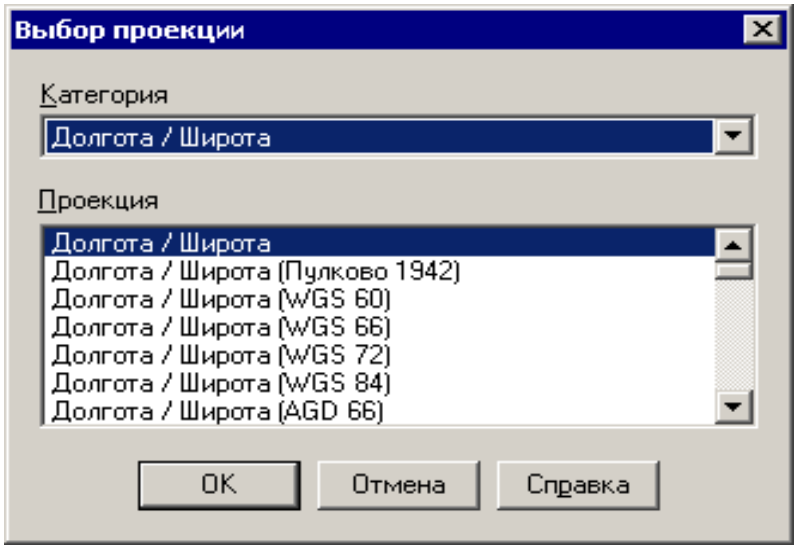

Рис. 17. Диалоговое окно *Выбор проекции.*

Установив стандартную категорию *Долгота/Широта* (для небольших планов используют план-схему), нажимаем кнопку *Создать*. На экране появляется пустое окно карты. Обратимся к пункту меню *Карта/Режимы* и в появившемся диалоговом окне поставим переключатель *Положение указателя* в рамке *Показывать внизу*. После этого в строке состояния внизу слева будет отображаться положение курсора мыши на поле карты. Для того чтобы переместить центр карты в район Ленинградской обл., обратимся к пункту меню *Карта/Показать по-другому* и установим более подходящее положение центра окна и масштаб.

Теперь можно зарегистрировать растровый рисунок. Для этого следует обратиться к пункту меню *Файл/Открыть таблицу* и в появившемся диалоговом окне показать тип *Растр.*

После нажатия на кнопку *Открыть* появляется диалоговое окно, в котором можно определить способ открытия таблицы. Выбрав *Регистрировать*, переходим в диалоговое окно *Регистрация изображения* (рис. 18).

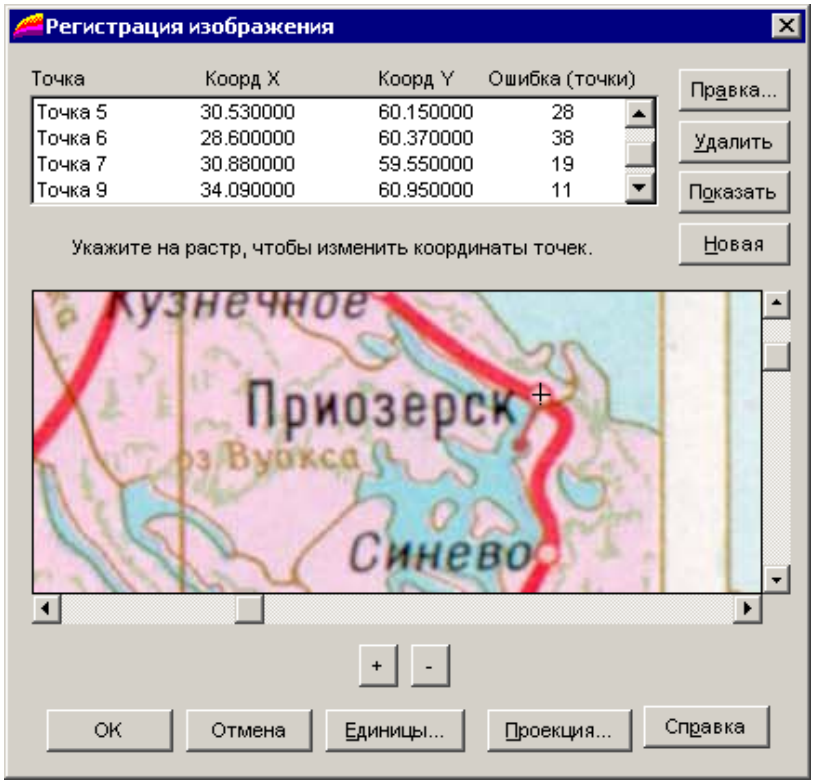

Рис. 18. Диалоговое окно *Регистрация изображения*.

Необходимо последовательно выделить на карте точки с известными координатами, которые нужно указать в соответствующем окне. После регистрации более трех точек на экране появляется значение ошибки в пикселях. Закончив ввод координат всех известных точек, нажмем кнопку *OK*. Теперь с рисунком можно работать, как с картой, например измерять расстояния, увеличивать/уменьшать масштаб, добавлять координатно-привязанную информацию. Для этого используют панель инструментов *Операции* (рис. 19).

Рис. 19. Панель инструментов *Операции.*

Она включает следующие инструменты: стрелка (указатель); кнопки выбора объектов в рамке, круге, полигоне, области и на графике; увеличивающая и уменьшающая лупы; кнопка изменения режима показа карты; ладошка для перемещения изображения внутри окна; кнопка *Информация* для добавления в таблицы координатно-привязанной информации; кнопка вызова диалога *Управление слоями*; линейка и некоторые другие. Так, для того чтобы измерить расстояние между двумя пунктами, щелкаем по кнопке *Линейка*, после чего появляется специальное окно *Линейка*. Теперь достаточно щелкнуть в начальную и конечную точки, и в окне отобразится расстояние между ними. Если последовательно щелкать по карте, проводя ломаную линию, суммарная длина всех отрезков отобразится во второй строке.

Обратим внимание на кнопку изменения режима показа, на которой нарисован вопросительный знак. Щелчок по ней вызывает появление диалогового окна, в котором можно изменить масштаб карты и положение центра окна. А в целом настройка режима показа осуществляется через команды пункта меню *Карта*.

4.3. Векторизация географических объектов

При векторизации географических объектов фиксируются только точки. Все остальные геометрические фигуры образуются по некоторым правилам. Так, ГИС MapInfo автоматически можно строить прямые и ломаные линии. Особым графическим примитивом является замкнутая ломаная линия, многоугольник. Эта геометрическая фигура может представляться и как замкнутая граница, и как площадь, ограниченная контуром. По умолчанию многоугольник – это площадная фигура, которую можно преобразовать в замкнутую ломаную. Дополнительной линейной фигурой в MapInfo является дуга. В общем случае это фрагмент параболы, но когда в процессе рисования придерживается клавиша *Shift*, дуга становится фрагментом окружности. При редактировании такую фигуру можно только переносить и деформировать.

Как правило, при векторизации этот графический примитив не используется.

При добавлении графической информации вначале создается новый слой (*Файл/Новая таблица/Добавить к карте*). Для добавления нового поля щелкаем кнопку *Добавить поле*. Закончив ввод информации, щелкаем кнопку *Создать*. Используя появившееся окно сохранения файла, сохраним таблицу.

Теперь, используя кнопку вызова окна управления слоями, установим режим редактирования нового слоя. Техника векторизации несложна. Объявив фигуру соответствующей кнопкой, щелкаем начальную, промежуточные и конечную точки левой кнопкой мыши. Завершается процедура клавишей *Esc* или двойным щелчком левой кнопки мыши.

Иногда линейные объекты удобно векторизовать по частям, которые потом можно объединить. Для этого при нажатой клавише *Shift* выделяются объединяемые объекты, затем выбираются команды меню *Объекты/Комбинация*. В данном диалоге предлагается задать правила заполнения полей в новой записи таблицы БД, например суммировать показатели. Нажатием на кнопку *OK* операция завершается.

Аналогичным образом можно разделять один объект на несколько или удалять части перекрывающихся объектов. Для этого используют пункты меню *Объект/Выбрать изменяемый объект, Объект/Разрезать* или *Объект/Удалить часть.*

Обводка замкнутых контуров в MapInfo производится инструментом *Многоугольник*. Техника его применения не отличается от векторизации линейных объектов. Но при завершении обводки двойной щелчок мыши производит автоматическое замыкание. Трудности возникают при создании примыкающих, пересекающихся и вкладывающихся контуров. Такие задачи решаются использованием *Изменяемых объектов*.

4.4. Импорт данных из внешних БД

Для импорта данных в среду MapInfo путем прямого доступа к внешним БД выбирается команда *Файл/Открыть таблицу*. Прямой доступ возможен к файлам типа DBF, XLS и MDB, а также текстовым. В диалоговом окне выбирается нужный файл и при необходимости нужная таблица, далее этот файл переводится во внутренний формат системы командой меню *Файл/Сохранить копию*. Вновь созданный файл заново загружается и преобразовывается (*Таблица/Изменить*

/Перестроить). В диалоговом окне уточняются названия полей, типовые характеристики и проекция. После завершения таблицу можно добавлять в список слоев карты.

Следующий этап – картографирование точечных объектов. Необходимые настройки вызываются командой меню *Таблица/Создать точечные объекты*. В открывшемся диалоговом окне указываются имя таблицы, графические свойства условного знака, имена полей БД, из которых должны считываться координаты *x* и *y*, масштабные коэффициенты по осям (по умолчанию равные 1) и определяется проекция. После завершения этой процедуры на карте появятся соответствующие графические образы.

Отметим, что если к точечным объектам приурочены результаты режимных наблюдений, точкам может соответствовать не единственная запись. Например, один слой может отражать режим водного питания, другой – механический состав грунтов, третий – характеристики отложений и т.д.

4.5. Выбор объектов и получение информации

Одно из преимуществ работы с ГИС заключается в способности организовывать и группировать данные. Типичными задачами являются также организация запросов и создание вычисляемых полей.

Для выбора объектов доступного слоя используют инструмент *Стрелка*. Он считается выбранным в начале сеанса работы и имеет форму стрелки в окнах списков, карт и отчетов. Объекты можно выбирать по одному или группами.

Чтобы выбрать отдельный объект, следует выбрать инструмент *Стрелка* на панели *Операции* и указать на любой объект (например, на город). Чтобы выделить несколько объектов, следует удерживать нажатой клавишу *Shift*.

Объекты, выбранные на карте, сохраняются в виде временной таблицы, и их можно просмотреть в окне списка. Для этого следует перейти в пункт меню *Окно/Новый список*, после чего появится диалоговое окно *Список*. Выберем позицию *Selection* (Выборка) и нажмем кнопку *OK*. Появится окно *Список* для текущей таблицы выборки. Чтобы отменить выбор объектов по одному, указываем на них, удерживая нажатой клавишу *Shift*. Чтобы отменить выбор всех объектов, достаточно указать на любое место карты. Следует помнить, что с помощью инструмента *Стрелка* нельзя выбирать объекты одновременно из нескольких слоев.

С помощью инструмента *Выбор в круге* можно выполнять простейший геоанализ. Так, чтобы найти список городов, расположенных от выделенного города не далее заданного расстояния, будем плавно перемещать мышь в сторону до тех пор, пока в строке сообщений внизу слева не появится нужная цифра. Отпустим кнопку мыши, и MapInfo выберет все города, лежащие в радиусе 300 км.

Все выбранные города будут помещены во временную таблицу с названием *Selection* (Выборка). Чтобы ее просмотреть, обратимся к пункту меню *Окно/Новый список*. Появится диалоговое окно *Список*. Выберем позицию *Selection* и нажмем кнопку *OK*. Выбранные в круге записи будут показаны в окне списка с названием *Query 1* (Запрос 1).

Подобным образом можно использовать и инструменты *Выбор в рамке, Выбор в полигоне* и *Выбор в области.* В последнем случае можно, например, выбирать города, лежащие в пределах одной ломаной замкнутой линии.

Техника составления запросов в БД MapInfo типична для современных СУБД. Обращение к таблицам запросов выполняется по пунктам меню *Запрос/SQL-запрос*. В данном окне определяется таблица и составляется логическое выражение либо с клавиатуры, либо с помощью шаблонов. В качестве примеров рекомендуется воспользоваться простейшими запросами для поиска записей (то же, что и объекта на карте) типа: name = «переходное болото»; name < 2,55 и т.д., где name – имя поля таблицы. При включенной опции *Результат в список* запрос представляется отдельной таблицей. Готовый запрос можно сохранить на жестком диске в файле с расширением qry. Загрузить сохраненный запрос в систему можно через пункт меню *Запрос/Выбрать* и далее щелчком по кнопке *Загрузить*.

Как и в любом SQL-запросе, можно создавать вычисляемые поля. Например, записав в верхнем окне городское_1995/sum_1995 *100, получим количество городского населения в процентах.

Можно также применять групповые операции с использованием поля со списком *Обобщения* в диалоговом окне *SQL-запрос*.

4.6. Отображение гидрогеологических данных на карте

Для графического отображения на карте числовой информации MapInfo располагает рядом способов генерирования картограмм, картодиаграмм и т.д. Для этого нужно обратиться к пункту меню *Карта/Создать тематическую карту*. В трех последовательно появляю-

щихся диалоговых окнах (аналог мастера диаграмм MS Excel) создаем тематическую карту. На первом шаге определяем тип карты, на втором – таблицу и поля для построения, на третьем настраиваем легенду и стиль отображения. В результате создаются окно легенды и тематическая карта.

Библиотека способов разделена на разделы: Диапазоны, Столбчатая, Pie Charts, Круговая, Значки, Плотность точек, Поверхность, Отдельные значения. Каждый раздел содержит от 2 до 15 способов оформления карт. Использование того или иного приема сопровождается заполнением соответствующих диалоговых окон. Методология работы с ними общая: сначала выбирается способ, далее указываются имена таблицы и полей, затем регулируются цветовые характеристики, стили символов и т.д.

Таким образом, например, можно построить тематические карты по минерализации подземных вод (Диапазоны), их химическому составу (Круговая диаграмма), стратиграфическим подразделениям водоносных горизонтов (Отдельные значения) и т.д.

Созданные условные обозначения шкал автоматически помещаются в список слоев, видимый в окне *Управление слоями*. Если это окно активно, то двойным щелчком мыши вызывается окно редактирования способа изображения. Такое состояние карты сохраняется в течение одного сеанса работы.

Аналогично числовые поля таблиц БД можно использовать и для построения графиков. Данные графические средства иллюстрации числовой информации в ГИС оформляются самостоятельными окнами. Конструктор графиков вызывается пунктами меню *Окно/Новый график* (или F4). MapInfo предлагает следующие типы построений: 3D, Гистограммы, Колонки, Круговые, Линейные, Площадные, Поверхности, Пузырьковые, Столбчатые, Точечные. Результаты оформления данных графическими функциями сохраняются в течение одного сеанса работы.

Создание легенды начинается с выбора пунктов меню *Карта/Создать легенду*. Затем нужно отобрать необходимые слои, включаемые в легенду, уточнить параметры оформления (задать шрифты, положение листа и т.п.), а также оформить основные заголовки. После того как будет нажата клавиша *Завершить*, на экране появится легенда. Каждый элемент легенды редактируется после двойного щелчка по левой кнопке мыши.

Рекомендуется легенду строить по всем тематическим слоям, включая и топоосновы. При таком порядке работы ни один элемент изображения пропущен не будет. Вместе с тем всегда остается возможность удалить из легенды условные знаки, не определяющие содержание тематических карт. Следует иметь в виду, что при таком способе построения легенда сохраняется в течение одного сеанса работы.

4.7. Создание отчета

Рабочий набор – это не только определенный порядок данных, но и способ их хранения. Сама процедура записи рабочего набора включает в себя всего одно действие File – Save Workspace, после чего в стандартном для Windows окне задается его имя. В рабочем наборе сохраняются все редакционные правки, выполненные в косметическом слое, легенда, графики, способы картографического изображения шкал.

Компоновка материалов в составе рабочего набора представляет собой вариант тематической карты, выводимый на печать. Таким образом, он должен готовиться заранее, параллельно с набором данных. Вместе с тем легенда готовится непосредственно перед печатью, так как любые более поздние редакционные правки на карте в легенде автоматически не отразятся.

Организация данных для печати начинается с пунктов меню *Окно/Новый отчет* (или F5). В появившемся диалоговом окне указывается, какие открытые на экране документы переместить в отчет. Выбрав нужные из них или все, клавишей *OK* вызывается на экран проект печатного документа. Чтобы задать количество листов для большой по размеру карты по высоте и ширине, необходимо вызвать по меню *Отчет/Режимы показа* дополнительное окно с установками параметров печатаемого документа. Масштаб карты задается в диалоговом окне управления соотношениями элементов изображения. Оно вызывается двойным щелчком по левой кнопке мыши при наведении курсора на карту. Таким же образом регулируются точные размеры любого фрагмента компонуемого материала.

Редактирование проекта отчета обеспечено теми же средствами, что и любого тематического слоя. Временные и вспомогательные записи рекомендуется выполнять именно в этом документе. Кроме того, в проект можно поместить врезку, используя инструмент *Выбор в рамке*. Чтобы сохранить проект отчета, его следует включить в рабочий набор.

Литература

1. *Требования* к цифровым гидрогеологическим картам масштаба 1:1 000 000 и 1:200 000 и разрезам к ним// Создание гидрогеологических карт с применением компьютерных технологий (методические материалы). М.: Министерство природных ресурсов РФ, 2001. С. 71–102.

2. *Лурье И. К.* Основы геоинформатики и создание ГИС. М.: Изд-во Моск. ун-та, 2002.

3. *Растоскуев В. В.* Информационные технологии экологической безопасности// www.ecosafe.nw.ru.

4. *Сеннов А. С.* Access 2003: Учеб. курс. СПб.: Питер, 2005.

5. *Тикунов С. В.* Географические информационные системы: сущность, структура, перспективы // Итоги науки и техники. Сер. Картография. М.: ВИНИТИ, 1991.

Содержание

Учебное издание

Андрей Светозарович Сеннов
Алексей Аркадьевич Шварц

ГЕОИНФОРМАЦИОННЫЕ СИСТЕМЫ В ГИДРОГЕОЛОГИИ

Учебное пособие

Зав. редакцией *Г. И. Чередниченко*
Редактор *Э. А. Горелик*
Технический редактор *Л. Н. Иванова*
Обложка *Е. В. Калининой*
Компьютерная верстка *А. А. Шварца*

Подписано в печать с оригинал-макета 10.06.2005.
Печать офсетная. Формат 60×84 1/16. Усл. печ. л. 3,72. Уч.-изд. л. 3,87.
Тираж 60 экз. Заказ №

РОПИ СПбГУ
199034, С.-Петербург, Университетская наб., 7/9.
Типография Издательства СПбГУ.
199061, С.-Петербург, Средний пр., 41.

Для записей