МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное автономное образовательное
учреждение высшего образования
«СЕВАСТОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт ядерной энергии и промышленности

Кафедра систем контроля и управления АС

С.А. Качур, А.А. Чуклин, Н.В. Шахова

ПРОЕКТИРОВАНИЕ РЕЛЯЦИОННЫХ БАЗ ДАННЫХ С ИСПОЛЬЗОВАНИЕМ СИСТЕМЫ MICROSOFT ACCESS

Учебно-методическое пособие
для выполнения расчетно-графической работы
по дисциплине «Информационные системы и комплексы»

Севастополь
СевГУ
2017

УДК 004.9 (07)
К12

Р е ц е н з е н т ы:
С.Т. Мирошниченко – зав. кафедрой ПТУ, к.т.н., доцент
К.Б. Матузаев – зав. кафедрой ЯЭУ, к.т.н., доцент

Качур С.А.

К12 Проектирование реляционных баз данных с использованием системы Microsoft Access: учеб.-метод. пособие / С.А. Качур, А.А. Чуклин, Н.В. Шахова. – Севастополь: СевГУ, 2017. – 32 с.; ил.

Рассмотрен подход к проектированию, созданию и манипулированию реляционной базой данных. Описаны возможности программного обеспечения Microsoft Access для реализации поставленной цели. Пособие предназначено для студентов очной и заочной форм обучения направления подготовки 14.05.02 «Атомные станции: проектирование, эксплуатация и инжиниринг» очной и заочной форм обучения.

УДК 004.9 (07)

Рекомендовано к изданию в качестве учебно-методического пособия на заседании кафедры «Системы контроля и управления АС», протокол № 2 от 6 сентября 2016 г.

ОГЛАВЛЕНИЕ

1. СОДЕРЖАНИЕ И ПОСЛЕДОВАТЕЛЬНОСТЬ ВЫПОЛНЕНИЯ РАБОТЫ

Тема: проектирование, создание и манипулирование реляционной базой данных с использованием возможностей программной среды Microsoft Access.

Задание: спроектировать на основании информации, представленной в виде иерархической структуры, реляционную базу данных и получить необходимую информацию из этой базы, используя средства программного обеспечения Microsoft Access.

Цель:

1. Научиться проектировать реляционные базы данных и получать информацию из них, использую язык манипулирования данными SQL.
2. Более глубоко и всесторонне изучить возможности работы существующих СУБД, в частности Access из комплекта Microsoft Office.

В результате выполнения расчетно–графической работы студент должен:

- ***знать*** методы проектирования реляционных баз данных и языки манипулирования ими, положенные в основу организации баз данных программного обеспечения ИВС ЯЭУ;
- ***уметь*** получать необходимую информацию из базы данных с помощью SQL-запросов.

Расчетно-графическую работу рекомендуется выполнять ***в следующей последовательности:***

1. Провести логическое проектирование реляционной базы данных на основе иерархической структуры, представленной в приложении Б в соответствии с заданным преподавателем вариантом.
2. Сформулировать к созданной базе данных 10 запросов.
3. Спроектировать в системе Microsoft Access разработанную базу данных.
4. Заполнить базу данных в системе Microsoft Access информацией, соответствующей 10 записям отношения, соответствующего корню исходной иерархической структуры
5. Реализовать созданные запросы к базе данных, используя средства Microsoft Access.
6. Произвести оформление работы в соответствии с примером Приложения А.

2. ОБЩИЕ ПОЛОЖЕНИЯ

В целом база данных (data base) представляет из себя набор бланков, точнее электронных карточек, в которых есть постоянные элементы – заголовки подлежащих заполнению областей (полей) и переменные (содержимое этих полей).

Создание и работа с базами данных осуществляется при помощи систем управления базами данных (СУБД, в англоязычной литературе это называется DBMS), таких как Access из комплекта Microsoft Office, dBase, FoxPro, Paradox, Clipper, Clarion.

Microsoft Access относится к так называемым реляционным (от английского слова relation – отношение) СУБД.

Реляционная база данных представляет из себя совокупность таблиц, связанных между собой определенными отношениями и предназначенных для хранения данных. Отношения между таблицами являются существенной частью этой модели данных.

Основу реляционной модели данных составляют таблицы, которые содержат данные об однотипных моделях. Таблица реляционной базы данных состоит из множества строк и столбцов. Каждая строка таблицы содержит данные об одном объекте и называется записью. Все записи имеют одинаковую структуру – они состоят из полей, в которых хранятся атрибуты (свойства) объекта. Каждое поле записи содержит некоторое свойство представляемого объекта. Все записи имеют одни и те же поля, поэтому каждый столбец таблицы содержит значения одного и того же свойства объектов, представляемых таблицей. А это значит, что данные в ячейках одного столбца должны быть одного типа, и в этом коренное отличие таблиц реляционной базы данных от таблиц Excel.

Допустим, таблица хранит данные о людях – сотрудниках предприятия, студентах или пациентах поликлиники. Один из столбцов таблицы может содержать дату рождения каждого человека. В этом случае столбец будет иметь тип «дата». Это означает, что в нем не может быть никакой другой информации. Другой столбец, содержащий, например, вес человека, будет иметь числовой тип данных и не может хранить ничего, кроме чисел. Можно сказать, что в таблице реляционной базы данных количество полей в каждой строке одинаковое, в каждом столбце храниться однотипная информация, в каждой строке храниться информация только об одном объекте, причем в других строках этих сведений быть не может.

3. ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ РЕЛЯЦИОННОЙ СТРУКТУРЫ БАЗЫ ДАННЫХ ИЗ ИЕРАРХИЧЕСКОЙ СТРУКТУРЫ

Процесс преобразования древовидной структуры в реляционцую называется нормализацией, т.е. свертыванием исходной структуры в двумерные таблицы со столбцами простейшей структуры (элементы столбцов должны быть атомарными). При этом получается так называемая 1-я нормальная форма (1НФ).

Метод нормализации был разработан Коддом. Существует 7 нормальных форм. Каждая следующая НФ получается из предыдущей путем уменьшения избыточности.

Рассмотрим процесс получения 1-ой НФ. Основой его является формирование ключей.

Ключ должен обладать двумя свойствами

1) Однозначная идентификация кортежа: кортеж должен однозначно определяться значением ключа.

2) Отсутствие избыточности: никакой атрибут нельзя удалить, не нарушая при этом свойства однозначной идентификации. Может быть несколько наборов атрибутов, обладающих такими свойствами – это возможные ключи.

Правило формирования ключа

Ключ некоторого отношения должен содержать ключ отношения, соответствующего вышестоящей вершине в дереве, если он уже не является атрибутом этого отношения. Если при этом не выполняется 2-е свойство ключей, то ключ предшествующего отношения рассматривается как атрибут последующего.

Примечание: у каждого отношения свой ключ – результат нормализации. Если есть отношения с одинаковым ключом, то его необходимо объединить, хотя этого при грамотном проектировании не должно быть.

4. СОЗДАНИЕ БАЗЫ ДАННЫХ В MICROSOFT ACCESS

Для создания базы данных в Access нам потребуется последовательно выполнить в строке меню следующие команды «Файл»=> «Создать» => «Новая база данных». В строке «Имя файла» ввести название

новой базы данных и щелкнуть левой кнопкой мыши по кнопке «Открыть». В результате появится окно «База данных». Здесь мы и будем создавать необходимые таблицы.

Таблицы в Access можно создавать тремя способами:

1) путем ввода данных. Такие таблицы называются справочниками и используются для того, чтобы исключить многократный ввод и хранение в базе данных одних и тех же значений.

2) с помощью мастера;

3) в режиме конструктора.

Создадим таблицу в режиме конструктора. Это самый сложный способ создания таблиц, но зато он позволяет точно определить все свойства таблицы и её полей.

Итак, дважды щелкните левой кнопкой мыши на ярлыке «Создание таблицы с помощью конструктора». Откроется пустое окно «Конструктор таблиц». В верхней части окна Конструктора находиться таблица, состоящая из трех столбцов: в столбце «Имя поля» вводятся имена полей создаваемой таблицы, в столбце «Тип данных» нужно задать тип данных для каждого поля. Щелкните на кнопке со стрелкой в правой части столбца – появится список типов данных, которые могут быть использованы в таблицах Access (текстовый, логический, счетчик, дата/время и др.). В столбце «Описание» можно ввести описания полей.

В нижней части окна задаются свойства полей таблицы:

1) «Размер поля» определяет максимальное количество символов, которое можно ввести в это поле;

2) «Новые значения» определяет способ генерации значений этого поля для новых записей в таблице;

3) «Формат поля» указывает, в каком формате будут вводиться данные при просмотре таблицы;

4) «Подпись» позволяет определить заголовок поля, который будет выводиться в разных режимах, при просмотре таблицы, печати отчетов и т.д.;

5) «Индексированное поле» позволяет указать необходимость создания индекса по данному полю. Индексы используются с целью ускорения поиска и сортировки в любой базе данных.

Прежде чем сохранить таблицу, необходимо определить в ней ключевое поле. Чтобы сделать поле ключевым, выделите его, установив курсор в любом месте описания поля, затем найдите на панели инструментов кнопку с изображением ключика и щелкните на ней. При необходимости определить ключевое поле в уже существующей таблице, найдите в ней поле, которое содержит уникальные значения. Ес-

ли выбранное поле содержит повторяющиеся или пустые значения, то его нельзя определить как ключевое.

После создания таблицы нужно закрыть окно «Конструктор таблиц». Для этого нажмите кнопку закрытия окна. На вопрос о сохранении изменений макета или структуры таблицы ответить да. В окне «Сохранить как» в поле «Имя таблицы» введите имя создаваемой таблицы и нажмите кнопку «ОК».

4.1. Ключевые поля

Каждая таблица должна иметь один или несколько столбцов (атрибутов), которые однозначно идентифицируют каждый объект в таблице, то есть позволяют четко отличить один объект от другого. Такие столбцы образуют первичный ключ (primary key), и если столбцов несколько, то говорят, что первичный ключ является составным. Поле, представляющее первичный ключ или являющееся частью первичного ключа, называется ключевым полем.

Чтобы поле сделать ключевым следует выделить его, установив курсор в любом месте строки описания поля, затем найти на панели инструментов кнопку с изображением ключика и щелкнуть на ней.

Если выбранное поле содержит повторяющиеся или пустые значения, то его нельзя определить как ключевое. Нужно сначала устранить повторы путем изменения значений. Если устранить невозможно, следует либо добавить в таблицу поле счетчика и сделать его ключевым, либо определить составной ключ.

4.2. Связи между таблицами

В реляционной модели очень важным является понятие связи между таблицами. Связь – это логическое отношение между объектами, представленными таблицами. Связь между записями двух таблиц обычно основана на совпадении значений атрибутов, по которым эта связь устанавливается.

Типы связей между таблицами:

1) отношение «один-ко-многим» – одной записи в таблице №1 соответствует несколько записей в таблице №2. На стороне «один» находится таблица №1, а на стороне «многие» - таблица №2. При этом таблицу №1 принято называть главной, а таблицу №2 – подчиненной.

2) отношение «многие-ко-многим» – нескольким записям в таблице №1 соответствуют несколько записей в таблице №2.

3) отношение «один-к-одному» – одной записи в таблице №1 соответствует только одна запись в таблице №2.

Связи можно создавать, когда уже созданы таблицы. Удобнее всего это сделать на схеме данных.

1) Чтобы открыть схему данных, выполните команды меню «Сервис» => «Схема данных». Откроется диалоговое окно «Схема данных», в котором могут быть изображены некоторые таблицы.

2) Нужно отобразить на схеме все наши таблицы, так как они все связаны между собой. Для этого щелкните правой кнопкой мыши на свободном поле на схеме и из контекстного меню выберите команду «Добавить таблицу».

3) Появиться диалоговое окно «Добавление таблицы» со списком всех таблиц в базе данных. Выделите в этом списке, пользуясь клавишей Ctrl, те таблицы, которые отсутствуют на схеме, и нажмите кнопку «Добавить».

4) По окончании закройте окно «Добавление таблицы», щелкнув на кнопке «Закрыть».

5) Подведите курсор мыши к тому полю, от которого устанавливается связь, и перетащите это поле к таблице, с которой будет устанавливаться связь. Отпустите кнопку мыши, когда указатель окажется над полем, с которым устанавливается связь. Появится диалоговое окно «Изменение связей».

6) В первой строке таблицы отображаются связанные поля. Установите флажки «Обеспечение целостности данных», «Каскадное обновление связанных полей» и «Каскадное удаление связанных полей». Нажмите кнопку «Создать».

Окно закроется, а на схеме данных появится линия, соединяющая две таблицы.

5. ЯЗЫК SQL

Язык SQL является языком манипулирования для реляционных баз данных.

В SQL реализованы три функции манипулирования данными: 1) определение; 2)выборка (запрос); 3) обновление.

5.1. Оператор выбора языка SQL

Рассмотрим оператор выбора SELECT.

Назначение

Возвращает данные из одной или нескольких баз данных.

Формат

SELECT [ALL | DISTINCT]
[<alias>.]<select_item> [AS <column_name>]
[, [<alias>.]<select_item> [AS <column_name>] ...]
FROM <database>[<local_alias>] [,<database> [<local_alias>] ...]
[WHERE <joincondition> [AND <joincondition> ...]
[AND | OR <filtercondition> [AND | OR <filtercondition> ...]]]
[GROUP BY <groupcolumn> [, <groupcolumn> ...]]
[HAVING <filtercondition>]
[UNION [ALL] <SELECT command>]
[ORDER BY <order_item>[ASC | DESC]
[, <order_item>[ASC | DESC]...]]

Описание

SELECT является командой SQL, и используется для нахождения и возврата данных из одной или нескольких баз данных. Когда Вы используете команду SELECT для посылки запроса, FoxPro интерпретирует запрос и возвращает данные из баз(ы) данных. Команда SELECT встроена в Access подобно остальным командам.

Дополнительные опции

SELECT [ALL | DISTINCT]
[<alias>.]<select_item> [AS <column_name>]
[, [<alias>.]<select_item> [AS <column_name>] ...]

SELECT опция запрашивает и специфицирует Поля базы данных, константы и выражения, появляющиеся в результате запроса.

По умолчанию, все строки полученные в результате запроса будут выведены на экран. Для исключения появления одинаковых строк в качестве результата запроса включите опцию DISTINCT.

Каждый <select_item> генерирует одну колонку в качестве результата запроса. Если более одного <select_item> имеют одинаковое имя, Вы должны быть уверены, что для квалификации имени было включено имя псевдонима базы данных или перед <select_item>.

<select_item> может быть:

- полем базы данных в опции FROM
- константой, указывающей что одно и тоже значение появляется в каждой строке результата запроса
- выражением, которое может включать функции.

Следующие функции допустимы для использования с <select_item>, они являются полями или выражениями включающими поля:

- AVG(<select_item>) – среднее по колонке числовых данных.
- COUNT(<select_item>) – счетчик <select_items> в колонке.

- COUNT(*) – счетчик числа строк в выходе запроса.

- MIN(<select_item>) – определяет наименьшее значение <select_item> в колонке.

- MAX(<select_item>) – определяет наибольшее значение <select_item> в колонке.

- SUM(<select_item>) – сумма по колонке числовых данных.

Необязательная опция AS <column_name> указывает название колонки в выходе запроса. Это используется, когда <select_item> является выражением или содержит функцию и Вы хотите задать название, имеющее смысл. <column_name> может быть выражением, но не должно содержать символов (например, пробелов) которые недопустимы в именах полей баз данных.

FROM <database> [<local_alias>] [, <database> [<local_alias>] ...]

FROM – обязательная опция, содержащая список баз данных, в которых находятся данные извлекаемые в результате запроса. Если база данных не открыта в рабочей области, но находится в текущей директории или в директории, описанной в пути Access, появляется окно диалога "Открытие файла", через которое можно выбрать базу данных.

Если база данных не была открыта в рабочей области, она открывается и остается открытой пока не выполнится запрос.

<local_alias> – это временное имя для <database>. Если Вы указываете <local_alias>, Вы должны указывать <local_alias> вместо имени базы данных везде в команде SELECT. Если несколько баз данных имеют поля с одинаковыми именами необходимо квалифицировать их, для чего может быть использован короткий <local_alias>.

При выводе результатов запроса, колонкам присваиваются названия в соответствии со следующими правилами:

1) Если <select_item> – поле с уникальным именем, выходная колонка имеет тоже имя.

2) Если более чем один <select_item> имеет одинаковое имя, например CUST.ZIP и STATE.ZIP, выходные колонки будут иметь имена ZIP_A и ZIP_B.

3) Если <select_item> – выражение, то выходная колонка именуется как EXP_A. Другие выражения именуются EXP_B, EXP_C, и так далее.

4) Если <select_item> содержит функцию, такую как COUNT(), выходная колонка имеет имя COUNT_A. Другой <select_item> содержащий функцию SUM() на выходе имеет имя SUM_B.

[WHERE <joincondition> [AND <joincondition> ...]

[AND | OR <filtercondition> [AND | OR <filtercondition>...]]]

Запрашивает данные из нескольких баз данных. Сообщает FoxPro на необходимость включения в результат запроса только некоторых записей. <joincondition> указывает поля, которые связывают базы данных в опции FROM. Если Вы указываете более одной базы данных, то необходимо указать условие связи для каждой базы данных после первой.

Внимание

Если Вы включаете две базы данных в запрос и не указываете условие связи. Каждая запись в первой базе данных будет связана с каждой записью во второй базе данных пока выполняется условие фильтрации. Это может породить огромный результат запроса.

Будьте внимательны при объединении баз данных с пустыми полями, так как Access допускает пустые поля. Так например при связывании двух баз данных имеющих пустые поля в 100 и 400 записях соответственно, результат запроса будет содержать 40.000 записей из пустых полей. Для того, чтобы обойти это, используйте функцию EMPTY().

При нескольких условиях базы должны объеденяться с использованием оператора AND.

Каждое <joinconditions> имеет форму:

<field1> <сравнение> <field2>

где <field1> – поле из одной базы данных, <field2> - поле из второй базы данных и <сравнение> принимает одно из следующих значений: =, < >, !=, #, = =, >, > =, <, < =.

<filtercondition> указывает условие, на основании которого записи включаются в результат запроса. Запрос может включать несколько условий фильтра, связанных операторами AND или OR. Можно использовать также оператор NOT или функцию EMPTY() для проверки пустых полей. Условие фильтра может иметь одну из следующих форм:

Форма: <field1> <comparison> <field2>

Пример: customer.cust_id = payments.cust_id

Форма: <field> <comparison> <expression>

Пример: payments.amount >= 1000

Форма: <field> <comparison> ALL (<subquery>)

Пример: taxrate < ALL ;

(SELECT taxrate FROM customer WHERE state = 'MI')

Когда фильтр включает ключевое слово ALL, <field> должно встретиться в условии сравнения для каждого значения сгенерированного подзапросом перед включением записи в результат запроса.

Форма: <field> <comparison> ANY | SOME (<subquery>)

Пример: taxrate < ANY ;

(SELECT taxrate FROM customer WHERE state = 'MI')

Когда фильтр включает ключевое слово ANY или SOME, <field> должно встретиться в условии сравнения хотя бы для одного значения сгенерированного подзапросом перед включением записи в результат запроса.

Форма: <field> [NOT] BETWEEN <start_range> AND <end_range>

Пример: customer.taxrate BETWEEN 5.50 AND 6.00

Проверка, лежит ли значение в диапазоне.

Форма: [NOT] EXISTS (<subquery>)

Пример: EXISTS ;

(SELECT * FROM invoice WHERE customer.zip = invoice.zip)

Проверка, удовлетворяет ли по меньшей мере одна колонка условию. Когда фильтр включает EXISTS, условие истинно пока подзапрос не станет пустым.

Форма: <field> [NOT] IN <value_set>

Пример: customer.zip NOT IN ('43411','43506','43667')

<field> -является частью набора значений, перед включением записи в результат запроса.

Форма: <field> [NOT] IN (<subquery>)

Пример: customer.cust_id IN ;

(SELECT tranfile.cust_id FROM tranfile WHERE tranfile.item='Fo'

<field> – является частью набора значений возвращаемого подзапросом, перед включением записи в результат запроса.

Форма: <field> [NOT] LIKE <expC>

Пример: customer.state NOT LIKE 'OH'

Этот фильтр ищет <field> соответствие указанному <expC>. Вы можете включить символы % и _ как часть <expC>. _ – указывает единичный неопределенный символ в строке а % – указывает последовательность неопределенных символов в строке.

Подзапрос – это SELECT внутри SELECT и должен быть ограничен скобками. Вы можете организовать множественные подзапросы одного уровня (не вложенные) в опции WHERE. Подзапросы могут иметь множественные условия связи.

[GROUP BY <groupcolumn> [, <groupcolumn> ...]]

Необязательная опция которая группирует строки в запросе на основании значения в одной или более колонках. <groupcolumn> может быть ее регулярным полем базы данных, полем базы данных которое включено в SQL функцию, или числовым выражением указы-

вающим положение колонки в результирующей таблице. (Номер самой левой колонки 1).

HAVING <filtercondition>

Сообщает о необходимости включения групп в результат запроса на протяжении <filtercondition>. <filtercondition> не может содержать подзапрос.

HAVING <filtercondition> используется с GROUP BY для указания критерия, по которому группа включается в результат запроса. HAVING может включать так много условий фильтрации, как это необходимо, они связываются операторами AND или OR. Можно также использовать оператор NOT для обращения логических выражений.

HAVING без GROUP BY действует подобно опции WHERE. Вы можете использовать локальные псевдонимы и функции полей в опции HAVING.

[UNION [ALL] <SELECT command>]

Необязательная опция, комбинирует окончательный результат одной команды SELECT с окончательным результатом другой <SELECT command>. По умолчанию UNION проверяет комбинацию результатов на предмет исключения повторения строк. Для избежания исключения повторов необходимо указать ключевое слово ALL. Можно использовать скобки для комбинирования нескольких опций UNION.

UNION правила:

1) Нельзя использовать UNION для комбинации подзапросов.

2) Обе команды SELECT должны выводить одинаковое число колонок.

3) Каждая колонка в результате запрос одной команды SELECT должна иметь тот же тип данных, что и соответствующая колонка в другой <SELECT command>.

4) Только последняя <SELECT command> может иметь опцию ORDER BY и ORDER BY должна описывать выходную колонку по номеру. Если ORDER BY включена, эффект сказывается на всех результатах.

[ORDER BY <order_item>[ASC | DESC]
[, <order_item> [ASC | DESC]...]]

Сортирует результат запроса на основании одной или нескольких колонок. Каждый <order_item> соответствует колонке в результате запроса и может быть:

1) Полем из FROM базы данных , которое также является <select_item>в главной опции SELECT (не в подзапросе)

2) Числовым выражением указывающим положение колонки в результирующей таблице.

Можно указать необязательный ключ DESC , для сортировки результата в убывающем порядке в соответствии с <order_item(s)>. ASC указывает на порядок по возрастанию и принимается по умолчанию. Если опцию не указывать результат будет неупорядоченным.

5.2. Примеры запросов на языке SQL

Выдать все кампании из базы данных CUSTOMER (БД заказчиков):

```
SELECT customer.company ;
    FROM customer
```

Высветить три поля из двух баз данных связанных по полю CUST_ID. Использовать локальный псевдоним для обеих баз данных.

```
SELECT a.company, b.date, b.amount ;
    FROM customer a, payments b ;
    WHERE a.cust_id = b.cust_id
```

Высветить только те записи, где в указанных полях записи уникальны.

```
SELECT DISTINCT a.cust_id, b.date, b.amount ;
    FROM customer a, payments b ;
    WHERE a.cust_id = b.cust_id
```

Высветить несколько полей из базы данных CUSTOMER упорядоченных по возрастанию по полям STATE, ZIP и COMPANY.

```
SELECT state, zip, company, cust_id ;
    FROM customer ;
    ORDER BY state, zip, company
```

Высветить записи которые были занесены до даты 10/04/04.

```
SELECT a.cust_id, a.company, b.po ;
    FROM customer a, invoice b ;
    WHERE (a.cust_id = b.cust_id) AND (b.inv_date < {10/04/04})
```

Высветить все компании в базе данных CUSTOMER, у которых zip код совпадает с кодом zip в базе данных INVOICE.

```
SELECT company FROM customer a WHERE ;
    EXISTS (SELECT * FROM invoice b WHERE a.zip = b.zip)
```

Высветить все записи в базе данных CUSTOMER, у которых названия начинаются с большой буквы "C", длина названия значения не имеет.

```
SELECT * FROM customer a WHERE a.company LIKE 'C%'
```

Высветить все записи в базе данных CUSTOMER, у которых названия государства начинаются с большой буквы "N" со следующим неопределенным символом.

SELECT * FROM customer a WHERE a.state LIKE 'N_'

6. СОЗДАНИЕ И ВЫПОЛНЕНИЕ ЗАПРОСОВ В MICROSOFT ACCESS

Запросы используются для просмотра, анализа и изменения данных одной или нескольких таблиц.

Создавать запросы возможно двумя способами:

1) создание запроса с помощью мастера;

2) создание запроса в режиме конструктора.

Создание запроса с помощью мастера

1) В окне базы данных необходимо перейти к вкладке «Запросы» нажать кнопку «Создать».

2) В диалоговом окне «Новый» запрос выбрать «Мастера» «Простой запрос».

3) Нажать кнопку «ОК».

4) Указать имя таблицы или запроса, на котором должен быть основан создаваемый запрос, а затем выбрать поля, из которых должны быть восстановлены данные.

5) Если необходимо, то нужно указать дополнительные таблицы или запрос, а затем выбрать из них поля, которые должны быть использованы. Эти действия должны повторяться до тех пор, пока не будут выбраны все необходимые поля.

6) Требуется следовать инструкциям, выдаваемым в диалоговых окнах «Мастера». В последнем диалоговом окне пользователю предлагается выбор: выполнить запрос или просмотреть его структуру в режиме «Конструктора».

Если полученный запрос не соответствует требованиям, можно снова обратиться к «Мастеру» или внести изменения в запрос в режиме «Коструктора».

Создание запроса в режиме «Конструктора»:

1) в окне «Базы данных» необходимо перейти к вкладке «Запросы» и нажать кнопку «Создать»;

2) в диалоговом окне «Новый запрос» выбрать «Конструктор»;

3) нажать кнопку «ОК»;

4) в диалоговом окне «Добавление таблицы» выбрать таблицы или запросы, на которых будет основан создаваемый запрос;

5) нажать кнопку «Закрыть»;

6) заполнить диалоговое окно «Запрос на выборку»: выбрать поля таблиц, которые будут участвовать в запросе, определить виды сортировки и условия отбора в тех полях, где это необходимо;

7) нажать кнопку «Запуск» (красный восклицательный знак).

Если выбрать в строке меню пункт «Вид», то можно посмотреть запрос в режиме таблицы, в режиме SQL.

7. СОДЕРЖАНИЕ ПОЯСНИТЕЛЬНОЙ ЗАПИСКИ РГР

1. Титульный лист.
2. Задание на РГР в соответствии с выбранным вариантом.
3. Проектирование реляционной структуры базы данных на основе заданной иерархической.
4. Перечень сформулированных запросов к созданной базе данных.
5. Распечатка экранных форм структуры базы данных, заполненных таблиц, выполнения запросов в режиме таблиц и SQL, полученных в программной среде Microsoft Access .
6. Перечень использованной литературы.

При защите РГР необходимо представить преподавателю пояснительную записку и файл результатов, выполненных в программной среде Microsoft Access для демонстрации на компьютере.

8. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое база данных?
2. Какие существуют способы создания таблиц в Access?
3. Что такое реляционная база данных?
4. Что такое СУБД?
5. В чем заключается отличие между таблицами реляционной базой данных и таблицами Excel?
6. Для чего создается ключевое поле?
7. Что такое связи между таблицами
8. Какие существуют типы связей между таблицами?
9. Как создать новую базу данных?
10. Как создать связи между таблицами базы данных?

11. Что такое запрос?
12. Какие существуют способы создания запросов в Access?
13. Что такое подзапрос и множественные условия связи?
14. Назначение оператора SELECT?

ЛИТЕРАТУРА

1. Арте Ш. Структурный подход к организации баз данных. – М. : Финансы и статистика, 1983. – 317 с.
2. Грей П. Логика, алгебра и базы данных. / Пер.с англ. – М.: Машиностроение, 1989. –386 с.
3. Дейт К. Введение в системы баз данных. / Пер. с англ. – М.:Наука. Гл. ред. Физ. – мат. Инт. , 1980. – 464 с.
4. Дейт К. Руководство по реляционной СУБД DB2. / Пер. с англ. – М. : Финансы и статистика, 1988. – 320 с.
5. Кочаловский М.Р. Технология баз данных на персональных ЭВМ. – М. : Финансы и статистика ,1992. – 224 с.
6. Мартин Дж. Организация баз данных в вычислительных системах. / Пер. с англ. – М. :Мир, 1978. – 615 с.
7. Мейер Д. Теория реляционных баз данных.– М.:Мир, 1978. – 615 с.
8. Ульман Дж. Основы систем баз данных. – М. : Финансы и статистика, 1983. – 608 с.
9. Холл П. Вычислительные структуры. Введение в нечисленное программирование. – М . : Мир, 1978. – 214 с.
10. Цикритзис Д. , Лоховский Ф. Модели данных. – М.: Финансы и статистика, 1985. – 343 с.

Приложение А

ФРАГМЕНТ ПРИМЕРА ВЫПОЛНЕНИЯ ЗАДАНИЯ

Логическое проектирование реляционной базы данных

Дана иерархическая структура базы данных «Служащий» (рис. 1). Необходимо спроектировать реляционную базу данных.

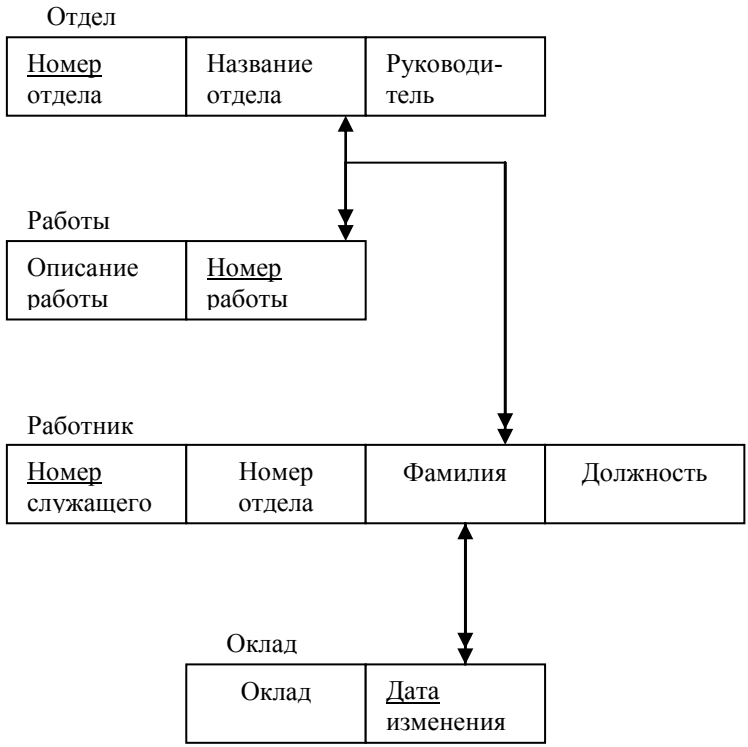
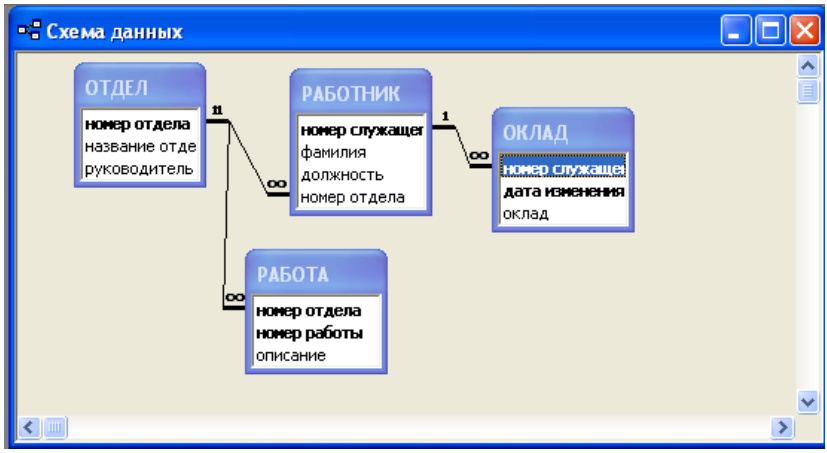


Рис. 1. Иерархическая структура база данных «Служащий»

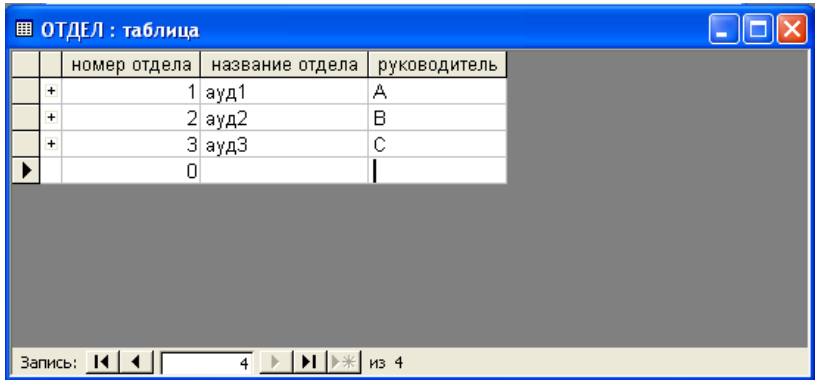
Нормализованная форма схемы:
Отдел (номер отдела, название отдела, руководитель)
Работа (номер отдела, номер работы, описание работы)
Служащий (номер служащего, имя служащего, должность номер отдела)
Изменение_оклада (номер служащего, дата изменения, оклад).

Допущения:
1. Номер служащего уникален по БД.
2. Номер работы уникален внутри отдела.

Схема реляционной базы «Служащий» данных
в программной среде Access



Экземпляры отношений реляционной базы «Служащий» данных
в программной среде Access



РАБОТА : таблица

номер отдела	номер работы	описание
1	1	быстро
1	2	медленно
2	1	быстро
2	2	медленно
3	1	быстро
3	2	медленно
0	0	

Запись: 7 из 7

РАБОТНИК : таблица

номер служащ	фамилия	должность	номер отдела
1	Иванов	инженер	1
2	Петров	инженер	2
3	Сидоров	техник	2
4	Козлов	техник	1
5	Уткин	инженер	3
6	Горбунов	техник	3
0			0

Запись: 7 из 7

ОКЛАД : таблица

номер служащ	дата изменени	оклад
1	01.02.2000	500
2	01.02.2000	400
3	01.02.2000	600
4	01.02.2003	700
5	01.02.2005	1000
6	01.02.2005	700
0		

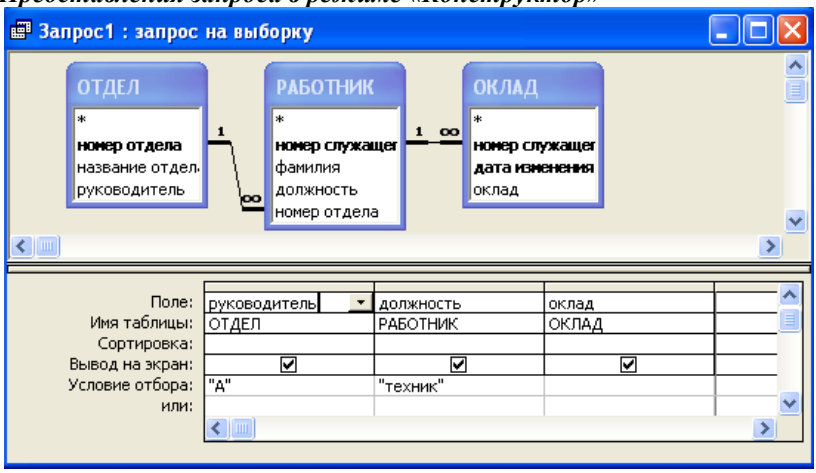
Запись: 7 из 7

Пример выполнения запроса к реляционной базе данных «Служащий» в программной среде Access

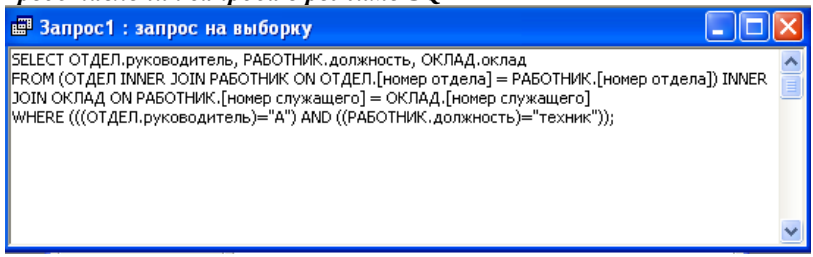
Формулировка запроса

Каков оклад у работника в должности «техник», если его руководитель «А»?

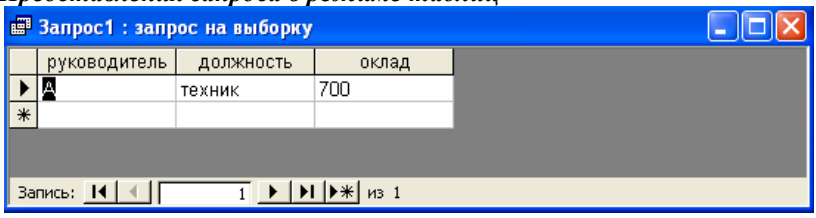
Представления запроса в режиме «Конструктор»



Представления запроса в режиме SQL



Представления запроса в режиме таблиц

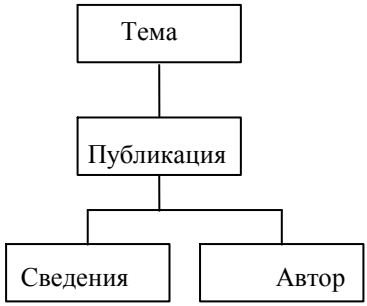


Приложение Б

ВАРИАНТЫ ЗАДАНИЙ БАЗ ДАННЫХ

Вариант 1

На рисунке изображена структура БД, которая содержит информацию о публикациях по ряду выбранных тем.

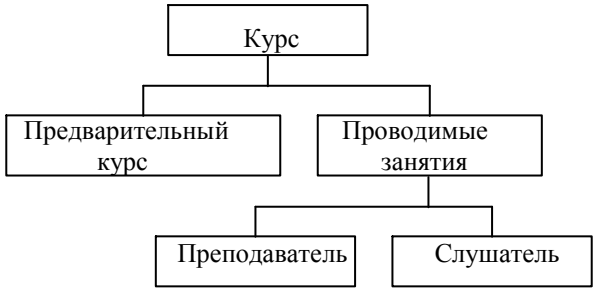


Состав информации:

- тема: номер темы (уникальный), наименование темы;
- публикация: признак типа (А – статья, М – монография), заглавие;
- сведения: дата опубликования, издательство (в случае монографии), наименование журнала с номером тома и номером выпуска (в случае статьи);
- автор: имя и адрес.

Вариант 2

На рисунке изображена структура БД, которая содержит информацию о внутренней истеме обучения в большой промышленной компании .

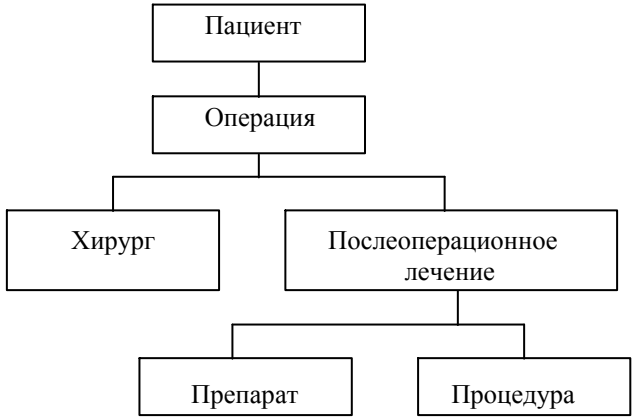


Состав информации:

- курс: номер курса (уникальный), название курса, описание курса, сведения о предварительном курсе, проводимых занятиях;
- предварительный курс: номер курса, название курса;
- проводимое занятие по данному курсу: дата, место, формат (продолжительность: полная или сокращенная);
- преподаватель: служебный номер (уникальный), имя;
- слушатель: служебный номер (уникальный), имя, оценка.

Вариант 3

На рисунке изображена структура БД, содержщая информацию о пациентах клиники.

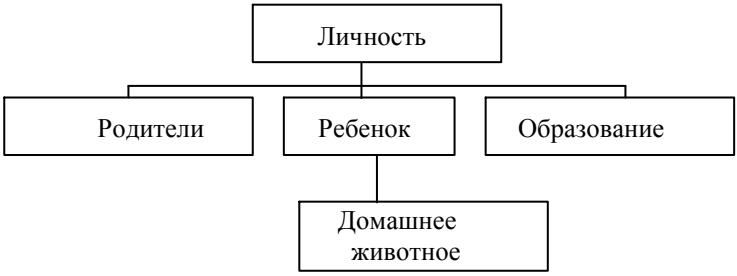


Состав информации:

- пациент: номер пациента (уникальный), его имя и адрес;
- операция: название операции, дата операции, сведения о хирурге и послеоперационном лечении;
- хирург, делавший операцию: номер патента хирурга, имя;
- послеоперационное лечение: препарат, назначенный после операции, продолжительность лечения в днях, физикотерапевтические процедуры;
- послеоперационное лечение: препарат, назначенный после операции, продолжительность лечения в днях, физикотерапевтические процедуры;
- препарат: наименование, побочный эффект;
- процедура: наименование, назначенное оборудование.

Вариант 4

На рисунке изображена структура БД, которая содержит анкетные данные служащих конторы.

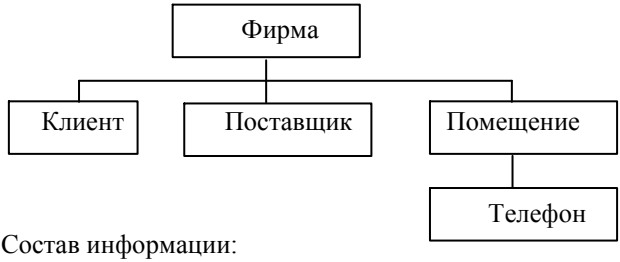


Состав информации:

- личность: имя служащего, номер служащего (уникальный), дата рождения, адрес, пол, количество детей;
- родитель: имя, год рождения, образование, место работы;
- ребенок: имя, возраст, количество животных;
- образование: наименование учебного заведения, специальность, место нахождения учебного заведения, дата поступления, дата окончания;
- домашнее животное: имя, вид.

Вариант 5

На рисунке изображена структура БД, которая содержит информацию о торговых фирмах.

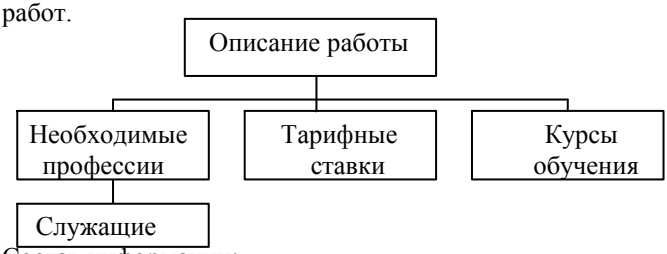


Состав информации:

- фирма: номер фирмы (уникальный), наименование, адрес, имя управляющего;
- клиент: имя, адрес, счет в банке;
- поставщик: имя, адрес, счет;
- помещение: вид (склад, магазин), площадь, адрес;
- телефон: номер телефона.

Вариант 6

На рисунке изображена структура БД, которая содержит каталог работ.

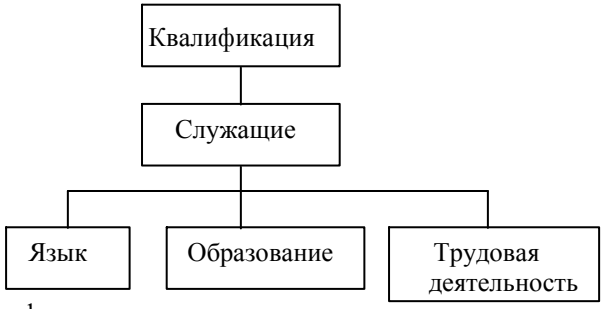


Состав информации:

- описание работы: код работы (уникальный), название работы, содержание работы, информация о необходимых профессиях, курсах обучения и тарифных ставках;
- необходимая профессия: код профессии, профессиональный уровень, описание профессии;
- тарифные ставки: коэффициент, выслуга, минимальное ставка, максимальная ставка;
- курс обучения: код курса (уникальный), описание курса;
- служащий: образование, опыт.

Вариант 7

На рисунке изображена структура БД, которая содержит информацию о квалификации служащих.



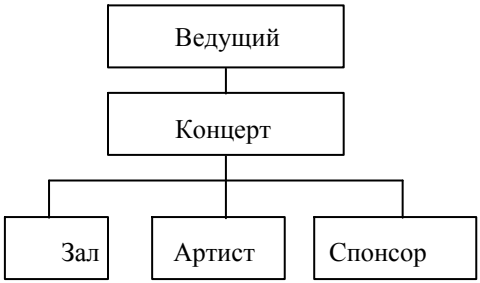
Состав информации:

- квалификация: уровень квалификации, разряд;
- служащий: имя, адрес, количество, общий стаж работы;
- образование: наименование учебного заведения или курсов, специальность, вид образования (высшее, среднее, средне-специальное, неоконченное высшее, курсы);

- трудовая деятельность: наименование предприятия, должность, место нахождения предприятия, дата поступления, дата увольнения с предприятия;
- язык (знание иностранных языков): язык, степень владения.

Вариант 8

На рисунке изображена структура БД, которая содержит информацию о концертах:

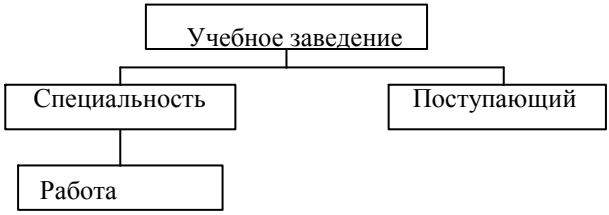


Содержание информации:

- ведущий: имя;
- концерт: дата, время, место (адрес);
- зал: наименование, адрес, количество мест, минимальная стоимость билета;
- артист: номер выступления, имя, содержание выступления;
- спонсор: наименование фирмы, суммы, счет.

Вариант 9

На рисунке изображена структура БД, которая содержит информацию об учебных заведениях и работах для использования работниками по профориентации:



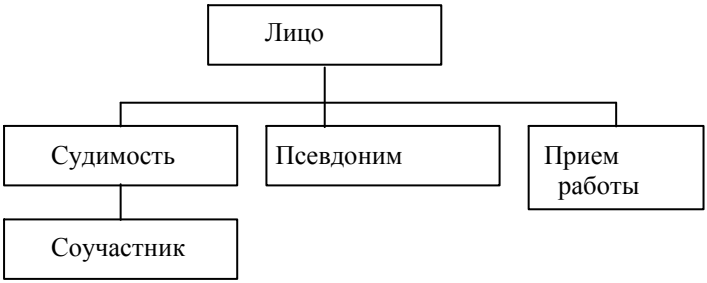
Состав информации:

- учебное заведение: наименование, место нахождения, уровень образования;

- специальность: код специальности, режим обучения, срок обучения;
- поступающий: уровень образования, пол, минимальный возраст, максимальный возраст;
- работа, которая может быть получена после окончания учебного заведения: наименование фирмы, ее место нахождения, требуемые специальности.

Вариант 10

На рисунке изображена структура БД, которая содержит картотеку осужденных лиц:

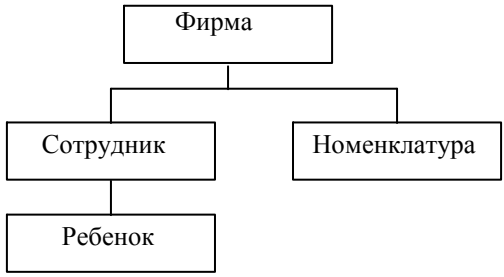


Состав информации:

- лицо: имя, дата рождения, пол, семейное положение, количество детей, особые приметы;
- судимость: статья, срок, дата осуждения, дата окончания срока, место отбывания наказания;
- псевдоним: имя (псевдоним);
- прием работы: особенности;
- соучастник: имя соучастника.

Вариант 11

На рисунке изображена структура БД, которая содержит картотеку фирм:

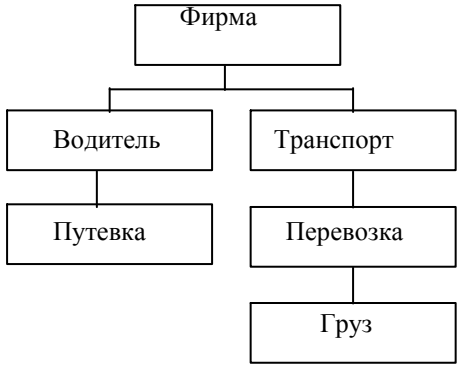


Состав информации:

- фирма: код фирмы, наименование, адрес правления, численность работающих, код головной фирмы;
- номенклатура: вид экономической деятельности, юридическая форма (акционерное общество, частное предприятие, общество с ограниченной ответственностью и т.д.), характер предприятия (цех, склад, конторы и т.д.);
- сотрудник: имя, год рождения, пол, стаж, должность, оклад;
- ребенок: имя, год рождения.

Вариант 12

На рисунке изображена структура БД, которая содержит информацию о грузовых перевозках, осуществляемых различными фирмами.

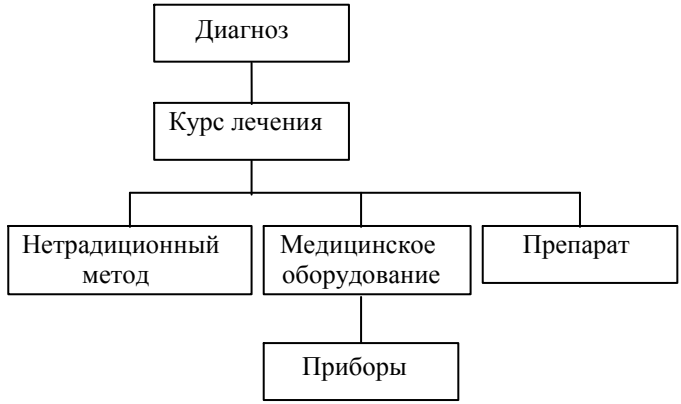


Состав информации:

- фирма: наименование, адрес, имя владельца;
- водитель: номер, имя, профессиональный уровень;
- путевка: номер путевки, дата рейса, номер транспорта;
- транспорт: номер транспорта, вид транспорта (автомобильный, ж/д, судоходный и т.д.), фирма, характеристика состояния;
- перевозка: дата, номер перевозки (= номеру путевки), наименование груза, место отправления, место назначения, тариф, расстояние, вид транспорта;
- груз: масса, стоимость, наименование груза, характер груза (твердый, жидкий и т.д.).

Вариант 13

На рисунке изображена структура БД, которая содержит информацию о лечении различных заболеваний:

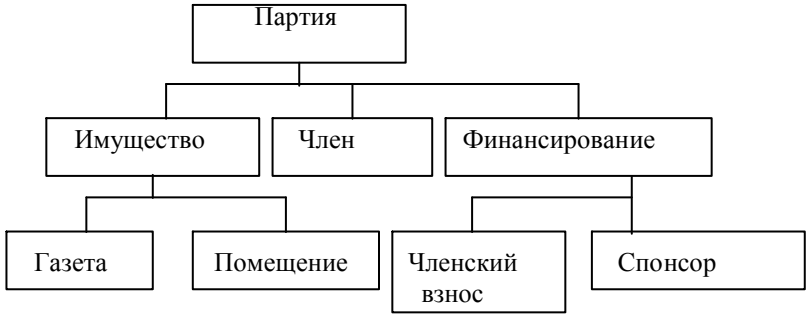


Состав информации:

- диагноз: наименование, сведения о необходимом лечении;
- курс лечения: номер курса, длительность, метод лечения, стоимость;
- медицинское оборудование: название, назначение, противопоказания;
- препарат: наименование, побочный эффект;
- нетрадиционный метод: наименование, описание метода, противопоказания;
- приборы: наименование.

Вариант 14

На рисунке изображена структура БД, содержащая информацию о политических партиях:

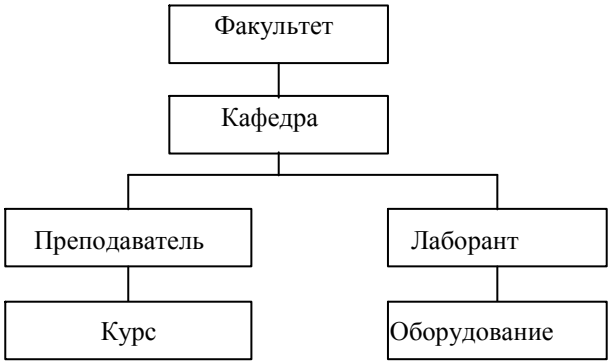


Состав информации:

- партия: регистрационный номер, дата регистрации, название, руководитель;
- член: номер членского билета, имя, год рождения, адрес, место работы, профессия;
- финансирование: наименование банка, номер счета;
- членский взнос: номер членского билета, сумма;
- спонсор: наименование организации, сумма;
- имущество: общая стоимость;
- газета: наименование, тираж, главный редактор количество сотрудников;
- помещение: адрес, площадь, телефон.

Вариант 15

На рисунке изображена структура БД, которая содержит информацию о факультетах вида:



Состав информации:

- факультет: номер факультета, наименование факультета, декан;
- кафедра: наименование, зав. Кафедрой, количество студентов, ассистентов, лаборантов, профессоров;
- преподаватель: имя, должность, оклад;
- курс: наименование курса, количество часов, преподаватель;
- оборудование: регистрационный номер, наименование оборудования, дата покупки, стоимость.

.

Качур Светлана Александровна
Чуклин Александр Алексеевич
Шахова Наталья Васильевна

У ч е б н о е и з д а н и е

ПРОЕКТИРОВАНИЕ РЕЛЯЦИОННОЙ БАЗЫ ДАННЫХ С ИСПОЛЬЗОВАНИЕМ СИСТЕМЫ MICROSOFT ACCESS

Учебно-методическое пособие
для выполнения расчетно-графической работы
по дисциплине «Информационные системы и комплексы»

Редактор – О.В. Дмитриева

Изд. № 75/17. Объем 2 п.л. Усл. печ. л. 1,86. Уч.-изд. л. 1,96.
РИИЦМ ФГАОУВО «Севастопольский государственный университет»