

**МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА (МИИТ)»
(РУТ (МИИТ))**

Одобрено кафедрой
«ЖЕЛЕЗНОДОРОЖНАЯ АВТОМАТИКА ТЕЛЕМЕХАНИКА И СВЯЗЬ»

Протокол № ____ от _____ 201__ г.

Автор: Ермакова О.П., к.тех.н, доц

**ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ С МЕТОДИЧЕСКИМИ
УКАЗАНИЯМИ**

ПО ДИСЦИПЛИНЕ

ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ

Уровень ВО: *Бакалавриат*

Форма обучения: *Заочная*

Курс: *3*

Специальность/Направление: *27.03.04 Управление в технических системах (УТб)*

Специализация/Профиль/Магистерская программа: *(УТ) Системы и технические средства автоматизации и управления*

Москва

ОБЩИЕ УКАЗАНИЯ

В курсовом проекте следует разработать приложение в среде Delphi для работы с базами данных.

Вариант задачи выбирается по последней цифре учебного шифра.

Выполненный курсовой проект предъявляется на компакт-диске, содержащем тексты программ задач, результаты их выполнения, а также условия задач. Кроме того, прилагается титульный лист формата А4, на котором указаны наименование дисциплины, данные студента и его учебный шифр.

ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ

Выбрав соответствующий вариант

I Создать базу данных из таблиц, поля которых приведены в варианте, связав таблицы в отношении один ко многим по указанному полю, причём, 1-я таблица - родительская, 2-я - дочерняя.

II Создать 1-е приложение, содержащее на форме обе таблицы, и заполнить каждую не менее, чем 10-ю записями.

III Создать 2-е приложение с запросом, который показывает указанные в варианте поля из двух таблиц и поле, вычисленное на их основе.

Вариант 0.

I-ая таблица содержит поля :

(№ рейса, пункт назначения, стоимость билета, число всех мест в автобусе).

II-ая таблица содержит поля :

(порядковый № записи, № рейса, число занятых мест, дата рейса).

Связать таблицы по полю № рейса.

В запросе показать поля :

(дата рейса, № рейса, число всех мест в автобусе, стоимость билета, число занятых мест, степень заполнения автобуса в %, вырученная за рейс сумма).

Вариант 1.

I-ая таблица содержит поля:

(название кинотеатра, адрес, стоимость билета, число всех посадочных мест).

II-ая таблица содержит поля:

(порядковый № записи, время начала сеанса, название кинотеатра, число купленных билетов, дата сеанса).

Связать таблицы по полю название кинотеатра.

В запросе показать поля:

(дата сеанса, время начала сеанса, название кинотеатра, число всех мест, стоимость билета, число купленных билетов, степень заполнения зала в %, выручка за сеанс).

Вариант 2.

I-ая таблица содержит поля:

(ФИО работника, месячный оклад, должность, дата рождения).

II-ая таблица содержит поля:

(порядковый № записи, ФИО работника, месяц, дополнительный приработок за месяц).
Связать таблицы по полю ФИО работника.

В запросе показать поля:

(ФИО работника, месяц, месячный оклад, дополнительный приработок за месяц, суммарный доход работника за месяц).

Вариант 3.

I-ая таблица содержит поля:

(название предприятия, бюджетная месячная сумма, адрес предприятия, телефон).

II-ая таблица содержит поля:

(порядковый № записи, название предприятия, месяц, собственный доход предприятия за месяц).

Связать таблицы по полю название предприятия.

В запросе показать поля:

(название предприятия, месяц, бюджетная месячная сумма, собственный доход предприятия за месяц, полные денежные средства предприятия за месяц).

Вариант 4.

I-ая таблица содержит поля:

(№ пункта приёма металлолома, адрес, телефон, цена 1 кг).

II-ая таблица содержит поля:

(порядковый № записи, ФИО сборщика, № пункта приёма, вес (в кг) сданного лома, дата сдачи).

Связать таблицы по полю № пункта приёма.

В запросе показать поля:

(ФИО сборщика, дата сдачи, № пункта приёма, вес (в кг) сданного лома, цена 1 кг, сумма, полученная сборщиком).

Вариант 5.

I-ая таблица содержит поля:

(название пушного зверя, цена 1 кг пушнины).

II-ая таблица содержит поля:

(порядковый № записи, ФИО охотника, адрес, название пушного зверя, вес пушнины (в кг)).

Связать таблицы по полю название пушного зверя.

В запросе показать поля:

(ФИО охотника, адрес, название пушного зверя, вес пушнины, цена 1 кг пушнины, выручка охотника).

Вариант 6.

I-ая таблица содержит поля:

(Название факультета, проходной балл на факультете).

II-ая таблица содержит поля:

(порядковый № записи, ФИО абитуриента, название факультета, оценка по математике, оценка по физике, оценка по сочинению).

Связать таблицы по полю название факультета.

В запросе показать поля:

(ФИО абитуриента, название факультета, оценка по математике, оценка по физике, оценка по сочинению, проходной балл на факультете, избыточный балл).

Вариант 7.

I-ая таблица содержит поля:

(№ комнаты в гостинице, категория, стоимость одного дня проживания).

II-ая таблица содержит поля:

(порядковый № записи, ФИО выбывшего из гостиницы, домашний адрес, номер комнаты проживания, дата убытия, число дней проживания).

Связать таблицы по полю номер комнаты.

В запросе показать поля:

(ФИО выбывшего, номер комнаты, категория, стоимость одного дня проживания, число дней проживания, сумма, уплаченная за проживание).

Вариант 8.

I-ая таблица содержит поля:

(марка автомобиля, вес, грузоподъёмность).

II-ая таблица содержит поля:

(порядковый № записи, ФИО водителя, номер рейса, дата выезда, дата прибытия, марка автомобиля, вес груза).

Связать таблицы по полю марка автомобиля.

В запросе показать поля:

(ФИО водителя, номер рейса, марка автомобиля, вес автомобиля, вес груза, грузоподъёмность, запас грузоподъёмности).

Вариант 9.

I-ая таблица содержит поля:

(марка самолёта, ёмкость бензобака (в л.)).

II-ая таблица содержит поля:

(порядковый № записи, номер рейса, длительность полёта, начальный запас бензина (в л.), марка самолёта).

Связать таблицы по полю марка самолёта.

В запросе показать поля:

(номер рейса, длительность полёта, начальный запас бензина, марка самолёта, ёмкость бензобака, запас ёмкости).

МЕТОДИЧЕСКИЕ УКАЗАНИЯ К РЕШЕНИЮ ЗАДАЧИ

Рассмотрим, например, в Delphi приложение, обслуживающее учёт книг кафедральной библиотеки.

Сначала для ведения необходимого учета создается база данных типа Paradox с двумя таблицами Knigi.db (Книги) и Obsluj.db. Таблицы эти создаются с помощью программы Database Desktop. Они имеют следующую структуру :

Структура таблицы Knigi.db

№ поля	Наименование поля	Тип поля	Длина поля	Ключ	Смысл поля
1	IDN	+		*	Идентиф. № книги
2	Price	I			Стоимость
3	Nazvan	A	20		Название
4	Author	A	30		Автор
5	GIZDAN	I			Год издания
6	BuiDate	D			Дата покупки

Структура таблицы Obsluj.db

№ поля	Название поля	Тип поля	Длина поля	Ключ	Смысл поля
1	Por_N	+		*	Пор. номер записи
2	IDN	I			Идентиф. № книги
3	FIO_Chit	A	30		ФИО читателя
4	Data_Poluch	D			Дата получения
5	Data_Vozvr	D			Дата возврата
6	Кафедра	A	20		Название кафедры
7	Predmety	A	50		Ведомые предметы
8	Adr_Tel	A	60		Домашний адрес и телефон

Для совместной работы с этими таблицами было создано приложение KnigiObsluj, позволяющее фиксировать и отслеживать процесс обслуживания читателей библиотеки. Оно позволяет добавлять в таблицу Knigi.db новые книги или же удалять из неё списанные книги, а также записывать во вторую связанную таблицу факты пользования читателями фондами библиотеки. Экран работающего приложения KnigiObsluj содержит окно с двумя таблицами. Приведённые выше таблицы создаются посредством программы Database Desktop. Затем поместим в папку D:\Biblioteka таблицы Knigi.db и Obsluj.db.

Теперь откроем таблицу Knigi.db командой главного меню Database Desktop File/Open/Table, после чего командой Table/Restructure откроем для этой таблицы окно Restructure Paradox7 Table: Knigi.DB и в строке со списком Table Properties этого окна выберем из списка значение Validity Checks и теперь поочередно делаем щелчок на каждом не ключевом поле структуры таблицы и включаем после этого справа в окне флажок Required Field (требование ненулевого заполнения поля), после чего щёлкнем кнопку Save внизу окна. Далее, сделаем то же самое с другой таблицей.

Затем откроем таблицу Obsluj.db и командой Table/Restructure откроем для этой таблицы окно Restructure Paradox7 Table: Obsluj.DB, после чего в строке со списком Table Properties этого окна выберем из списка значение Secondary Indexes. Мы хотим создать для

этой таблицы вторичный индекс по полям Por_N, IDN, для этого щёлкнем кнопку Define в нашем окне и откроется диалоговое окно, в котором в левом окне Fields список всех полей, а правое окно Indexes Fields будет пока пустым. Выделяем по очереди в левом окне поля Por_N, IDN и копируем их в правое окно, нажимая находящуюся между окнами кнопку с изображением стрелки, идущей вправо (последовательность добавления имён существенна, ибо она определяет порядок чередования полей в списке). После этого щёлкнем кнопку ОК внизу диалогового окна, появится запрос об имени созданного индекса, мы введём, например, имя Sort_IDN и нажмём ОК.

Теперь в строке со списком Table Properties окна Restructure Paradox7 Table: Seans.DB выберем из списка значение Referential Integrity и щёлкнем ниже кнопку Define, откроется окно Referential Integrity. Мы хотим создать ссылочную целостность между таблицами Knigi.db и Obsluj.db по полю IDN. В левом списке Fields выберем поле IDN и нажмём кнопку с изображением стрелки вправо. Название IDN будет записано в поле Child Fields (поле внешнего ключа дочерней таблицы). В правом списке Tables выберем таблицу Knigi.db и нажмём кнопку с изображением стрелки влево. Теперь в поле Parent's Key (ключ родительской таблицы) появится ключ IDN. Переключатели Update rule в левой нижней части окна Referential Integrity определяют вид каскадных воздействий на дочернюю таблицу Seans.db при изменении значения поля связи в родительской таблице Computers.db:

- Cascade – каскадные изменения и удаления подчинённых записей в дочерней таблице Seans.db;
- Prohibit – запрет на изменение поля связи или удаление записи в родительской таблице, если для данной записи есть связанные записи в дочерней таблице Seans.db;

Выберем Prohibit и щёлкнем ОК. Система запросит указать имя созданной ссылочной целостности и мы введём имя например Ref_IDN и щёлкнем ОК. После этого щёлкнем кнопку Save в окне Restructure Paradox7 Table: Obsluj.DB и оно закроется. Структура базы создана. В ней у нас дочерняя таблица Obsluj.db имеет родительскую Knigi.db и связана с ней созданной ссылочной целостностью по полю IDN. Выйдем из Database Desktop командой File/Exit. Заполнять таблицы записями будем уже создав специальное приложение KnigiObsluj. И будем помнить, что изменять или удалять запись в родительской таблице можно лишь удалив прежде все подчинённые к ней записи из дочерней таблицы, поскольку мы выбрали в созданной ссылочной целостности тип каскадных взаимодействий Prohibit.

Создадим приложение KnigiObsluj. На форму Form1 поместим из вкладки Data Access палитры компонентов два компонента типа TDataSource, система назовёт их соответственно DataSource1 и DataSource2, далее, из вкладки BDE палитры компонентов надо будет поместить на экран Form1 два компонента типа TTable, система их назовёт соответственно Table1 и Table2. Далее, в верхнюю и нижнюю части экрана из вкладки Data Controls палитры компонентов поместим два компонента типа TDBGrid (сетка для отображения таблицы), система назовёт их соответственно DBGrid1 и DBGrid2 и с помощью Object Inspector настроим их параметры Align соответственно на значение alTop (сетка прижата к верхнему краю экрана) и alBottom (сетка прижата к нижнему краю экрана). Далее, взаимно настроим тройку компонентов (DataSource1, Table1, DBGrid1) с помощью Object Inspector, а именно, параметру DataSet компонента DataSource1 присвоим значение Table1 (выберем из всплывающего списка), параметру DataSource компонента DBGrid1 присвоим значение DataSource1 (выберем из всплывающего списка), параметру DatabaseName компонента Table1 присвоим значение

псевдонима нашей базы Bibliotek(выберем из всплывающего списка), затем для параметра TableName компонента Table1 выберем из всплывающего списка имя таблицы Knigi.DB и после этого обязательно для параметра Active этого же компонента выберем из списка значение True. Теперь таблица (пустая пока) появится в верхней сетке экрана. Совершенно аналогично взаимно настроим другую тройку компонентов (DataSource2, Table2, DBGrid2), здесь уже для параметра TableName компонента Table2 выберем из всплывающего списка имя таблицы Obsluj.DB. Обе пустые пока таблицы должны быть видны в верхней и нижней сетках. Теперь командой File/Save As... сохраним модуль KnigiObsl.pas в папку D:\Biblioteka, где в данный момент находится и наша база, а затем командой File/Save Project As...сохраним наш проект в ту же самую папку как KnigiObsluj.dpr . Закроем программу Delphi, затем снова её вызовем и загрузим командой File/Open или File/Reopen только что сохранённое приложение в виде проекта KnigiObsluj.dpr. Мы можем теперь для обоих компонентов Table1, Table2 изменить значение параметра DatabaseName с псевдонима на пустую строку (удалив из строки величину 'Bibliotek'). Это означает, что базой является текущая директория. (Не забыть после изменений снова восстановить для каждого компонента типа TTable значение параметра Active = True, иначе таблица на экране не появится). Сохраним приложение и выйдем из Delphi.

Осталось реализовать связь Master Detail между наборами данных. Выделяем щелчком компонент Table2 и в его Object Inspector для свойства Master Source выбираем из списка этой строки значение DataSource1. Теперь щёлкнем в Object Inspector справа в строке MasterFields, появится кнопка с тремя точками, щёлкнем на ней, откроется окно Дизайнер связи полей, в нём надо будет установить параметры связи.

Теперь сохраним созданное приложение командой File/Save и затем командой Run запустим на компиляцию и выполнение и создадим сначала ряд записей в основной таблице , а затем - ряд подчинённых к ним записей в подчинённой таблице .

ЛИТЕРАТУРА

Кандзюба С.П., Громов В.Н. "DELPHI. Базы данных и приложения. Лекции и упражнения." - Москва, Санкт-Петербург, Киев: DiaSoft, 2010 г.