МИНИСТЕРСТВО ОБЩЕГО И ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ РФ

Московский Государственный Институт Электроники и Математики

(Технический университет)

КУРСОВАЯ РАБОТА

по курсу «Информационные технологии в экономике»

на тему:

«Учет успеваемости студентов»

студента группы ЖК-21

Иванова Петра Сидоровича

Москва, 2010 г.

ОГЛАВЛЕНИЕ

Введение

Задачей любого учебного заведения является повышение эффективности учебно-воспитательного процесса, который влечет за собой рост успеваемости учеников. Имея возможность оценить успеваемость учеников, можно говорить о сбалансированном учебном процессе в школе, помогающем ученикам развить физические и умственные способности, нравственные качества и художественные вкусы, необходимые будущему гражданину.

Каждый период обучения ученика характеризуется своими особенностями, связанными с процессами роста и развития. Кроме объективных причин (таких как наличие профессионального педагогического состава, грамотно разработанных методик образовательных программ для учеников различного возраста, методика оценки знаний и т.п.), существуют и субъективные причины: недопустимость повышения уровня учебной нагрузки, четкий режим учебной работы, смена умственных и физических занятий, приобретение трудовых навыков, эстетическое и нравственное воспитание, развитие потребностей в самообразовании, особенно в интересующих учеников областях знаний, индивидуальный подход к каждому ученику с учетом его интересов и наклонностей и т.д.

Все эти мероприятия ведут к тому, что даже менее талантливые, относительно своих сверстников, достигают больших успехов. Одной из главных задач является возможность учитывать в любое время успеваемость ученика. Оценка успеваемости будет свидетельствовать о правильно или неправильно отлаженном учебно-воспитательном процессе. Рост успеваемости будет отражать успехи всего педагогического коллектива, а снижение – об ошибках, допущенных в работе, и необходимости в проведении анализа деятельности учебного заведения.

Оценка успеваемости с возможностью проведения сложного анализа – достаточно трудоемкая задача. Применение современных информационных технологий позволяет снизить трудоемкость и вывести учет успеваемости на новый уровень.

Необходимо разработать информационную систему. При этом, нужна качественная информационная система, построенная с использованием современных методов и инструментов.

ИС должна управлять вводом, обработкой, хранением и предоставлением информации об успеваемости студентов.

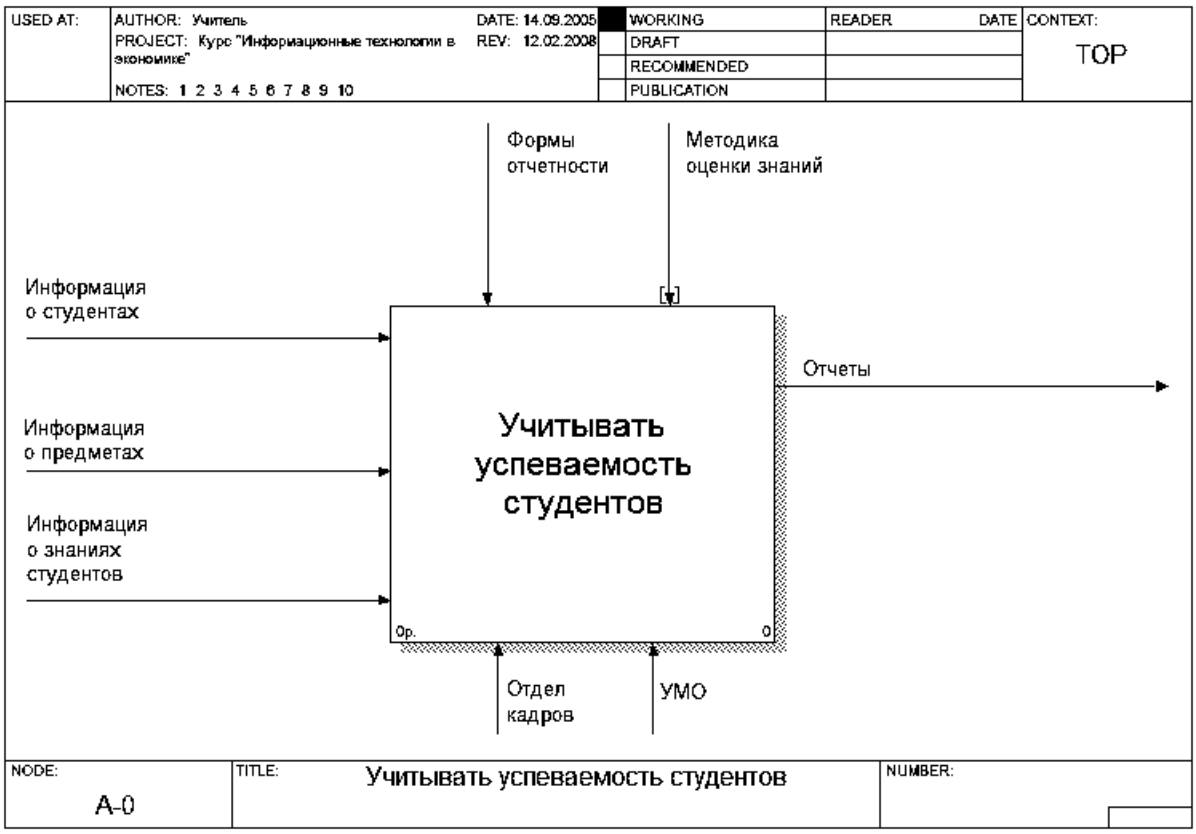
Основные отчеты на выходе ИС: Карточка успеваемости выбранного ученика; Ведомость успеваемости по выбранному предмету.

В работе будем применять современные методы. Информационная модель и модель данных будут созданы с использованием специальных программных пакетов. ИС будет построена на базе промышленной СУБД поддерживающей языки программирования четвертого поколения. Все это не только повысит качество конечного продукта, но и позволит повысить эффективности разработки, внедрения, эксплуатации и последующей модернизации системы.

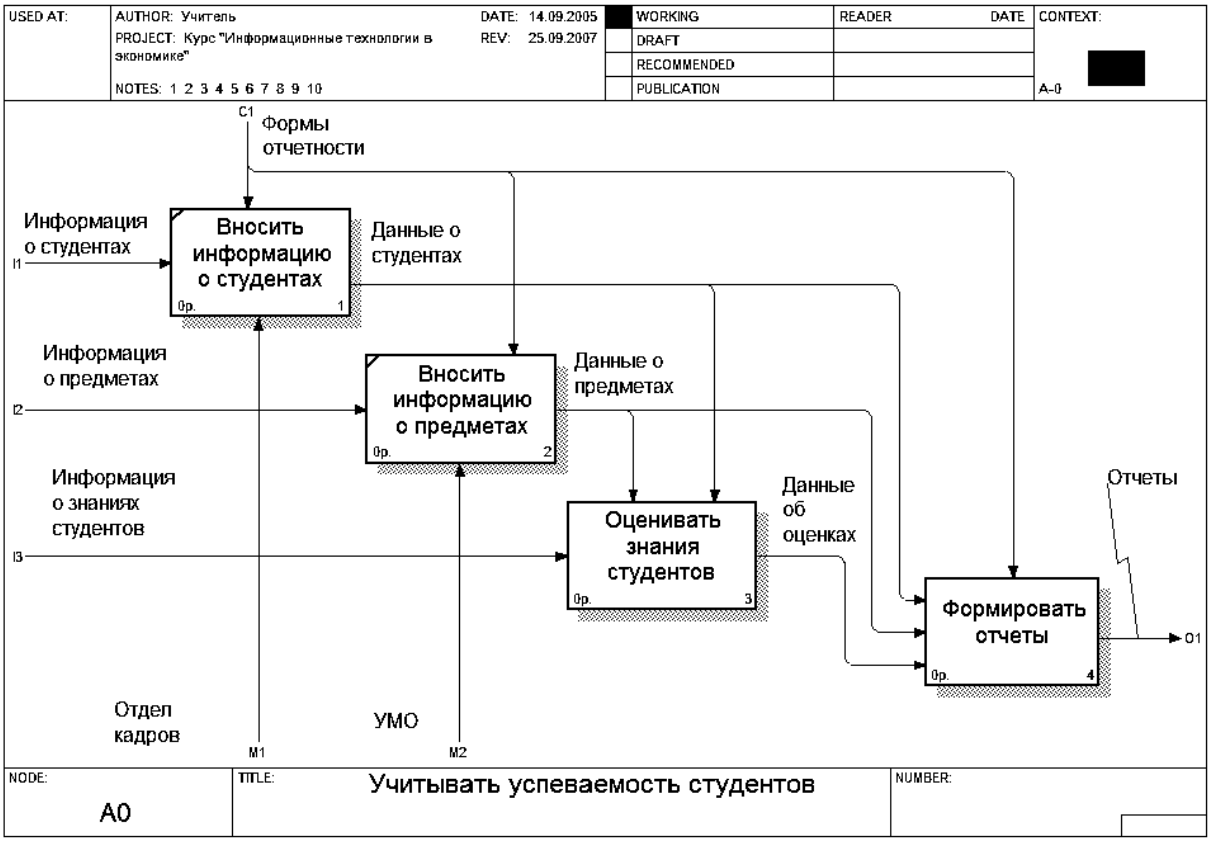
Функциональная структура ИС. Модель IDEF0.

Основная функция проектируемой информационной системы – учитывать успеваемость студентов.

При работе системы задействованы ресурсы отдела кадров и учебно-методического отдела. С использованием системы производится обработка первичной информации о студентах, предметах и знаниях студентов. При этом, функционирование системы подчинено правилам, описанным в методике оценки знаний, и направлено на получение отчетности в соответствии с заданными формами.



Более детально, система обеспечивает выполнение следующих функций: Вносить информацию о студентах; Вносить информацию о предметах; Оценивать знания студентов; Формировать отчеты.



Отдел кадров в соответствии с формами отчетности вносит первичную информацию о студентах и получает в результате формализованные данные о студентах.

Учебно-методический отдел в соответствии с формами отчетности вносит информацию о предметах и получает в результате данные о предметах.

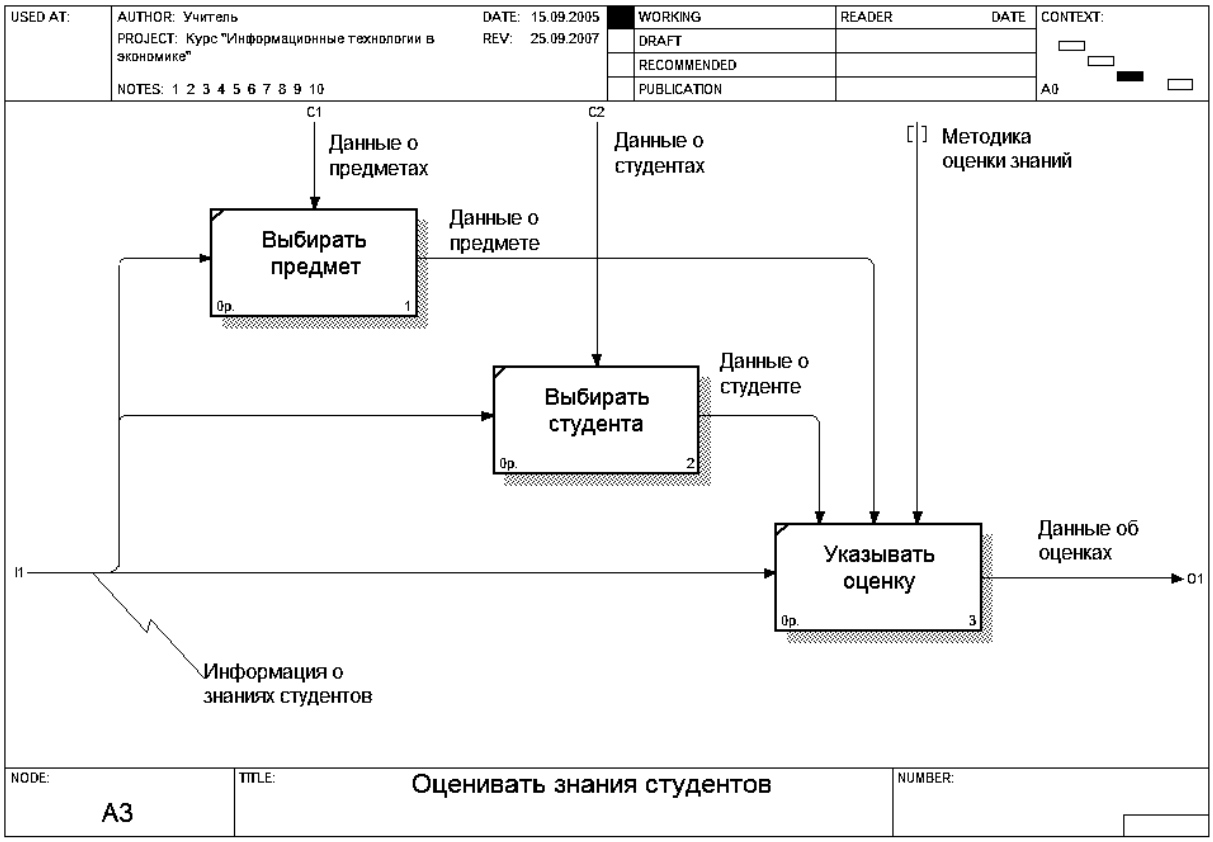
Функции внесения информации о студентах и внесения информации предметах не зависят друг от друга и могут выполняться параллельно.

Полученные данные о студентах и предметах управляют последующей оценкой знаний студентов, при которой информация о знаниях студентов порождает данные об оценках.

Далее данные о студентах, предметах и оценках обрабатываются при формировании отчетов. При этом формирование отчетов подчинено заданным формам отчетности.

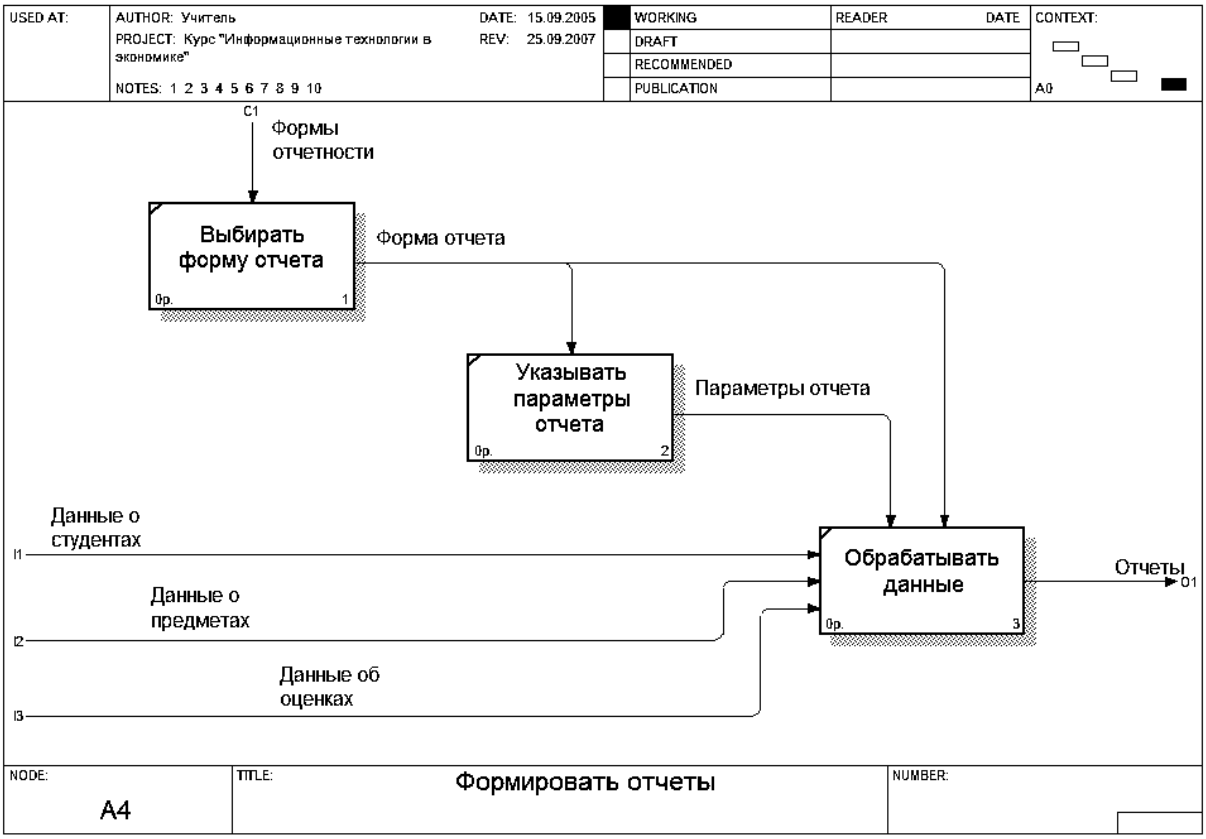
Оценка знаний студентов производится в три этапа. На каждом этапе, на входе имеем информацию о знаниях студентов. Сначала, основываясь на данных о предметах, необходимо выбрать предмет и получить в результате данные о конкретном предмете. Далее имея в виду данных о студентах выбрать студента – получить данные о студенте.

После с учетом данных о студенте, данных о предмете и методики оценки знаний указывается оценка. В результате получаем данные об оценках.



Формирование отчетов производится по стандартной схеме. Прежде всего, выбирается форма отчетности. При этом выбор осуществляется в рамках заданных форм отчетности. Выбранная форма позволяет указать параметры отчета.

Для того, чтобы получить отчеты на основании выбранной формы отчета и указанных параметров, обрабатываются данные о студентах, предметах и оценках.


USED AT:
AUTHOR: Учитель
DATE: 15.09.2005
PROJECT: Курс "Информационные технологии в экономике"
REV: 25.09.2007
NOTES: 1 2 3 4 5 6 7 8 9 10
WORKING
DRAFT
RECOMMENDED
PUBLICATION
READER
DATE
CONTEXT:
A0
C1
Формы отчетности
Выбирать форму отчета
0p.
1
Форма отчета
Указывать параметры отчета
0p.
2
Параметры отчета
Данные о студентах
I1
Данные о предметах
I2
Данные об оценках
I3
Обрабатывать данные
0p.
3
Отчеты
O1
NODE:
A4
TITLE:
Формировать отчеты
NUMBER:

Логическая и физическая модели БД. Модель IDEF1x.

Логическая модель БД

В рамках решаемой задачи модель данных состоит из трех сущностей: Студент; Предмет; Оценка.

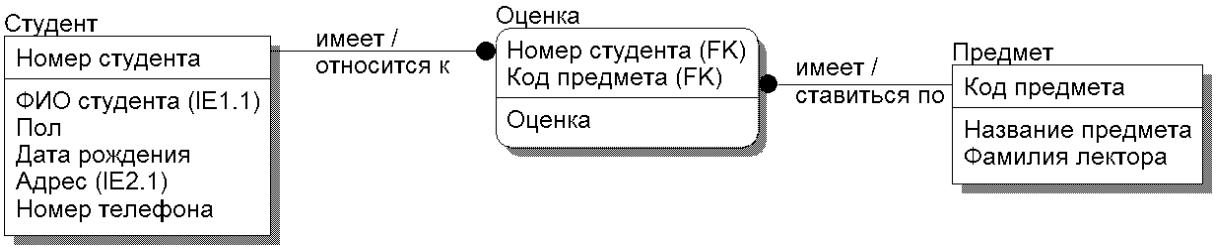
К сущности студент относятся следующие атрибуты: Ф.И.О. студента; Пол; Дата рождения; Адрес; Номер телефона; Номер студента. При этом, номер студента – специально введенный атрибут для идентификации сущности, поскольку прочие атрибуты (как отдельно, так и в совокупности) не могут однозначно идентифицировать сущность. Адрес и Ф.И.О. студента выступают только в качестве альтернативных ключей.

Атрибуты сущности предмет: Название предмета; Фамилия лектора; Код предмета. Код предмета однозначно идентифицирует сущность.

Сущности студент и предмет являются независимыми, в отличие от сущности оценка, которая зависит от студента и предмета. Действительно, оценка всегда ставится для некоторого студента по определенному предмету. Другими словами студент может иметь одну или несколько оценок (по предметам). Предмет может иметь одну или несколько оценок (для студентов). Оценка всегда относится к определенному студенту и ставится по конкретному предмету.

Атрибуты сущности оценка следующие: Оценка; Номер студента; Код предмета. Пара атрибутов номер студента и код предмета однозначно идентифицирует сущность оценка. Номер студента и код предмета – внешние ключи. Для каждого студента по определенному предмету может быть указана только одна оценка.

Графически логическая модель БД выглядит следующим образом:

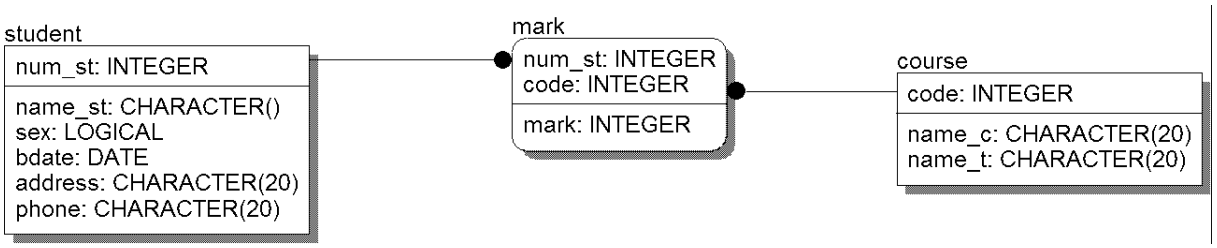


Физическая модель БД

На основании логической модели БД без особых усилий строится физическая модель БД.

Физическая реализация, как правило, требует использовать в названиях таблиц и полей только латинских букв, цифр и некоторых символов.

Соответствующая физическая модель графически выглядит следующим образом.



База данных состоит из трех таблиц: student; mark; course. Таблица mark зависит от таблиц student и course.

Поля таблицы student:

num_st (номер студента) – числового типа;

name_st (Ф.И.О. студента) – строка;

sex (пол) – логического типа (истина –мужской, ложь – женский);

bdate (дата рождения) – тип дата;

address (адрес) – строка;

phone (телефон) – строка.

Первичный ключ – num_st.

Поля таблицы course:

code (код предмета) – число;

name_c (название предмета) – строка;

name_t (фамилия лектора) – строка.

Первичный ключ – code.

Поля таблицы mark:

mark (оценка) – число;

num_st;

code.

Первичный ключ – num_st, code. Поля num_st и code – мигрировали из таблиц student и course.

Правила поддержки целостности

В процессе функционирования, в базе данных не должно возникать оценок которые относятся к несуществующему студенту или выставлены по несуществующему предмету.

Таким образом, чтобы данные были непротиворечивыми, на уровне базы данных нужно заложить выполнение следующих правил.

После каждого изменения оценки проверяется наличие студента с указанным номером и наличие предмета с указанным кодом. Если отсутствует студент или предмет, то выдается соответствующее сообщение об ошибке и действие отменяется

Изменение номера студента отменяется, если существуют оценки, ссылающиеся на старое значение номера студента.

Изменение кода предмета отменяется, если существуют оценки, ссылающиеся на старое значение кода предмета.

Запрещено удаление студента, если есть относящиеся к нему оценки.

Запрещено удаление предмета, если существуют выставленные по нему оценки.

Триггеры БД

При удалении записи из таблицы Студенты, если можно найти запись таблицы Оценки с таким номером студента, то возвращаем ошибку "Студента удаляешь напрасно ты. Связанные с ним Оценки существуют."

При удалении записи из таблицы Предметы, если можно найти запись таблицы Оценки с таким кодом предмета, то возвращаем ошибку "Предмет удаляешь напрасно ты. Связанные с ним Оценки существуют."

При изменении записи таблицы Студенты, если изменился номер студента и можно найти запись таблицы Оценки по старому значению номера студента, то возвращаем ошибку "Не возможно изменить номер студента, т.к. присутствуют оценки с таким номером."

При изменении записи таблицы Предметы, если изменился код предмета и можно найти запись таблицы Оценки по старому значению кода предмета, то возвращаем ошибку "Не возможно изменить код предмета, т.к. присутствуют оценки с таким кодом."

При изменении записи таблицы Оценки, если не удается найти запись таблицы Студенты с таким номером студента, то возвращаем ошибку "Введен неизвестный номер студента.", а если не удается найти запись таблицы Предметы с таким кодом предмета, то возвращаем ошибку "Введен неизвестный код предмета."

Отчеты

Выводы

Список литературы

Приложения

Тексты триггеров БД

```
/* ТРИГГЕР НА УДАЛЕНИЕ СТУДЕНТА */
TRIGGER PROCEDURE FOR DELETE OF student.
IF CAN-FIND(FIRST mark WHERE mark.num_st = student.num_st)
    THEN RETURN ERROR
        "Студента удаляешь напрасно ты. Связанные с ним Оценки
существуют.".

/* ТРИГГЕР НА УДАЛЕНИЕ ПРЕДМЕТА */
TRIGGER PROCEDURE FOR DELETE OF course.
IF CAN-FIND(FIRST mark WHERE mark.code = course.code)
    THEN RETURN ERROR
        "Предмет удаляешь напрасно ты. Связанные с ним Оценки существуют.".

/* ТРИГГЕР НА ИЗМЕНЕНИЕ СТУДЕНТА */
TRIGGER PROCEDURE FOR WRITE OF student
    NEW BUFFER new-student OLD BUFFER old-student.
IF old-student.num_st <> new-student.num_st AND
     CAN-FIND(FIRST mark WHERE mark.num_st = old-student.num_st)
    THEN RETURN ERROR
        "Не возможно изменить номер студента, т.к. присутствуют оценки с
таким номером.".

/* ТРИГГЕР НА ИЗМЕНЕНИЕ ПРЕДМЕТА */
TRIGGER PROCEDURE FOR WRITE OF course
    NEW BUFFER new-course OLD BUFFER old-course.
IF old-course.code <> new-course.code AND
    CAN-FIND(FIRST mark WHERE mark.code = old-course.code)
    THEN RETURN ERROR
        "Не возможно изменить код предмета, т.к. присутствуют оценки с
таким кодом.".

/* ТРИГГЕР НА ИЗМЕНЕНИЕ ОЦЕНКИ */
TRIGGER PROCEDURE FOR WRITE OF mark.
IF NOT CAN-FIND(FIRST course WHERE course.code = mark.code)
    THEN RETURN ERROR
        "Введен неизвестный код предмета.".
IF NOT CAN-FIND(FIRST student WHERE student.num_st = mark.num_st)
    THEN RETURN ERROR
        "Введен неизвестный номер студента.".
```

Отчет о структуре БД

```
Table Name     Description
--------------------------------------------------------------------------------
course
mark
student


================================================================================
============================== Table: course ===============================

              Table Flags: "f" = frozen, "s" = a SQL table
```

```
Table                          Dump     Table Field Index Table
Name                           Name     Flags Count Count Label
------------------------------ -------- ----- ----- ----- --------------------
course                         course             3     1 ?

 Storage Area: Schema Area

       Trigger Event Trigger Procedure    Overridable? Check CRC?
       ------------- -------------------- ------------ ----------
       DELETE        D_course.p           no           no
       WRITE         W_course.p           no           no


============================== FIELD SUMMARY ==============================
============================== Table: course ==============================

Flags: <c>ase sensitive, <i>ndex component, <m>andatory, <v>iew component

Order Field Name                Data Type   Flags Format          Initial
----- ------------------------- ----------- ----- --------------- ----------
   10 code                      inte        i     ->,>>>,>>9      0
   20 name_c                    char              X(20)
   30 name_t                    char              X(20)

Field Name                     Label                  Column Label
------------------------------ ---------------------- ----------------------
code                           Код предмета           Код предмета
name_c                         Название предмета      Название предмета
name_t                         Фамилия лектора        Фамилия лектора


============================== INDEX SUMMARY ==============================
============================== Table: course ==============================

Flags: <p>rimary, <u>nique, <w>ord, <a>bbreviated, <i>nactive, + asc, - desc

Flags Index Name                       Cnt Field Name
----- -------------------------------- --- --------------------------------
pu    XPKcourse                          1 + code

** Index Name: XPKcourse
 Storage Area: Schema Area


============================== FIELD DETAILS ==============================
============================== Table: course ==============================



================================================================================
============================== Table: mark ==============================

              Table Flags: "f" = frozen, "s" = a SQL table


Table                          Dump     Table Field Index Table
Name                           Name     Flags Count Count Label
------------------------------ -------- ----- ----- ----- --------------------
mark                           mark               3     3 ?

 Storage Area: Schema Area

       Trigger Event Trigger Procedure    Overridable? Check CRC?
       ------------- -------------------- ------------ ----------
       CREATE        C_mark.p             no           no
       WRITE         W_mark.p             no           no


============================== FIELD SUMMARY ==============================
============================== Table: mark ==============================

Flags: <c>ase sensitive, <i>ndex component, <m>andatory, <v>iew component

Order Field Name                Data Type   Flags Format          Initial
----- ------------------------- ----------- ----- --------------- ----------
   10 num_st                    inte        im    ->,>>>,>>9      0
   20 code                      inte        im    ->,>>>,>>9      0
```

```
   30 mark                      inte              ->,>>>,>>9      0

Field Name                     Label                  Column Label
------------------------------ ---------------------- ----------------------
num_st                         Номер студента         Номер студента
code                           Код предмета           Код предмета
mark                           Оценка                 Оценка


============================= INDEX SUMMARY ==============================
============================= Table: mark ================================

Flags: <p>rimary, <u>nique, <w>ord, <a>bbreviated, <i>nactive, + asc, - desc

Flags Index Name                       Cnt Field Name
----- -------------------------------- --- --------------------------------
      XIF1mark                           1 + num_st

      XIF2mark                           1 + code

pu    XPKmark                            2 + num_st
                                           + code

** Index Name: XIF1mark
 Storage Area: Schema Area
** Index Name: XIF2mark
 Storage Area: Schema Area
** Index Name: XPKmark
 Storage Area: Schema Area


============================= FIELD DETAILS ==============================
============================= Table: mark ================================



============================================================================
============================= Table: student =============================

              Table Flags: "f" = frozen, "s" = a SQL table


Table                          Dump     Table Field Index Table
Name                           Name     Flags Count Count Label
------------------------------ -------- ----- ----- ----- --------------------
student                        student            6     3 ?

 Storage Area: Schema Area

       Trigger Event Trigger Procedure    Overridable? Check CRC?
       ------------- -------------------- ------------ ----------
       DELETE        D_student.p          no           no
       WRITE         W_student.p          no           no


============================= FIELD SUMMARY ==============================
============================= Table: student =============================

Flags: <c>ase sensitive, <i>ndex component, <m>andatory, <v>iew component

Order Field Name                Data Type   Flags Format          Initial
----- ------------------------- ----------- ----- --------------- ----------
   10 num_st                    inte        i     ->,>>>,>>9      0
   20 name_st                   char        i     x(8)
   30 sex                       logi              yes/no          no
   40 bdate                     date              99/99/99        ?
   50 address                   char        i     X(20)
   60 phone                     char              X(20)

Field Name                     Label                  Column Label
------------------------------ ---------------------- ----------------------
num_st                         Номер студента         Номер студента
name_st                        ФИО студента           ФИО студента
sex                            Пол                    Пол
bdate                          Дата рождения          Дата рождения
address                        Адрес                  Адрес
phone                          Номер телефона         Номер телефона
```

```
============================== INDEX SUMMARY ==============================
============================== Table: student ==============================

Flags: <p>rimary, <u>nique, <w>ord, <a>bbreviated, <i>nactive, + asc, - desc

Flags Index Name                       Cnt Field Name
----- ------------------------------- --- ---------------------------------
w     adress                            1 + address

      name_st                           1 + name_st

pu    XPKstudent                        1 + num_st

** Index Name: adress
 Storage Area: Schema Area
** Index Name: name_st
 Storage Area: Schema Area
** Index Name: XPKstudent
 Storage Area: Schema Area


============================== FIELD DETAILS ==============================
============================== Table: student ==============================



============================== SEQUENCES ==================================
```

Примеры экранных форм