



Базы данных и информационные системы
1.4 Нормализация данных.
Ссылочная целостность.

Сумы
СумГУ, 2013



Ход работы

- ▶ Нормализация данных
 - ▶ Задание 1
 - ▶ Приведение к 3-й нормальной форме
 - ▶ Генерация скрипта создания таблиц
- ▶ Изменение структуры таблицы
 - ▶ Задание 2
- ▶ Ссылочная целостность данных
 - ▶ Задание 3
- ▶ Требования к отчету



Нормализация данных

На ER-диаграмме представлена сущность: рейсовый вылет самолета. Основные поля: номер самолета, дата и время вылета, авиакомпания, аэропорт прибытия, аэропорт отбытия, тип и марка самолета, количество посадочных мест, члены экипажа и их роли, аэропорт прибытия и отбытия.

1. Проанализуйте сущность, нормализуйте ее согласно 4НФ
2. Постройте новую ER-диаграмму
3. Создайте скрипт генерации структуры БД
4. Используя SQL*Plus создайте таблицы, отвечающие новым сущностям.

Рейсовый полет

Дата вылета (PK)

Время Вылета (PK)

Номер рейсового полета (PK)

Название Авиакомпания

Аэропорт вылета

Аэропорт прилета

Тип самолета

Количество мест

Член экипажа 1

Роль члена экипажа 1

Член экипажа 2

Роль члена экипажа 3

Член экипажа N

Роль члена экипажа N



Нормализация данных: 1НФ

- ▶ Отношение находится в первой нормальной форме (1НФ), если все атрибуты отношения являются простыми (требование атомарности атрибутов в реляционной модели), т.е. не имеют компонентов.
 - ▶ Проблема: В сущности «рейсовый вылет» много раз повторяется (Член экипажа К, Роль члена экипажа К).
 - ▶ Для приведения к 1НФ нужно выделить (Член экипажа К, Роль члена экипажа К) в отдельную сущность. «Член экипажа» и «Рейсовый вылет» связаны как многие-к-одному.
-



Нормализация данных: 2НФ

- ▶ Отношение находится во второй нормальной форме (2НФ), если оно находится в 1НФ, и все неключевые атрибуты отношения функционально полностью зависят от составного ключа отношения.
- ▶ Проблема: название аэропортов, авиакомпании не меняются от вылета к вылету. Потому их лучше выделить в новую сущность – «Рейс». «Рейс» и «рейсовый вылет» связаны как один-ко-многим.



Нормализация данных: 3НФ

- ▶ Отношение находится в третьей нормальной форме (3НФ), если оно находится во 2НФ, и все неключевые атрибуты отношения зависят только от первичного ключа.
- ▶ Можно предположить, что кроме имени члена экипажа ИС потребуются и другие поля: стаж, дата рождения и прочее. Поэтому сущность «Член экипажа» лучше разделить на две «Член экипажа» и «Сотрудник».
- ▶ Из сущности «Рейс» нужно выделить новые сущности «Аэропорт», «Авиакомпания», «Самолет» - которые не зависят от рейса.



Изменение структуры таблицы

- ▶ В случае ошибок проектирования ИС или внесения дополнительных функций часто приходится изменять структуру БД или отдельных таблиц.
 - ▶ С командами:
 - ▶ `CREATE TABLE ИмяТаблицы (...`
 - ▶ `DROP TABLE ИмяТаблицы`
- вы уже знакомы, пора вспомнить как можно изменить структуру существующей таблицы.



Изменение структуры таблицы

▶ Добавление колонок

- ▶ `ALTER TABLE Students ADD student_id number;`
- ▶ `ALTER TABLE Students ADD (student_id integer, nickname varchar(20));`

▶ Удаление колонок

- ▶ `ALTER TABLE Students DROP COLUMN student_id;`

▶ Переименование колонок

- ▶ `ALTER TABLE Students RENAME nicknmae TO login;`



Изменение типа атрибута

- ▶ Изменение типа атрибута
 - ▶ **ALTER TABLE** <ИМЯ ТАБЛИЦЫ> **MODIFY** <ИМЯ АТТРИБУТА> <ТИП ДАННЫХ> <размер/точность >;
 - ▶ **ALTER TABLE** Students **MODIFY** student_id integer;
- ▶ изменение типа данных возможно только в том случае, если столбец пуст;
- ▶ для незаполненного столбца можно изменять размер/точность. Для заполненного столбца размер/точность можно увеличить, но нельзя понизить;
- ▶ ограничение NOT NULL может быть установлено, если ни одно значение в столбце не содержит NULL. Опцию NOT NULL всегда можно отменить;
- ▶ разрешается изменять значения, установленные по умолчанию.



Задание

```
CREATE TABLE STUDENT(  
  SURNAME VARCHAR(60),  
  NAME VARCHAR (60),  
  STIPEND DOUBLE,  
  KURS INTEGER,  
  CITY VARCHAR(60),  
  BIRTHDAY DATE,  
  UNIV_ID INTEGER  
);
```

- ▶ Создайте таблицу
- ▶ Добавьте атрибут для хранения среднего бала
- ▶ Переименуйте колонку **SURNAME** в **FIO**
- ▶ Удалите колонку **NAME**
- ▶ Измените тип колонки **UNIV_ID** на **NUMBER**.



Поддержка целостности данных

- ▶ Простой первичный ключ можно задать непосредственно при определении атрибута.
- ▶ Составной первичный ключ можно задать как ограничение таблицы.
- ▶ Если таблица не имела первичного ключа, его можно добавить.

```
CREATE TABLE STUDENT(  
  STUDENT_ID INTEGER PRIMARY KEY,  
  SURNAME CHAR (25),  
  STIPEND INTEGER ,  
  CITY VARCHAR (15)  
  BIRTHDAY DATE,  
);
```

```
CREATE TABLE STUDENT(  
  SURNAME VARCHAR (25),  
  STIPEND INTEGER ,  
  CITY VARCHAR (15)  
  BIRTHDAY DATE,  
  CONSTRAINT PR_KEY PRIMARY KEY (SURNAME, BIRTHDAY)  
);
```

```
CREATE TABLE STUDENT(  
  SURNAME VARCHAR (25),  
  STIPEND INTEGER ,  
  CITY VARCHAR (15)  
  BIRTHDAY DATE);  
ALTER TABLE STUDENT ADD CONSTRAINT PR_KEY PRIMARY KEY  
  (SURNAME, BIRTHDAY)
```



Поддержка целостности данных

- ▶ Простой первичный ключ можно задать непосредственно при определении атрибута.
- ▶ Составной первичный ключ можно задать как ограничение таблицы.
- ▶ Если таблица не имела первичного ключа, его можно добавить.

```
CREATE TABLE STUDENT(  
  STUDENT_ID INTEGER REFERENCES PEOPLE(PEOPLE_ID),  
  SURNAME CHAR (25),  
  STIPEND INTEGER,  
  CITY CHAR (15)  
  BIRTHDAY DATE  
);
```

```
CREATE TABLE STUDENT(  
  SURNAME CHAR (25),  
  STIPEND INTEGER,  
  CITY CHAR (15)  
  BIRTHDAY DATE,  
  CONSTRAINT FR_KEY FOREIGN KEY (SURNAME, BIRTHDAY) REFERENCES  
  PEOPLE(SURNAME, BIRTHDAY)  
);
```

```
CREATE TABLE STUDENT(  
  SURNAME CHAR (25),  
  STIPEND INTEGER,  
  CITY CHAR (15) ,  
  BIRTHDAY DATE);  
ALTER TABLE STUDENT ADD CONSTRAINT FR_KEY FOREIGN KEY (SURNAME  
  BIRTHDAY) REFERENCES PEOPLE(SURNAME, BIRTHDAY)
```



Действия с ограничениями при модификации данных

СМ: http://en.wikipedia.org/wiki/Foreign_key

```
CREATE TABLE NEW_EXAM_MARKS(  
    STUDENT_ID INTEGER NOT NULL,  
    SUBJ_ID INTEGER NOT NULL,  
    MARK INTEGER,  
    DATA DATE,  
CONSTRAINT EXAM_PR_KEY PRIMARY KEY  
    (STUDENT_ID, SUBJ_ID),  
CONSTRAINT SUBJ_ID_FOR_KEY FOREIGN KEY  
    (SUBJ_ID)  
REFERENCES SUBJECT,  
CONSTRAINT STUDENT_ID_FOR_KEY FOREIGN KEY  
    (STUDENT_ID) REFERENCES STUDENT ON DELETE  
    SET NULL);
```



Задание

- ▶ Создайте таблицу с именем SUBJ_LECT (учебные дисциплины преподавателей), с полями LECTURER_ID (идентификатор преподавателя) и SUBJ_ID (идентификатор преподаваемой дисциплины).
- ▶ Первичным ключом (составным) таблицы является пара атрибутов LECTURER_ID и SUBJ_ID, кроме того, поле LECTURER_ID является внешним ключом, ссылающимся на таблицу LECTURER, а поле SUBJ_ID является внешним ключом, ссылающимся на таблицу SUBJECT
- ▶ Добавьте для всех ее внешних ключей режим обеспечения ссылочной целостности при котором при удалении строк в родительской таблице строки в дочерней таблице удаляются.



Требования к отчету

- ▶ Отчет должен содержать:
 - ▶ ER-диаграмму для [задания 1](#) , где отношение нормализовано согласно 3НФ
 - ▶ SQL-скрипт для создания таблиц для задания 1
 - ▶ SQL-скрипт создания и модификации таблиц для [задания 2](#)
 - ▶ SQL-скрипт создания и модификации таблиц для [задания 3](#)



Контрольные вопросы

- ▶ Что такое первичный ключ?
- ▶ Могут ли быть пустыми (NULL) атрибуты составного первичного ключа?
- ▶ Каким образом, используя SQL, можно создать внешний ключ?
- ▶ Как в таблицу можно добавить новый атрибут?
- ▶ При каких условиях можно изменить тип атрибута?

