~ Базы Данных ~

Автор: Пицик Харитон

Лекция 4 сентября 2025г.

О курсе

Курс называется “Базы_Данных_ФИИТ_ПИ_ИВТ”. Кодовые слова:

- “2025_311_1”;
- “2025_311_2”.

Правила курса: автоматом можно получить только 2 и 5. Для оценки 5:

- **Все** задания **в срок** (Темы 2-9, факультатив—дополнительный);

(последним заданием курса является разработка интерфейса);

- Самый сложный момент — построение реалиционных структур данных (примеры в курсе сознательно не дописаны, т.е. представлена

только некоторая идея, а далее — разработка самостоятельная);

- Допуск к экзамну: 7 заданий (на последних парах **НЕ** будут принимать много задач);
- На курсе есть книги, хороший источник — русский сайт `postgres pro`;
- Нет штрафа за количество попыток.

Книги

- Дейт “Введение в СУБД” 6-е издание;
- Пушников “СУБД”.

Очень общие понятия

Определение. База данных (БД) — набор постоянных данных, которые используются прикладными системами для какого-либо предприятия.

Определение. Система управления базами данных (СУБД) — программно-аппаратный комплекс - обеспечивает сохранность, целостность данных, доступ пользователей к данным.

Определение. Система баз данных — это, по сути, не что иное, как компьютеризированная система хранения записей. Саму же базу данных можно рассматривать как подобие электронной картотеки, т.е. хранилище для некоторого набора занесенных в компьютер файлов данных. (где файл — абстрактный набор данных) (Определение по *К.Дейту*).

Первые БД были созданы на основе файловых систем. Для каждой прикладной программы предоставлялся свой набор данных, оформленный в виде файла со своей структурой. Проблема: *ФС* не знает конкретной структуры файла: структура записи файла известна только программе, которая с ним работает.

Базовые понятия реляционных БД

Основные понятия — тип данных, домен, атрибут, кортеж, первичный ключ и отношение.

Определение. Понятие *тип данных* в реляционной модели данных полностью соответствует понятию типа данных в языках программирования. Обычно в современных РБД допускается хранение символьных числовых данных, битовых строк, специализированных числовых данных (например, деньги), а также специальных “темпоральных” данных (дата, время, временной интервал).

Определение. Понятие *домена* более специфично для баз данных, хотя и имеет некоторые аналогии с подтипами в некоторых ЯП. Стоит понимать домен как допустимого потенциального множества значений данного типа. Например, в число значений домена “Имена” могут входить только те строки, которые могут изображать имя. Данные считаются *сравнимыми* только в том случае, если они относятся к одному домену.

Определение. Фундаментальным понятием реляционной модели данных является понятие *отношения. Атрибут отношения* есть пара вида

< Имя_атрибута : Имя_домена >

Имена атрибутов должны быть уникальны в пределах отношения. Часто имена атрибутов в отношениях совпадают с именами соответствующих доменов.

Отношение R, определенное на множестве доменов $D_1, ..., D_n$ (не обязательно различных), содержит 2 части: заголовок и тело. *Заголовок отношения* содержит фиксированное количество атрибутов отношения:

$(< A_1 : D_1 >, ..., < A_n : D_n >)$

Тело отношения содержит множество кортежей отношения. Каждый *кортеж отношения* представляет собой множество пар вида

$(< A_1 : Val_1 >, ..., < A_n : Val_n >)$ таких что значение Val_i атрибута A_i принадлежит домену D_i. Отношение обычно записывают как

$R(< A_1 : D_1 >, ..., < A_n : D_n >)$ или $R(A_1, ..., A_n)$ или просто R.

Определение. *Реляционной базой данных* называется набор отношений.

Определение. *Схемой РБД* называется набор заголовков отношений, входящих в БД.

Свойства отношений

1. *В отношении нет одинаковых кортежей.* Тело отношений есть множество кортежей и, как всякое множество, не может содержать неразличимые элементы. Таблицы в отличие от отношений могут содержать одинаковые строки;
2. *Кортежи не упорядочены (сверху вниз).* Порядок атрибутов в таблице не несёт никакой смысловой нагрузки;
3. *Атрибуты не упорядочены (слева направо).* Т.к. каждый атрибут имеет уникальное имя в пределах отношения, то порядок атрибутов не имеет значения;
4. *Все значения атрибутов атомарны.* В ячейки таблиц можно поместить что угодно — массивы, структуры, и даже другие таблицы.

Первая нормальная форма.

Труднее всего дать определение вещей, которые всем понятны. Именно такая ситуация с определением отношения в *Первой Нормальной Форме* (1НФ).

Объяснение. Говорят, что отношение R находится в 1НФ, если оно удовлетворяет определению 2 (как в презентации). Говорят, что отношение R находитя в 1НФ, если его атрибуты содержат только скалярные (*атомарные*) значения.

Целостность реляционных данных.

Существует два ограничения, которые должны выполняться любой РБД. Это:

- **Целостность сущностей**;
- **Целостность внешних ключей**.

Прежде чем говорить о целостности сущностей, опишем использование `null`-значений в РБД.

Парадокс 1. `null` значение не равно самому себе. Выражение `null=null` даёт значение НЕИЗВЕСТНО.

Парадокс 2. Также неверно, что `null` значене не равно самому себе. Выражение `null <> null` также принимает значение НЕИЗВЕСТНО.

Парадокс 3. `a or (not a)` не обязательно ИСТИНА. Значит, в трехзначной логике не работает принцип исключительного третьего (любое высказывание либо истинно, либо ложно).

Важно: если атрибут существеннен для построения БД, то он никогда не может принимать `null`.

Потенциальные ключи

Определение. Пусть дано отношение R. Подмножество атрибутов K отношения R будем называть *потенциальным ключом*, если K обладает следующими свойствами:

- В отношении R не может быть двух различных кортежей, с одинаковым значением K;
- Никакое подмножество в K не обладает свойством уникальности.

Потенциальный ключ, состоящий из одного атрибута, называется *простым*, а из нескольких атрибутов — *составным*. Традиционно, один из потенциальных ключей объявляется *первичным*, а остальные — *альтернативными*.

Замечание. Поняти потенциального ключа является *семантическим* понятием и отражает некоторый смысл (трактоку) понятий из конкретной предметной области.

Также сущесвует т.н. *фиктивный* ключ.

Лекция 11 сентября 2025г.

Целостность сущностей

Атрибуты, входящие в состав некоторого потенциального ключа не могут принимать `null`-значений.

Внешние ключи

Номер товара	Товар	Количество	Номер поставщика	Поставщик
1	товар_1	100	1	ООО Премьер видео
2	товар_2	200	3	ООО ДиВиДи Клуб
2	товар_2	150	1	ООО Премьер Видео

Потенциальный ключ — (Номер товара, Номер поставщика).

Проблемы:
1. Если изменилось наименование таблицы — необходимо внести изменение во все строки таблицы;
2. Если поставщик прекратил поставки — удаление информации о поставках приведет к удалению информации о поставщике.

Идея: разбить таблицу на 3:
- Таблица *Номер поставщика, Поставщик*
- Таблица *Номер товара, Товар*
- Таблица *Номер поставщика, Номер товара, Количество*

В этой таблице, в 3й таблице Номер поставщика и Номер товара являются **FOREIGN KEY**, в 1й таблице Номер поставщика, во 2й таблице Номер товара — **PRIMARY KEY**.

Определение. Подмножество атрибутов *FK* отношения *R* будем называть **внешим ключом**, если
1. Существует отношение *S* (*R* и *S* не обязательно различны) с потенциальным ключом *K*;
2. Каждое значение *FK* в отношении *R* всегда совпадает со значением *K* для некоторого кортежа из *S*, либо является `null`-значением.

Отношение *S* называется **родительским** отношением, а *R* — **дочерним**. Внешний ключ, как правило, не обладает свойством *уникальности*.

Хотя каждое значение внешнго ключа обязано совпадать со значениями потенциального ключа в некотором кортеже родительского отношения, то обратное, вообще говоря, неверно. Например, могут существовать поставщики, не поставляющие никаких деталей.

Для внешнего ключа не требуется, чтобы он был компонентом некоторого потенциального ключа.

Примеры:
- Таблицы *Автобус-Водители-Рейс* — отношение *многим-ко-многим* (у автобусов много водителей, у водителей много автобусов)
- Таблицы *Книги-Авторы* — отношение *многие-к-одному* (у книг 1 автор, у автора много книг)
- Таблицы ещё бывают *один-к-одному*.

Целостность внешних ключей

Т.к. внешне ключи фактически служат ссылками на кортежи в другом (или в том же самом) отношении, то эти ссылки не должны указывать на несуществующие объекты.

Это определяет следующее *правило целостности внешних ключей*:

- Внешние ключи не должны быть несогласованными, т.е. для каждого значения внешнего ключа должно существовать соответствующее значение

первичного ключа в родительском отношении.

Явная формулировка правил целостности помогает четко понять, какие опасности несёт в себе пренебрежение этими правилами.

ДОПИСАТЬ ПРО insert, update, delete (ДЛЯ РО И ДЛЯ ДО)

Родитель: insert+, delete+-, update+-

Потомок: insert+-, update+-, delete+

Лекция 18 сентября 2025

Целостность внешних ключей

Внешние ключи не должны быть несогласованными, т.е. для каждого значения внешнего ключа должно существовать соответствующее значение первичного ключа в родительском отношении.

Стратегии поддержания ссылочной целостности:

- `Restrict` — не разрешать выполнение операции, приводящей к нарушению ссылочной целостности. Это самая простая стратегия, требующая только проверки, имеются ли кортежи в дочернем отношении, связанные с некоторым кортежем в родительском отношении
- `Cascade` — разрешить выполнение требуемой операции, но внести при этом необходимые поправки в других отношениях так, чтобы не допустить нарушения ссылочной целостности и сохранить все имеющиеся связи

Дополнительно контроллировать ход подобных операций можно, например, триггеров.

- Триггеры – запускаются в момент, когда происходит определённое событие. Пример с `delete`:

```
create trigger on postavshik from delete
  update table postavshik status = false where id.p = 10
delete from postavshik where id.p = 10
```