**Лекция 5**

**ТЕМА: Проектирование БД. Нормализация отношений.**

**Проектирование баз данных.**

Создание БД необходимо начинать с ее проектирования. В результате проектирования должна быть определена структура реляционной БД, т. е. состав реляционных таблиц, их структура и логические связи. Структура таблицы определяется составом столбцов, их последовательностью, типом данных каждого столбца, их размером, а также ключом таблицы.

При этом структура должна быть эффективной и обеспечивать:

1). Быстрый доступ к данным.

2). Отсутствие дублирования (повторения) данных.

3). Целостность данных.

При проектировании структуры данных можно выделить два основных подхода:

1). Сбор информации об объектах решаемой задачи в рамках одной таблицы (одного отношения) и последующая декомпозиция её на несколько взаимосвязанных таблиц на основе нормализации отношений.

2). Формулирование знаний о системе (определение типов исходных данных и их взаимосвязей) и требований к обработке данных, а затем получение с помощью средств CASE схемы БД или прикладной информационной системы.

Проектирование структуры данных (структуры БД) также называют **логическим проектированием,** или проектированием на логическом уровне.

**Целостность БД.**

Под целостностью БД понимают, прежде всего, ее непротиворечивость. Она обеспечивается в частности тем, что каждый элемент данных хранится только в одном месте и его физическое изменение означает логическое изменение всех тех контекстов БД, где это значение логически присутствует. Говоря о целостности БД иногда различают 2 основных вида целостности БД.

1. **Сущностная целостности**. Она достигается за счет того, что каждый объект (сущность) однозначно идентифицируется или логически адресуется с помощью уникального первичного ключа.

Например, СУБД, заботясь о сохранении сущностной целостности БД, не должна позволить пользователю изменить на ВО1 значение первичного ключа во втором кортеже отношения «Отделения», т. к. при этом возникнет неоднозначность адресации и БД станет противоречивой.

2. **Ссылочная целостность**. Суть её в том, что любому значению внешнего ключа, должно соответствовать значение первичного ключа в том отношении, на которое ссылается данное отношение.

Пример: К нарушению ссылочной целостности привела бы операция удаления второго кортежа из отношения «Отделения», т. к. при этом соответствующие ссылки в отношении «Подразделения» показывали бы вникуда. СУБД не должна позволять пользователю выполнять такого рода операции.

**Нормализация БД.**

Информация в БД должна быть непротиворечивой, неизбыточной и целостной.

Нормализация БД представляет собой процесс уменьшения избыточности информации в БД и заключается в приведении хотя бы к третьей нормальной форме.

Процесс проектирования БД с использованием метода нормальных форм является с итерационным (пошаговым) и заключается в последовательном переводе по определенным правилам отношений из первой нормальной формы в нормальные формы более высокого порядка. Каждая следующая нормальная форма ограничивает определенный тип функциональных зависимостей, устраняет соответствующие аномалии и сохраняет свойства предшествующих нормальных форм.

**Первая нормальная форма.** Она требует, чтобы каждое поле таблицы БД было:

1) неделимым;

2) не содержало повторяющихся групп.

Неделимость поля означает, что значение поля не должно делиться на более мелкие значения. Например, если в поле «Подразделение» содержится название факультета и название кафедры, то требование неделимости не соблюдается и необходимо из данного поля выделить название кафедры в отдельное поле.

Повторяющимися считаются поля, содержащие одинаковые по смыслу значения.

Пример таблицы, имеющей повторяющиеся группы полей.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | Физика | История | Химия | Биология |
| Иванов | 3 | 4 | – | – |
| Петров | – | 5 | – | 5 |
| Сидоров | – | – | 3 | 3 |

Пример нормализованной таблицы (1НФ):

|  |  |  |
| --- | --- | --- |
| Студент | Предмет | Оценка |
| Иванов | Физика | 3 |
| Иванов | История | 4 |
| Петров | История | 5 |
| Петров | Биология | 5 |
| Сидоров | Химия | 3 |
| Сидоров | Биология | 3 |

Вторая и третья нормальные формы касаются отношения между ключевыми и неключевыми полями.

Приведение ко второй и третьей нормальным формам рассмотрим на примере.

Пусть у нас есть таблица, описывающая поставки деталей с полями:

1. Номер поставщика (№пост).

2. Номер детали (№дет).

3. Город поставщика (город).

4. Статус поставщика (статус).

5. Количество поставленных деталей (количество).

В первичный ключ данного отношения входят поля №пост и №дет.

Причем значения полей «город» и «статус» зависят только от части первичного ключа -поля №пост и при этом статус поставщика однозначно определяется городом.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| №пост | №дет | город | статус | количество |
| П1 | Д1 | Лондон | 20 | 10 |
| П1 | Д4 | Лондон | 20 | 35 |
| П2 | Д2 | Париж | 10 | 30 |
| П3 | Д2 | Лондон | 20 | 7 |
| П3 | Д3 | Лондон | 20 | 11 |

**Вторая нормальная форма.** Отношение находится во второй нормальной форме, если:

1. Оно находится в 1НФ.

2. Любой неключевой атрибут этого отношения зависит от всего первичного ключа.

Если ключ составной, то не существует подмножества ключевых атрибутов, от которого зависит к-л неключевой атрибут.

Из этого определения следует, что наше отношение не находится во 2НФ.

Подобная БД содержит избыточную информацию, т. к. информация о городе и статусе, являясь справочной, повторяется многократно. Также существует ряд неудобств:

1). Аномалия добавления: невозможно в нашу БД добавить информацию о поставщике (т. е. название города и статус), если он не поставил деталей.

2). Аномалия удаления: удаляя информацию о поставке, например поставщика П2, мы одновременно удаляем и информацию о поставщике.

3). Аномалия обновления: если, например, поставщик П1 переехал в Париж, то придется искать все записи с этим поставщиком и исправлять город на Париж. Если этого не сделать всюду, то БД станет противоречивой.

Для приведения нашей БД ко 2НФ, выделим поля «№пост», «город» и «статус» в отдельную таблицу:

Поставщики Поставки

№пост

№дет

город

№пост

Количество

статус

В отношении «Поставщики» первичным ключом будет атрибут «№пост». Теперь можно внести в БД информацию о поставщике, даже если у него нет поставок. Удаление информации о поставке не приведет к удалению информации о соответствующем поставщике. Изменение города поставщика требует изменения ровно одного кортежа.

Тем не менее, анализирую таблицу поставщиков, также можно заметить ее избыточность. В ней информация о статусе города повторяется многократно. Для устранения этой аномалии приведем нашу БД в ЗНФ.

**Третья нормальная форма.** Отношение находится в третьей нормальной форме, если:

1. Оно находится во второй нормальной форме.

2. Все его неключевые атрибуты напрямую зависят от ключа. (Т. е. не существует функциональных зависимостей между неключевыми атрибутами).

Отношение «Поставщики» не находится в ЗНФ, т. к. существует функциональная связь между неключевыми атрибутами «город» и «статус». Это приводит к ряду неудобств:

1. Аномалия добавления: в БД нельзя вставить информацию о городе, если нет поставщика из этого города.

2. Аномалия удаления: исключение информации о поставщике, например, П2, приводит к исключению информации о соответствующем городе (статус).

3. Аномалия обновления: изменение статуса города потребует просмотра нескольких кортежей. Выполнив декомпозицию отношения «Поставщики», выделим в отдельную таблицу поля «город» и «статус».

Поставщики Города Поставки

статус

Количество

№пост

№дет

город

№пост

город

Первичным ключом в отношении «Города» будет атрибут «город».

Теперь наша БД находится в ЗНФ. Это гарантирует:

1. Нет избыточности, а, следовательно, нет проблем с удалением, вставкой и обновлением.

2. Все выполненные декомпозиции обратимы, т. е. при необходимости можно восстановить, например, исходную таблицу, при этом никакая информация не будет потеряна.