Анализ БД на избыточность

Проанализировать информацию по избыточности баз данных и привести примеры неверного логического проектирования. Предложить алгоритм решения (по устранению) недостатков логической структуры.

В проектировании БД не редко встречается проблема по избыточности баз данных - когда в базе данных содержится ненужная информация, которую можно упростить. Зачастую данная проблема возникает из-за неверного логического проектирования. Ниже мы рассмотрим примеры неверного проектирования и способы устранения ошибок (аномалий).

1) Выбор первичного ключа (primary key, далее PK).

Одна из самых частых ошибок проектирования, приводящая к избыточности баз данных – это неоптимальный выбор PK.

PK должен обладать не только свойствами уникальности, но также и свойством минимальности (или неделимости, атомарности). Почему? Зачастую модель данных состоит из нескольких отношений, которые связаны между собой при помощи внешних ключей (foreign key, далее FK).

Представим, что у нас есть отношения A и B. FK отношения B идентичны PK отношения A, именно за счет этого обеспечивается связь отношений A и B. Если PK отношения A состоит из нескольких атрибутов (допустим, трех), то мы будем вынуждены добавить все эти атрибуты (в нашем случае, все 3) как FK в отношение B. Такой подход не оптимален, так как обработка и хранение данных в таком случае будет расходовать лишние ресурсы ЭВМ.

Пример, как делать не надо:

СОТРУДНИК

Таб номер (Правильный PK)	Фамилия (Наш PK)	Имя (Наш PK)	Отчество (Наш PK)	Супер Уникальный Номер (лишний атрибут)
K01	Иванов	Владимир	Кириллович	1
K02	Веллингтон	Петр	Кириллович	2
H01	Иванов	Владимир	Гаврилович	3

ЗАКАЗ

Номер	Получение	Создан	Фамилия Сотрудника (FK)	Имя Сотрудника (FK)	Отчество Сотрудника (FK)	Ид клиента
1	Самовывоз	12.09.16 16:12:30	Иванов	Владимир	Кириллович	K1
2	Самовывоз	12.09.16 22:42:18	Иванов	Владимир	Кириллович	K2

3	Доставка	13.09.16 10:01:23	Веллингтон	Петр	Кириллович	К2

Правильным решением было бы в качестве РК отношения СОТРУДНИК выбрать атрибут “Таб номер”, а затем внести атрибут “Номер сотрудника” в качестве FK отношения ЗАКАЗ, не добавляя в него атрибуты “Фамилия”, “Имя”, “Отчество”.

Другим примером неправильного выбора РК было бы создание нового уникального атрибута (например, “Супер Уникальный Номер”) и назначение его РК, если отношение уже содержит атрибут, отлично годящийся на роль РК (в нашем случае, Таб номер).

При выборе РК в отношении убедитесь, что он обладает свойствами уникальности **и минимальности.** Только в случае**,** если найти такой атрибут не удается, можно рассмотреть вариант добавление нового атрибута и назначения его в качестве РК.

2) Проведение декомпозиции.

Декомпозиция – один из самых распространённых способов решения различных проблем (аномалий) при моделировании БД. Декомпозиция – это разбиение одного отношения на несколько. Признаком необходимости декомпозиции является взаимосвязь между не ключевыми атрибутами отношения.

Например, представим, что в фирме, занимающейся тех обслуживанием, каждый отдел имеет свой номер телефона для приема звонков.

Тем не менее, мы имеем следующее отношение:

СОТРУДНИКИ_ОТДЕЛЫ

Номер Сотрудника (РК)	Фамилия Сотрудника	Номер Отдела	Телефон
1	Иванов	1	11-22-33
2	Петров	1	11-22-33
3	Сидоров	2	33-22-11

В данном отношении, неосновной атрибут “Телефон” имеет зависимость с атрибутом “Номер отдела”

Номер Отдела -> Телефон.

Проведем декомпозицию и получим 2 отношения:

Сотрудники:

Номер Сотрудника (РК)	Фамилия Сотрудника	Номер Отдела (FK)
1	Иванов	1
2	Петров	1
3	Сидоров	2

Отделы:

Номер Отдела (PK)	Телефон
1	11-22-33
2	33-22-11

За счет этого устраняется избыточность связанная с многократным хранением одинаковых данных (в нашем случае, одинаковых номеров телефонов)