Санкт-Петербургский государственный политехнический университет

Институт информационных технологий и управления

Кафедра компьютерных систем и программных технологий

Отчет по лабораторной работе

По дисциплине «Базы данных»

«Разработка структуры и нормализация БД»

Работу выполнил студент группы № 43501/4 Смирнов Л.А.

Работу принял преподаватель Мяснов А.А.

Санкт-Петербург

2015

1. Цель работы

   Познакомить студентов с основами проектирования схемы БД, способами нормализации отношений в БД.

2. Программа работы

   - Представить SQL-схему БД, соответствующую заданию (должно получиться не менее 7 таблиц)

   - Привести схему БД к 3НФ

   - Согласовать с преподавателем схему БД. Обосновать соответствие схемы 3НФ.

   - Продемонстрировать результаты преподавателю

3. Выполнение задания

   Выбранное задание: Военкомат. ИС содержит информацию о призывниках различного возраста и статуса, текущем местонахождении призывников и личных делах, медицинских данных.

   Для начала определим, из каких таблиц будет состоять наша база:

   Главной таблицей будет «Личное дело». В нем ожидаются поля номера личного дела призывника, номер призывника и дата постановки на учет. Primary key – Номер личного дела.

   От описанной таблицы выходит связь к призывнику. Внешний ключ – номер призывника. В данной таблице будут присутствовать такие поля как ФИО призывника, дата рождения, семейное положение, ID матери, ID отца, ID места обучения. Primary key – Номер призывника.

   Далее выходят из Призывника таблицы отца и матери призывника (2 таблицы). У них атрибуты похожие, а именно: ID матери\отца, ФИО, дата рождения, место прописки.

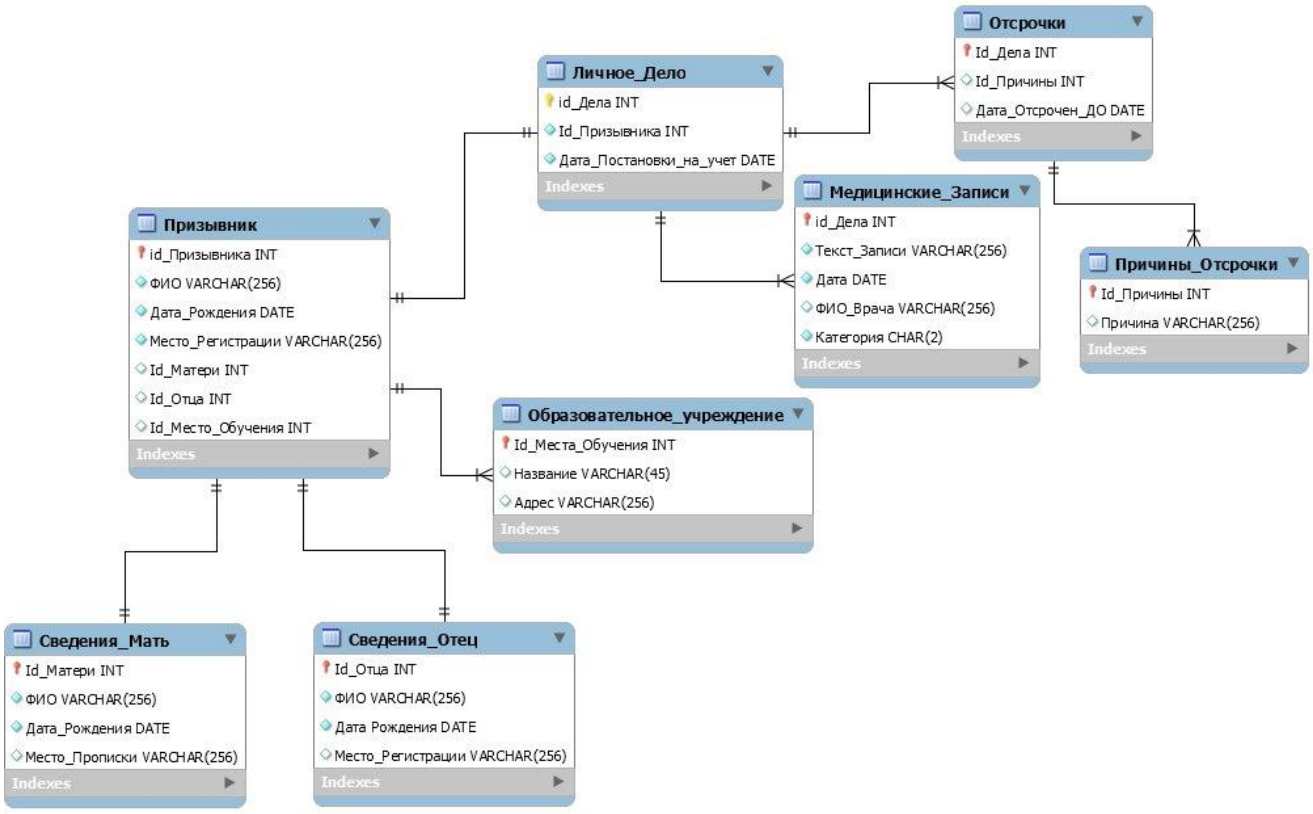
   У призывника есть место обучения будь то институт или колледж. В таблице «Образовательное учреждение» есть такие атрибуты как идентификатор места обучения, название и адрес.

Также в личном деле есть информация об отсрочках на данный момент. Ожидается в нем появление полей ID личного дела, ID причины отсрочки (так как в основном отсрочки похожи друг на друга) и дата отсрочки.

Соответственно, есть таблица с перечисленными причинами отсрочек. У данной таблицы должны быть как минимум идентификатор и строковые «причины».

В личном деле есть пометка о медицинских обследованиях. В таблицу будут входить такие атрибуты как идентификатор (первичный ключ), текст последней записи , дата отметки, ФИО врача.

Используя MySQL (MySQL Workbench), получим следующую схему:



Приведу доказательство того, что база данных находится в 3нормальной форме.

Начнем с **первой нормальной формы**, потому что если модель не находится в первой нормальной форме и второй, то она, соответственно не находится и в третьей нормальной форме.

Первая нормальная форма:

- запрещает повторяющиеся столбцы (содержащие одинаковую по смыслу информацию).
- запрещает множественные столбцы (содержащие значения типа списка и т.п.).
- требует определить первичный ключ для таблицы, то есть тот столбец или комбинацию столбцов, которые однозначно определяют каждую строку.

Например, в если в таблице «Личное дело» оставить все медицинские записи на протяжении службы (в одном поле), то сразу же пропадает атомарность атрибутов, то есть необходимо разделить связь, добавив новую таблицу «Медицинские Записи». Будем считать, что все атрибуты атомарны (хотя бы для определенной задачи), потому что если будет требование, например, разделить ФИО на имя, фамилию, отчество для удобства поиска по базе, то это будет принципиально важным моментом.

Вторая нормальная форма

Вторая нормальная форма требует, чтобы неключевые столбцы таблиц зависили от первичного ключа в целом, но не от его части. Вторая нормальная форма связана с избыточностью данных. Означает это то, что мы должны хранить в таблице только данные, которые напрямую связаны с первичным ключом и не имеют отношения к другой сущности.

Пример: если мы решим в таблице «Призывник» сделать составной ключ, состоящий из ФИО призывника и даты рождения. Также в данной таблице присутствуют ID матери, ID отца, ID места обучения. Если нам потребуется узнать имя отца призывника, то, чтобы добраться до его значения, нам нужно знать не только ФИО, но и избыточное значение даты рождения, которое никак не характеризует, например, дату рождения отца (при необходимости).

Третья нормальная форма

Чтобы таблица находилась в третьей нормальной форме, необходимо, чтобы неключевые столбцы в ней не зависели от других неключевых столбцов, а зависели только от первичного ключа. То есть если в таблице существуют расчетные столбцы, значения которых можно получить путем каких-либо манипуляций с другими столбцами таблицы. Для приведения таблицы в третью нормальную форму такие столбцы из таблиц надо удалить.

В данной базе данных транзитивных зависимостей не наблюдалось. Но для примера можно придумать одну из возможных. Например, если бы в личном деле было поле даты призыва на службу и дополнительное поле, которое соответствовало оставшимся дням службы, высчитывающееся из полного количество дней службы, то это бы означало не соответствие 3НФ. Функциональных зависимостей в таблицах присутствовать не должно.

4. Выводы

На этой работе мы познакомились с основами проектирования схем БД и приведением SQL-схемы БД к 3НФ.

Рассмотрим цели нормализации и ее достоинства:

1) **Исключение некоторых типов избыточности.** Избыточность данных приводит к непродуктивному расходованию свободного места на диске и затрудняет обслуживание баз данных, а точнее удаление данных из базы данных, с модификацией данных в таблице базы данных и аномалия добавления данных в базу данных. Например, если данные, хранящиеся в нескольких местах, потребуется изменить, в них придется внести одни и те же изменения во всех этих местах. Например, представим, что у нас есть таблица с преподавателями и предметами, которые преподаватели ведут. Избыточность в этой таблице заключается в том, что любой преподаватель может вести несколько предметов и для каждого нового предмета приходится добавлять новые записи в таблицу.
2) **Упрощение процедуры применения необходимых ограничений целостности.** Отношения, определенные с помощью первичных и внешних ключей позволяют организовать СУБД автоматический контроль согласованности данных, в том числе позволяют реализовать каскадное обновление связанных по внешнему ключу полей в соответствующих таблицах.
3) **Устранение некоторых аномалий обновления.** Например, если данные, хранящиеся в нескольких местах, потребуется изменить, в них придется внести одни и те же изменения во всех этих местах.

Но существуют случаи, когда следует применить денормализацию. Денормализация - это не недоделанная нормализация, это намеренное нарушение нормальных форм, для увеличения производительности. Рассмотрим некоторые ситуации, когда требуется применить денормализацию:

1) **Расчётные значения**. Зачастую медленно выполняются и потребляют много ресурсов запросы, в которых производятся какие-то сложные вычисления, особенно при использовании группировок и агрегатных функций (Sum, Max и т.п.). Иногда имеет смысл добавить в таблицу 1-2 дополнительных столбца, содержащих часто используемые (и сложно вычисляемые) расчетные данные. В этом случае для получения требуемого результата достаточно извлечь из данного столбца предварительно рассчитанные значения. Создание столбца, содержащего предварительно рассчитываемые значения, позволяет значительно сэкономить время при выполнении запроса, однако требует своевременного изменения данных в этом столбце.
2) **Большое количество соединений и таблиц**. В запросах к полностью нормализованной базе нередко приходится соединять до десятка, а то и больше, таблиц. А каждое соединение — операция весьма ресурсоемкая. Как следствие, такие запросы кушают ресурсы сервера и выполняются медленно. В этом случае Лучше объединять в одну несколько таблиц, имеющих небольшой размер, содержащих редко изменяемую информацию, причем информацию, по смыслу тесно связанную между собой.