**Министерство образования и науки Российской Федерации**

**федеральное государственное автономное образовательное учреждение высшего образования**

**“САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ,**

**МЕХАНИКИ И ОПТИКИ”**

Факультет информационных технологий и программирования

Образовательная программа Корпоративные информационные системы

Направление подготовки 09.03.03 Прикладная информатика

О Т Ч Е Т

об учебной практике

Тема задания: Developing SQL Databases

Обучающийся Балашов Степан Алексеевич, M3300

Руководитель практики от профильной организации: Повышев В.В.

Руководитель практики от университета: Зубок Д. А., доцент

Практика пройдена с оценкой \_\_\_\_

Подписи членов комиссии:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *Зубок Д. А.*

(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *Маятин А. В.*

(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *Повышев В. В.*

(подпись)

Дата \_\_\_\_

Санкт-Петербург

2019

**Модуль 1: Designing a Normalized Relational Database Schema**

Нормализация баз данных

Для поддержания БД в устойчивом состоянии используется ряд механизмов, которые получили обобщенное название средств ***поддержки целостности***. Приведение структуры БД в соответствие этим ограничениям - это и есть **нормализация**.

**Первая нормальная форма**(1НФ) говорит, что каждый атрибут отношения должен хранить атомарное значение, каждое отношение (строка в таблице) должно содержать одинаковое количество атрибутов (столбцов), т.е.

* 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

**Вторая нормальная форма** (2НФ) говорит, что отношение находится во второй нормальной форме, если оно находится в 1НФ, и при этом все неключевые атрибуты зависят только от первичного ключа, т.е.

* Вторая нормальная форма требует, чтобы неключевые столбцы таблиц зависили от первичного ключа в целом, но не от его части.
* Если таблица находится в первой нормальной форме и первичный ключ у нее состоит из одного столбца, то она автоматически находится и во второй нормальной форме.

Отношение находится в **третьей нормальной форме** (3НФ), если оно находится во второй нормальной форме и каждый неключевой атрибут зависит только от первичного ключа и не зависят друг от друга.

**Денормализация** — намеренное приведение структуры базы данных в состояние, не соответствующее критериям [нормализации](https://ru.wikipedia.org/wiki/%D0%9D%D0%BE%D1%80%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%84%D0%BE%D1%80%D0%BC%D0%B0), обычно Денормализация (англ. denormalization) — намеренное приведение структуры базы данных в состояние, не соответствующее критериям нормализации, обычно проводимое с целью ускорения операций чтения из базы за счет добавления избыточных данных.проводимое с целью ускорения операций [чтения](https://ru.wikipedia.org/wiki/Select_(SQL)) из базы за счет добавления избыточных данных.

**Модуль 2: Designing and Implementing tables**

*Типы данных в Transact-SQL*

*Числовые типы данных*

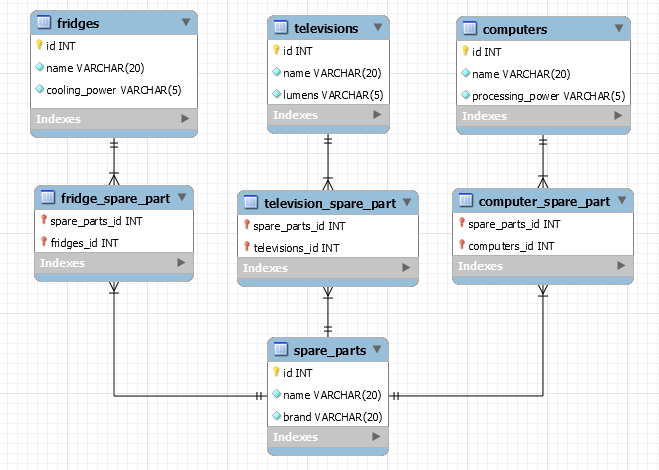
* INT: хранит числа от –2 147 483 648 до 2 147 483 647. Занимает 4 байта.
* DECIMAL: хранит числа c фиксированной точностью. Занимает от 5 до 17 байт в зависимости от количества чисел после запятой.

*Строковые типы данных*

* CHAR: хранит строку длиной от 1 до 8 000 символов.
* VARCHAR: хранит строку. На каждый символ выделяется 1 байт.
* NVARCHAR: хранит строку в кодировке Unicode. На каждый символ выделяется 2 байта.

*Типы данных, представляющие дату и время*

* TIME: хранит время в диапазоне от 00:00:00.0000000 до 23:59:59.9999999. Занимает от 3 до 5 байт.
* DATETIME: хранит даты и время от 01/01/1753 до 31/12/9999. Занимает 8 байт.
* DATETIME2: хранит даты и время в диапазоне от 01/01/0001 00:00:00.0000000 до 31/12/9999 23:59:59.9999999. Занимает от 6 до 8 байт в зависимости от точности времени.



*Схемы в Transact-SQL*

*Схема* — структура базы данных, описанная на формальном языке, поддерживаемом СУБД. В реляционных базах данных схема определяет таблицы, поля в каждой таблице, и ограничения целостности (первичный, потенциальные и внешние ключи и другие ограничения).

**Модуль 3: Ensuring Data Integrity Through Constrains**

Под ***целостностью*** понимают свойство базы данных, означающее, что она содержит полную, непротиворечивую и адекватно отражающую пред­метную область информацию.

Поддержание целостности БД включает проверку (контроль) целостнос­ти и ее восстановление в случае обнаружения противоречий в базе. Целост­ное состояние БД задается с помощью *ограничений целостности* в виде усло­вий, которым должны удовлетворять хранимые в базе данные.

Среди ограничений целостности можно выделить два основных типа ограничений: *ограничения значений* атрибутов отношений и *структурные ограничения* на кортежи отношений.

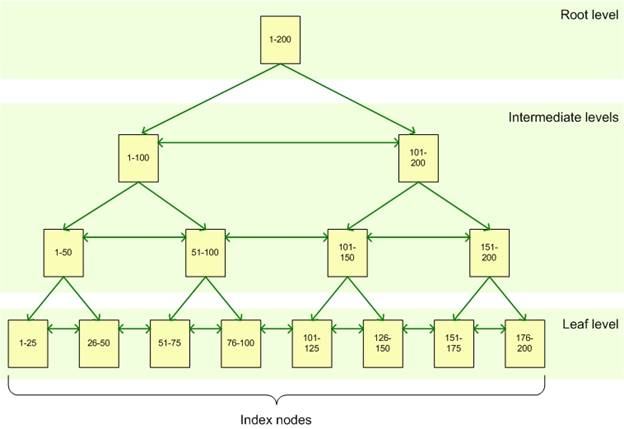
Примером ***ограничений значений*** атрибутов отношений является тре­бование недопустимости пустых или **повторяющихся значений в** атрибу­тах, а также контроль принадлежности значений атрибутов заданному **диа­пазону.** Так, в записях отношений о кадрах значения атрибута Дата рожде­ния не могут превышать значений атрибута Дата приема.

Наиболее гибким средством реализации контроля **значений** атрибутов яв­ляются *хранимые процедуры* **и** *триггеры,* имеющиеся в некоторых СУБД.

***Структурные ограничения*** определяют требования *целостности сущ­ностей* **и** *целостности ссылок.* Каждому экземпляру сущности, представлен­ному в отношении, соответствует только один его кортеж. Требование *цело­стности сущностей* состоит в том, что любой кортеж отношения должен быть отличим от любого другого кортежа этого отношения, т. е., иными сло­вами, любое отношение должно обладать первичным ключом.

Формулировка требования целостности ссылок тесно связана **с** понятием *внешнего ключа.* Напомним, что внешние ключи служат для связи **отношений** (таблиц БД) между собой. При этом атрибут одного отношения (родительс­кого) называется ***внешним ключом*** данного отношения, если он является *пер­вичным* ключом другого отношения (дочернего). Говорят, что отношение, **в** котором определен внешний ключ, ссылается на отношение, в котором **этот** же атрибут является первичным ключом.

***Ссылочная целостность*** – это ограничение базы данных, гарантирующее, что ссылки между данными являются действительно правомерными и неповрежденными. Ссылочная целостность является фундаментальным принципом теории баз данных и проистекает из той идеи, что база данных должна не только сохранять данные, но и активно содействовать обеспечению их качества.

**Модуль 4: Creating Indexes**

Индекс — объект базы данных, создаваемый с целью повышения производительности поиска данных. Таблицы в базе данных могут иметь большое количество строк, которые хранятся в произвольном порядке, и их поиск по заданному критерию путём последовательного просмотра таблицы строка за строкой может занимать много времени.

Индекс состоит из набора страниц, узлов индекса, которые организованы в виде древовидной структуры — *сбалансированного дерева*. Эта структура является иерархической по своей природе и начинается с корневого узла на вершине иерархии и конечных узлов, листьев, в нижней части, как показано на рисунке.

*Глубина индекса* - это количество уровней от корневого узла индекса до конечных узлов. Глубокий индекс будет страдать от проблемы снижения производительности. Напротив, индекс с большим количеством узлов на каждом уровне может создать очень плоскую структуру индекса. Индекс только с 3 до 4 уровнями очень распространен.

*Плотность индекса* является мерой отсутствия уникальности данных в таблице. Плотный столбец - это столбец с большим количеством дубликатов.

*Селективность индекса* - это показатель количества отсканированных строк по сравнению с общим числом строк. Индекс с высокой селективностью означает небольшое количество строк, отсканированных по отношению к общему количеству строк.

*Составной индекс* может содержать более одного столбца. Вы можете включить до 16 столбцов в индекс, но их общая длина ограничена 900 байтами. Как кластеризованный, так и некластеризованный индексы могут быть составными.

*Уникальный индекс* обеспечивает уникальность каждого значения в индексируемом столбце. Если индекс составной, то уникальность распространяется на все столбцы индекса, но не на каждый отдельный столбец. Уникальный индекс автоматически создается когда вы определяете ограничения столбца: первичный ключ или ограничение на уникальность значений:

**Модуль 5: Introduction to Transactions and Managing**

В [информатике](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0) [акроним](https://ru.wikipedia.org/wiki/%D0%90%D0%BA%D1%80%D0%BE%D0%BD%D0%B8%D0%BC) **ACID** описывает требования к [транзакционной системе](https://ru.wikipedia.org/wiki/%D0%A2%D1%80%D0%B0%D0%BD%D0%B7%D0%B0%D0%BA%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0) (например, к [СУБД](https://ru.wikipedia.org/wiki/%D0%A1%D0%A3%D0%91%D0%94)), обеспечивающие наиболее надёжную и предсказуемую её работу. Требования ACID были в основном сформулированы в конце 70-х годов [Джимом Греем](https://ru.wikipedia.org/wiki/%D0%94%D0%B6%D0%B8%D0%BC_%D0%93%D1%80%D0%B5%D0%B9)[[1]](https://ru.wikipedia.org/wiki/ACID#cite_note-1).

**Модуль 6: Isolation Levels**

*Уровни изоляции транзакции* - это мера степени успешности изоляции транзакции. В частности, уровни изоляции транзакций определяются наличием или отсутствием следующих явлений:

* "Грязных" операций чтения
* Неповторяющееся чтение
* Фантомные объекты

Все уровни изоляции подразделяют на типы:

* *Пессимистичное изолирование* - подразумевает, что конфликты могут быть.
  + READ UNCOMMITED - работа с транзакциями, которые не выполнились
  + READ COMMITED - работа с транзакциями, которые выполнились
  + REPEATABLE READ - изолирование данных, с которыми идет работа
  + SERIALIZABLE - изолирование данных данных, с которыми потенциально может идти работа
* *Оптимистичное изолирование* - подразумевает невозможность конфликтов.
  + Row versioning - система, подразумевающая сохранение всех изменений во временном производительном хранилище, создание логики обработки ошибок, возможность применения частных методов пессимистичного изолирования в запросах.