Лабораторная работа N 1. Общение информации по созданию реляционной базе данных.

Цель - создание реляционной базы данных.

Теория:

Да́нные -зарегистрированная информация; представление фактов, понятий или инструкций в форме, приемлемой для общения, интерпретации, или обработки человеком или с помощью автоматических средств.

Данные — поддающееся многократной интерпретации представление информации в формализованном виде, пригодном для передачи, связи или обработки (ISO/IEC 2382:2015).

Данные — формы представления информации, с которыми имеют дело информационные системы и их пользователи.

- Входные данные – это информация, передаваемая системе с терминала или рабочей станции. Когда эта информация сохранена в таблицах, она становится частью постоянных данных или влечет за собой изменения постоянных данных.
- Выходные данные – это сообщения и результаты, выдаваемые системой на экран, печать и иное устройство вывода.

Модель данных – это совокупность структуры данных и операций их обработки.

Это способ представления данных в виде таблиц. Элементы: поле (столбец), запись (строка) и таблица (отношение).

Под реляционной системой понимается система, основанная на следующих принципах:

- данные пользователя представлены только в виде таблиц;

- пользователю предоставляются операторы, генерирующие новые таблицы из старых (для выборки данных).

Преимущества:

1. Простота. В такой модели всего одна информационная конструкция, формализующая табличное представление. Она наиболее привычна для пользователя.
2. Теоретическое обоснование. Существуют строгие методы нормализации данных в таблицах
3. Независимость данных. При изменении БД, ее структуры необходимы бывают лишь минимальные изменения прикладных программ.

Недостатки:

1. Низкая скорость, т.к. требуются операции соединения.
2. Большой расход памяти в силу организации всех данных в виде таблиц.

Задание

Создать реляционную базу данных на 10 таблиц. Тему выбирается произвольно.

Лабораторная работа N 2 Отношения

Цель: Изучить различные виды отношений в реляционной БД.

Теория:

Модели данных можно разделить на три категории:

-концептуальные (объектные) модели данных – описание данных высокого уровня (уровня объектов): модель типа «сущность-связь» или ER-модель (Entity-Relationship)

-логические модели данных (модели данных на основе записей) – БД состоит из логических записей фиксированного формата: сетевая модель данных (network),

иерархическая модель данных (hierarchical) и реляционная модель данных (relational)

-физические модели данных – описывают, как данные хранятся в компьютере (информация о структуре записей, порядке расположения записей и путях доступа к ним): обобщающая модель (unifying), модель памяти кадров (frame memory).

Одним из основных понятий являются отношения. Отношением называют вид и тип связи между таблицами. Существуют три основных вида отношения:

Один-к-одному - максимальная мощность отношения в обоих направлениях равна одному

Один-ко-многим - максимальная мощность отношения в одном направлении равна одному, а в другом – многим;

Многие-к-многим - максимальная мощность отношения в обоих направлениях равна многим

Иногда в литературе выделяют Многие-к-одному.

Отношение R, определенное на множестве доменов D1, D2, …, Dn (не обязательно различных), содержит две части: заголовок (строка заголовков столбцов в таблице) и тело (строки таблицы).

Заголовок содержит фиксированное множество атрибутов или пар вида <имя_атрибута : имя_домена>:

Тело содержит множество кортежей. Каждый кортеж, в свою очередь, содержит множество пар <имя_атрибута : значение _атрибута>:

Свойства отношений:

- отношение имеет уникальное имя (т.е. отличное от имен других отношений в БД);

- в любом отношении нет одинаковых кортежей (это свойство следует из того факта, что тело отношения – математическое множество (кортежей), а множества в математике по определению не содержат одинаковых элементов);
- в любом отношении кортежи не упорядочены (т.е. нет упорядоченности «сверху-вниз») (это свойство следует из того, что тело отношения – математическое множество, а простые множества в математике неупорядочены);
- в любом отношении атрибуты не упорядочены (т.е. нет упорядоченности «слева-направо») (это свойство следует из того, что заголовок отношения также определен как множество атрибутов);
- в отношении все значения атрибутов атомарные (неделимые) (это следует из того, что все лежащие в основе домены содержат только атомарные значения);

Задание

Реализовать БД на произвольную тему в которой будут использоваться 3 основных вида связи.

Лабораторная работа N 3 Ключи

Цель - получить базовые представление о видах и типах ключей.

Теория

Второй важной частью реляционной модели данных являются ключи. Ключи могут делиться на несколько видов:

1) Первичный ключ - уникальный идентификатор, позволяющий однозначно идентифицировать кортеж. Первичные ключи делиться на несколько типов

1.1 Простой первичный ключ - одна ячейка таблицы.

1.2 составной первичный ключ - ключ состоящий из 2 и более ячеек позволяющие однозначно идентифицировать кортеж.

1.3 Естественный первичный ключ - естественный первичный ключ выбирается из множества потенциальных ключей и являясь естественным признаком позволяет идентифицировать запись.

1.4 Искусственный первичный ключ - уникальный идентификатор введенных искусственно. Пример id

2) Потенциальный ключ - один или множество признаков позволяющих уникально идентифицировать кортеж (запись таблицы) характеризующий объект. В качестве примера может выступать вин-номер автомобиля.

3) Вторичные или внешне ключи - это ключ, используемый для объединения двух таблиц. Иногда его также называют ссылочным ключом. Так же могут быть составным или простым.

Задание

Создать схему БД которая будет использовать все типы первичных и вторичных ключей.

Лабораторная работа N 4 Связи, ссылочная целостность.

Теория

Целостность данных предназначена для сохранения в БД «отражения действительности реального мира», т.е. устранения недопустимых конфигураций (состояний) значений и связей (например, вес детали может быть только положительным), которые не имеют смысла в реальном мире (не следует путать с «безопасностью» - хоть и есть некоторое сходство, но безопасность относится к защите от несанкционированного доступа, а целостность – защита БД от санкционированных пользователей).

Правила целостности можно разделить на:

• специфические или корпоративные ограничения целостности – правила, определяемые пользователями или администратором БД, которые указывают дополнительные ограничения, специфические для конкретных БД (например, максимальные и минимальные значения некоторых атрибутов (диапазон оценок));

- общие правила целостности – правила, которые применимы к любой реляционной БД (относятся к потенциальным (первичным) и к внешним ключам).

Ссылочная целостность. Правило ссылочной целостности – база данных не должна содержать несогласованных значений внешних ключей (здесь «несогласованное значение внешнего ключа» - это значение внешнего ключа, для которого не существует отвечающего ему значения соответствующего потенциального ключа в соответствующем целевом отношении) – т.е. если В ссылается на А, тогда А должно существовать.

Задание

Модернизировать любую базу данных из созданных ранее под правила ссылочной целостности. Обосновать что полученная БД соответсвует правилам.

Лабораторная работа N 5. Создание инфологической модели данных

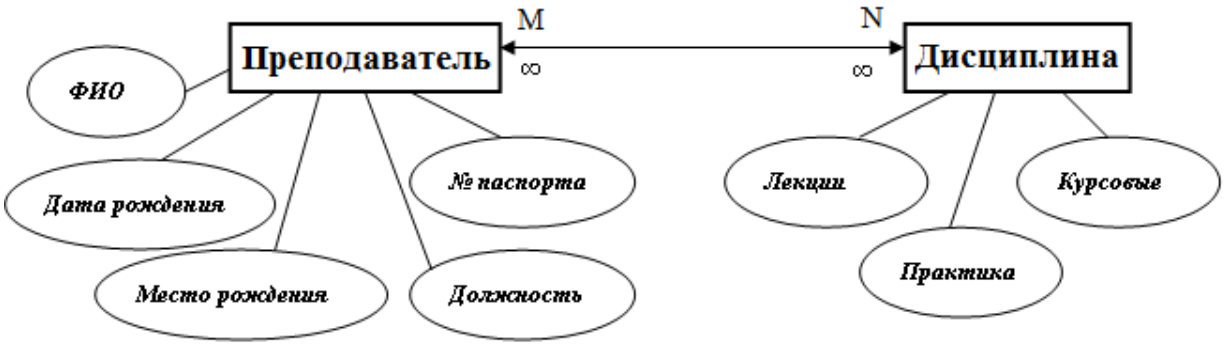
Цель: ознакомиться с концептуальным уровнем проектирования БД. Получить навыки абстрактного описания предметной области.

Теория:
На первом этапе исследуется предметная область, выявляются в ней объекты и процессы, которые нужно будет отобразить в информационной системе при решении задач, для которых разрабатывается информационная система. Модель, используемая на этом этапе, служит для наглядного представления семантических связей в предметной области. Строгая формализация структуры данных на этом этапе не обязательна. Такие модели называются инфологическими. В настоящее время наиболее распространённой инфологической моделью является модель сущность-связь.
Информационая система (ИС) создаётся для решения задач некоторой организации (завода, банка, вуза, библиотеки и т.д.). Для создания и эксплуатации ИС требуется её описание. Полное, исчерпывающее, описание ИС должно включать в себя не только саму ИС, но и окружающую среду, то есть, должно быть описанием *предметной области*.
Подробное описание предметной области можно дать в общем случае только в свободной форме. Для графического

описания абстрактной модели проектируемой системы используется UML (Unified Modeling Language - унифицированый язык моделирования).
Существенной, если не главной частью ИС являются хранящиеся в ней данные. При проектировании ИС данные нужно представить в виде простой модели, отображающей смысл данных, их взаимосвязь и не привязываться при этом к конкретному типу базы данных. Такие модели получили название *инфологических*.
Инфологическую модель можно построить, опираясь только на интуитивное представление о данных.

Пример простейшей инфологичесчкой модели. ← Объекты (типы сущностей)

Представление объектов на диаграмме: прямоугольник ← Свойства (атрибуты)



Представление свойств на диаграмме: в виде эллипсов с именем внутри него ← Отношения (типы связей)

Представление отношений на диаграмме: в виде ромба

Руководитель может управлять только *одним* отделом, отдел может управляться только *одним* руководителем

←

Лабораторная работа N 6. Преобразование инфологической модели в даталогическую.

Цель работы: преобразовать инфологическую модель в даталогическому виду. Провести нормализацию до 3го нормального уровня.

Теоретическая часть:

Даталогической называется такая модель которая отображает инфологи ческую модель с учетом выбранной модели данных и учитывает ограничения налогаемые моделью данных и типа СУБД который будет использоваться в дальнейшем для реализации схемы сущность-связь на физический уровень. Реляционная модель данных строится на основе модели сущность-связь. Каждой сущности из модели сущность-связь в реляционной модели ставится в соответствие отношение (таблица), каждому свойству сущности - атрибут отношения. В отношение, как правило, добавляется атрибут-счётчик, который служит формальным первичным ключом

(идентификатором кортежа). В базе данных кортеж - это строка или запись.

Связь один ко многим

Для задания связи *один ко многим* в отношеннии со стороны *многие* создаётся дополнительный атрибут *внешний ключ*. Внешний ключ принимает значения только из множества значений первичного ключа отношения со стороны *один*. ***Связь многие ко многим***

Для задания связи *многие ко многим* необходимо создать дополнительную таблицу связей, атрибутами которой служат внешние ключи, соответствующие первичным ключам связываемых таблиц.
Преобразование ER-диаграммы в реляционную модель (схему) позволяет

выполнить переход от концептуальной фазы проектирования к логической. Алгоритм преобразования следующий:

- 1) Каждая простая сущность (объект на ER-диаграмме) превращается в таблицу. Простая сущность - сущность, не являющаяся подтипом и не имеющая подтипов. Имя сущности становится именем таблицы.
  2) Каждый атрибут становится возможным столбцом с тем же именем; при этом может выбираться более точный формат. Столбцы, соответствующие необязательным атрибутам, могут содержать неопределенные (NULL) значения; столбцы, соответствующие обязательным атрибутам, - не могут.

- 3) Компоненты уникального идентификатора сущности (уникальные атрибуты объекта) превращаются в первичный ключ таблицы. Если имеется несколько возможных уникальных идентификаторов, выбирается наиболее используемый. Если в состав уникального идентификатора входят связи, к числу столбцов первичного ключа добавляется копия уникального идентификатора сущности, находящейся на дальнем конце связи (этот процесс может продолжаться рекурсивно). Для именования этих столбцов используются имена концов связей и/или имена сущностей.

- 4) Связи многие-к-одному (и в частности - один-к-одному) становятся внешними ключами. Внешний ключ добавляется в виде столбца (столбцов) в таблицу, соответствующую объекту со стороны «многие» связи. Необязательные связи соответствуют столбцам, допускающим неопределенные значения; обязательные связи - столбцам, не допускающим неопределенные значения. Связи этого типа, имеющие дополнительные атрибуты, желательно реализовывать через промежуточную таблицу, как описано в п. 5 алгоритма.

- 5) Связи многие-ко-многим реализуются через промежуточную таблицу. Эта таблица будет содержать внешние ключи на соответствующие объекты, а также другие атрибуты, которые описывают указанную связь. Первичный ключ промежуточной таблицы должен

включать в себя как минимум внешние ключи на объекты связи. Также в первичный ключ должны входить атрибуты временной связи (например, дата события), для наиболее полной реализации модели.

- 6) Если в концептуальной схеме присутствовали подтипы, то возможны два способа: (а) все подтипы в одной таблице и (б) для каждого подтипа - отдельная таблица. При применении способа (а) таблица создается для наиболее внешнего супертипа, а для подтипов могут создаваться представления. В таблицу добавляется по крайней мере один столбец, содержащий код ТИПА; он становится частью первичного ключа. При использовании метода (б) для каждого подтипа первого уровня (для более нижних - представления) супертип воссоздается с помощью представления UNION (из всех таблиц подтипов выбираются общие столбцы - столбцы супертипа).
  7) Имеется два способа работы при наличии исключающих связей: (а) общий домен и (б) явные внешние ключи. Если остающиеся внешние ключи все в одном домене, т.е. имеют общий формат (способ (а)), то создаются два столбца: идентификатор связи и идентификатор сущности. Столбец идентификатора связи используется для различения связей, покрываемых дугой исключения. Столбец идентификатора сущности используется для хранения значений уникального идентификатора сущности на дальнем конце соответствующей связи. Если результирующие внешние ключи не относятся к

одному домену, то для каждой связи, покрываемой дугой исключения, создаются явные столбцы внешних ключей; все эти столбцы могут содержать неопределенные значения.
Данный алгоритм применяется ко всем сущностям и связям ER- диаграммы. Нормализация - процесс приведения отношений БД к определенным правилам.

Лабораторная работа N 7. Нормализация базы данных.

Цель закрепление знаний по нормализации БД

Теория

Процесс проектирования БД с использование метода НФ является итерационным и заключается в последовательном переводе отношения из 1НФ в НФ более высокого порядка по определенным правилам. Каждая следующая НФ ограничивается определенным типом функциональных зависимостей и устранением соответствующих аномалий при выполнении операций над отношениями БД, а также сохранении свойств предшествующих НФ.
Атрибут — свойство некоторой сущности. Часто называется полем таблицы.

Домен атрибута — множество допустимых значений, которые может принимать атрибут.

Кортеж — конечное множество взаимосвязанных допустимых значений атрибутов, которые вместе описывают некоторую сущность (строка таблицы).

Отношение — конечное множество кортежей (таблица).

Схема отношения — конечное множество атрибутов, определяющих некоторую сущность. Иными словами, это структура таблицы, состоящей из конкретного набора полей.

Проекция — отношение, полученное из заданного путём удаления и (или) перестановки некоторых атрибутов.

Функциональная зависимость между атрибутами (множествами атрибутов) X и Y означает, что для любого допустимого набора кортежей в данном отношении: если два кортежа совпадают по значению X, то они совпадают по значению Y. Например, если значение атрибута «Название компании» — Canonical Ltd, то значением атрибута «Штаб-квартира» в таком кортеже всегда будет Millbank Tower, London, United Kingdom. Обозначение: {X} -> {Y}.

Нормальная форма — требование, предъявляемое к структуре таблиц в теории реляционных баз данных для устранения из базы избыточных функциональных зависимостей между атрибутами (полями таблиц).

Метод нормальных форм (НФ) состоит в сборе информации о объектах решения задачи в рамках одного отношения и последующей декомпозиции этого отношения на несколько взаимосвязанных отношений на основе процедур нормализации отношений.

Цель нормализации: исключить избыточное дублирование данных, которое является причиной аномалий, возникших при добавлении, редактировании и удалении кортежей(строк таблицы).

Аномалией называется такая ситуация в таблице БД, которая приводит к противоречию в БД либо существенно усложняет обработку БД. Причиной является излишнее дублирование данных в таблице, которое вызывается наличием функциональных зависимостей от не ключевых атрибутов.

Аномалии-модификации проявляются в том, что изменение одних данных может повлечь просмотр всей таблицы и соответствующее изменение некоторых записей таблицы.

Аномалии-удаления — при удалении какого либо кортежа из таблицы может пропасть информация, которая не связана на прямую с удаляемой записью.

Аномалии-добавления возникают, когда информацию в таблицу нельзя поместить, пока она не полная, либо вставка записи требует дополнительного просмотра таблицы.
Отношение находится в 1НФ, если все его атрибуты являются простыми, все используемые домены должны содержать только скалярные значения. Не должно быть повторений строк в таблице.
Отношение находится во 2НФ, если оно находится в 1НФ и каждый не ключевой атрибут неприводимо зависит от Первичного Ключ
Отношение находится в 3НФ, когда находится во 2НФ и каждый не ключевой атрибут нетранзитивно зависит от первичного ключа. Проще говоря, второе правило требует выносить все не ключевые поля, содержимое которых может относиться к нескольким записям таблицы в отдельные таблицы.

Задание Привести к третьей нормальной форме БД полученную в прошлой лабораторной работе.

Лабораторная работа N 8. Способы создания реляционной БД.

Цель работы: рассмотреть несколько способов создания физической БД на языке SQL. Сравнить эффективность данных способов.
Теория:

Теоретические сведения

Очень рекомендуется при работе в этих средах почаще пользоваться клавишей F1.

Power Designer

Как известно, реляционная база данных состоит из таблиц. На ER-модели таблицы представляются в виде сущностей. Для того, чтобы разместить сущность на схеме, выберем значок сущности (прямоугольник с горизонтальной чертой посередине) и будем щелкать им в нужных местах в окне схемы. После того, как мы поставили необходимое нам количество этих сущностей (количество которых и будет количеством таблиц в нашей базе данных), можно перейти к заданию атрибутов сущностей. Атрибуты ER-модели в конечном итоге будут преобразованы в столбцы таблиц. Таким образом, задавая атрибуты, мы задаем количество, названия и типы столбцов таблицы базы данных. Но перед тем, как определять столбцы (или атрибуты) необходимо сначала задать множества допустимых значений атрибутов, или так называемые домены (domains). Домен по сути дела является типом столбца, ограничивая значения, которые можно записать в данный столбец.

пределение домена производится следующим образом:

• Из меню Dictionary выбираем пункт List of Domains... если мы создаем новую модель, то он пуст, если же мы редактируем ранее созданную модель, или

модель, полученную с помощью процедуры Reverse Engineering, то список будет содержать ранее определенные или существующие в базе данных домены.

• Для того, чтобы определить новый домен просто щелкаем мышкой в пустой строчке и начинаем вводить данные

• Данные основной таблицы представляют из себя следующее:

Name — название домена, как он будет представляться в полях для выбора типов атрибутов

Code — уникальный идентификатор атрибута для внутреннего представления

DataType — базовый тип домена, который можно задать либо вписав его в строку, либо выбрав из выпадающего при нажатии на кнопку [...] диалога. На этот базовый тип затем можно накладывать дополнительные ограничения New, Delete —Создание, удаление домена SortBy — не влияет на реальные данные, просто показывает порядок сортировки записей доменов

1. Check — Открывает диалог редактирования дополнительных ограничений, в котором можно задать:

2.минимальные, максимальные и значения по умолчанию (поля Minimum, Maximum, Default),

3. Unit — задание единицы измерения параметра, Format — задает маску формата данных,

4. Lowercase — переводит все символы в нижний регистр,
5. Uppercase — переводит все значения в верхний регистр,
6. Cannot modify — запрещает изменение данных в колонке физической таблицы
7. List of values — позволяет задать список значений, которые может принимать атрибут, соответствующий доменуПри нажатии на кнопку Rules можно задать бизнес-правила, характеризующие этот домен. Кнопки Add и Remove позволяют добавлять и удалять правила, кнопка List показывает список определенных на данный момент бизнес-правил. Но вернемся обратно в предыдущее окно List of Domains.

• При нажатии на кнопки Describe и Annotate вызывается редактор, в котором на обыкновенном человеческом языке можно описать предназначение этого

домена

• Кнопка Used by выдает список атрибутов сущностей, которые имеют тип, соответствующий текущему домену. Это просто информационное окно, программа не допускает редактирования этой информации.

И теперь, после создания всех необходимых доменов, можно начать создавать атрибуты сущностей, которые, как уже говорилось ранее, станут столбцами таблиц. Для этого два раза щелкнем мышкой по соответствующей сущности. Вызывается диалог редактирования свойств сущности, в котором мы можем задать название сущности, которое будет отображаться на схеме, уникальный идентификатор сущности, примерное количество записей в таблице (поле Number), и необходимость или отсутствие необходимости генерировать код для данной сущности. Вкладки Description

и Annotation позволяют нам, как уже описывалось выше, сделать описание сущности на понятном для пользователя языке. При нажатии на кнопку Attributes мы попадаем в диалоговое окно, позволяющее нам задавать и изменять атрибуты для выбранной сущности. Вид редактора атрибутов достаточно сильно похож на редактор доменов, поэтому остановимся лишь на некоторых отличающихся деталях. В конце каждой строки находятся Check box'ы, помеченные как I, M, D. Их содержимое для текущего атрибута дублируется в полях Identifier, Mandatory, Display ниже основного поля таблицы. Включение галочки в поле Identifier позволяет назначить этому атрибуту свойства ключа. Такие атрибуты подчеркиваются при отображении на схеме включение галочки в поле Mandatory указывает на то. Что это поле должно быть обязательно заполнено, ну и включение галочки Display свидетельствует о том, что этот атрибут будет выведен в поле сущности, выключение же галочки скрывает атрибут. В диалоге редактирования атрибутов появилось новое поле Domains, где из выпадающего списка можно выбрать соответствующий домен для создаваемого атрибута. При нажатии на кнопку [...] справа от этого поля, вызывается уже ранее рассмотренный редактор доменов, в котором можно доопределить недостающие домены или изменить определения уже существующих. Вернемся к предыдущему окну. Кнопка Rules в окне

определения свойств сущности позволяет определить бизнес правила для сущности.

После определения атрибутов сущностей можно заняться установлением отношений между сущностями. Для этого из палитры инструментов выбираем значок связи (две

сущности с линией между ними) и рисуем связи между сущностями. После рисования связи необходимо определить ее свойства. Для этого два раза щелкнем мышкой по связи. Перед нами — окно редактирования свойств связи. В верхней части окна представлена схема из двух сущностей и связи. При изменении параметров в этом диалоге, вид связи меняется. Ниже представлены две кнопки, каждая из которых соответствует сущности, с которой ассоциирована данная связь. При нажатии на одну из этих кнопок вызывается ранее рассмотренный диалог редактирования свойств сущности. Поля Name и Code задают метку сущности и ее уникальный идентификатор. Переключатель Cardinality позволяет определить вид связи (один ко многим, один к одному и т.д.) Ниже в окне диалога представлены свойства каждой стороны связи.

После определения всех связей и сущностей, необходимо проверить получившуюся модель. Для этого выберем из меню Dictionary пункт Check Model. В появившемся окне будут выведены найденные ошибки и предупреждения. Для создания физической модели базы данных из меню Dictionary выбираем пункт Generate Physical Model, где после ответа на вопрос о используемой СУБД будет создана физическая модель.

ER-Win

Работа с этой оболочкой во многом схожа с работай в среде Power Designer. Небольшие отличия будут описаны ниже.

• Определение атрибутов: атрибуту можно присваивать тип ранее определенного домена, который выбирается из правого окна редактора атрибутов, в то время как в Power

Designer не обязательно типом поля является определение домена.

• Немного отличаются редакторы отношений Relationship Editor.
• Определение доменов, атрибутов и т.д. доступно через меню Edit

• Генерация моделей, реинжениринг структуры и т.д. вынесены в отдельный пункт верхнего меню Tasks.

• Для некоторых типов баз данных (включая InterBase) имеются заранее определенные типы триггеров для поддержания ссылочной целостности и

проведения каскадных обновлений. Эти шаблоны определений включаются в сгенерированный программой скрипт создания базы данных.

Задание:

1. Построить ER-модель нормализованную бд и реализовать ее в средах Power Designer и ER-Win или Rational Rose(Использовать реинжиниринг структуры базы данных запрещается)

2. Создать физическую модель базы данных
3. Сгенерировать скрипт создания базы данных и создать саму базу

4. Создать скрип бд используя средства sql встроенные в субд. (возможно использование любого sql-СУБД за исключением Microsoft Access и систем SQLite)

5. Провести сравнение баз данных полученных в первом и втором случает

Лабораторная работа N 9 Операции управления структурами базы данных

Цель: проанализировать основные операции манипулирования данными.

Задание

Изучите синтаксис SQL-конструкций для определения структуры данных, найдите раздел на сайте mysql.com в котором приводится перечисление команд DDL

Изучите SQL-запросы по созданию таблиц, который генерируются при создании базы из MySQL Workbench по спроектированной структуре

Создайте индексы

Создайте триггеры

Создайте процедуры

Создайте функции

Изучите работу команд INSERT, UPDATE, DELETE

Проанализировать и сфрмулировать отличия процедур, функций и триггеров.

Лабораторная работа N 10 Управление данными

Цель: получить навыки уверенной работы с языком DML.

Задание:

Выполнить запросы аналогичные тем, которые прописаны в задании для курсового проектирования.

Разобраться с работой основных команд DML.

Лабораторная работа N 11. Создание прикладных программ для обработки БД.

Цель: научиться созданию простейших прикладных программ для взаимодействия с данными в СУБД.

Задание:

Выполнить запросы аналогичные тем, которые прописаны в задании для курсового проектирования

1. Создать desktop приложение (язык создания на Ваш выбор). Приложение должно иметь главное окно, из которого будет открываться окна с таблицами. На главном окне должно быть реализована возможность выбрать любую из существующих таблиц и запросов. В окнах с таблицами должны отображаться сами таблицы. Клавиши управления данными в таблице (обновление таблицы, удаление данных, редактирование данных, добавление данных).

2. Создать Web-приложение. Функционал аналогичен desktop приложению. Язык программирования для Web-приложения не ограничен.