Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«Северо-Кавказский федеральный университет»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к выполнению лабораторных работ
по дисциплине «**Системы обработки и хранения данных в электроэнергетике**»
для студентов направления
13.03.02 «Электроэнергетика и электротехника»
Направленность (профиль): «Цифровые технологии в электроэнергетике»

Ставрополь 2025

Содержание

Лабораторная работа 1.Построение хранилища данных с помощью Deductor Studio

Теоретические сведения

В промышленной эксплуатации основными источниками данных для ХД являются системы оперативной обработки транзакций (OLTP - Online Transaction Processing).

OLTP-системы, как правило, автоматизируют хорошо структурированные и регламентированные, повторяющиеся задачи обработки данных. Поэтому основная их задача - выполнение максимального количества транзакций за короткие промежутки времени на основе фиксированного набора надёжных и безопасных средств ввода, модификации, удаления данных, а также формирования оперативной отчётности.

Таким образом, ХД представляет собой предметно-ориентированную информационную БД, предназначенную для анализа больших массивов данных и формирования отчётности с целью поддержки принятия решений в организации. Хранилища строятся на базе СУБД и СППР.

В настоящее время существует несколько моделей организации данных в ХД, основными из которых являются многомерная и реляционная.

В реляционных ХД данные содержатся в таблицах третьей нормальной формы. Их преимущество заключается в простоте разработки и управления. Недостатком является необходимость денормализации данных «на лету» при их извлечении из множества таблиц при выполнении сложных аналитических запросов. При формировании больших выборок это приводит к значительным задержкам в получении данных, требуемых для анализа.

В основе многомерных ХД лежит идея разделения данных на два вида – измерения и факты. Факты – это данные, количественно описывающие бизнес-процессы (суммы, количества, цены и т.д.), а измерения – качественные данные, формирующие контекст для фактов. Действительно,

любые количественные характеристики бессмысленны без контекста, который представляет собой объекты, связанные с этими характеристиками. Например, для количества необходимо указать объекты, для которых оно определяется; для цены – какие-то товары и услуги и т.д. Раздельное использование измерений и фактов не имеет смысла, поскольку, выражаясь языком статистики, при этом нарушается единство количественной и качественной сторон описываемых явлений и процессов.

Основным преимуществом многомерной модели данных является то, что она лучше отражает бизнес-модели организации, что делает её проще для понимания и использования конечными пользователями. Недостатками многомерной модели являются сложность загрузки из нескольких источников данных, а также сложность модификации многомерной структуры данных в хранилище при изменении бизнес-процессов на предприятии.

Как правило, непосредственное перемещение данных из источников в хранилище невозможно. Это связано с тем, что данные представлены в различных форматах, могут быть противоречивыми, дублирующими, неполными и т.д. В то же время в ХД данные должны храниться в едином формате, интегрироваться из источников различных типов, быть целостными и непротиворечивыми. Поэтому в процессе перемещения данных из источников в ХД предусмотрена специальная процедура, называемая ETL (Extract, Transform, Load) – извлечение, преобразование, загрузка.

Извлечение – процесс получения доступа к данным в источнике, их выгрузка и помещение в специальную область памяти, называемую промежуточной областью (staged area). Использование промежуточной области необходимо еще и потому, что различные источники, данные из которых должны быть интегрированы, могут располагаться как на локальных носителях, так и на сетевых, что обусловливает различное время их получения. Следовательно, данные, полученные раньше, должны где-то ожидать данные, которые поступят позже. Таким местом и является

промежуточная область. Данные в промежуточной области хранятся в виде временных таблиц, которые удаляются после завершения работы с данными.

Выгрузка данных из источников может производиться как средствами самого ХД, так и средствами ИС, в которой они содержатся (например, системы OLTP, АСУП и т.д.).

Преобразование – производится в промежуточной области и включает в себя преобразование данных различных типов и форматов к единому формату хранилища. Кроме этого, на данном этапе производится первичная предобработка данных с целью их дедупликации (исключения повторяющихся элементов данных), устранения противоречий, поиска ошибок, разбора составных значений (например, адресов) и других операций, определенных разработчиком системы.

Загрузка заключается в переносе данных из промежуточных таблиц в структуры ХД. При загрузке в хранилище переносится не вся информация из источников, а только та, которая была изменена в течение некоторого промежутка времени, прошедшего с предыдущей загрузки. При этом, как правило, формируется два потока данных: поток добавления и поток обновления. В первом потоке передаются новые, ранее не существовавшие данные, а во втором – данные, которые существовали и ранее, но были изменены со времени предыдущей загрузки.

Задание для выполнения

1. Развернуть хранилище данных на платформе Deductor Studio. Данные для ХД получить у преподавателя.

   Полученные данные содержат информацию о потреблении электроэнергии за месяц.

   Структура данных:

   - Дата месяца;
   - Время замера;
   - Температура в момент замера;

- Облачность в момент замера (значения от 0 до 3, где 0 – безоблачно, 3 – пасмурно);
- Ветер в момент замера (в м/с).

2. Построить OLAP-куб для анализа данных
3. Ответьте на контрольные вопросы.

Указания по технике безопасности

Техника безопасности при выполнении лабораторной работы совпадает с общепринятой для пользователей персональных компьютеров: самостоятельно не производить ремонт персонального компьютера, установку и удаление программного обеспечения; в случае неисправности персонального компьютера сообщить об этом обслуживающему персоналу лаборатории (оператору, администратору); соблюдать правила техники безопасности при работе с электрооборудованием; не касаться электрических розеток металлическими предметами; рабочее место пользователя персонального компьютера должно содержаться в чистоте; не разрешается возле персонального компьютера принимать пищу, напитки.

Защита работы

Перед выполнением лабораторной работы каждый студент получает индивидуальное задание. Защита лабораторной работы происходит только после выполнения индивидуального задания и оформления отчета по работе. При защите лабораторной работы студент отвечает на контрольные вопросы, приведенные в конце, и поясняет выполненное индивидуальное задание. Ход защиты лабораторной работы контролируется преподавателем.

Требования к отчету

Отчет должен содержать иллюстрации с подтверждением выполнения задания, пояснения к иллюстрациям, ответы на контрольные вопросы.

Пример выполнения задания

Необходимо выполнить процедуру развертывания и загрузки хранилища данных Deductor Warehouse на базе аналитической платформы (АП) Deductior Academic (программа доступна на сайте разработчика www.basegroup.ru).

Deductor Warehouse – многомерное кросс-платформенное ХД, входящее в состав аналитической платформы Deductor Enterprise и её учебной версии Deductor Academic. Хранилище позволяет консолидировать корпоративные данные для оперативного доступа к ним из аналитического приложения.

Первым шагом на пути к построению ХД является проектирование логической структуры данных на основе определенной бизнес-модели. Для этого рассмотрим источник данных (фрагмент таблицы).

data	time	Temp	MWt	wind	cloud
1	0	9	800	2	3
1	1	9	792	2	3
1	2	8	764	2	3
1	3	8	781	2	3
1	4	8	793	2	3
1	5	7	834	2	3

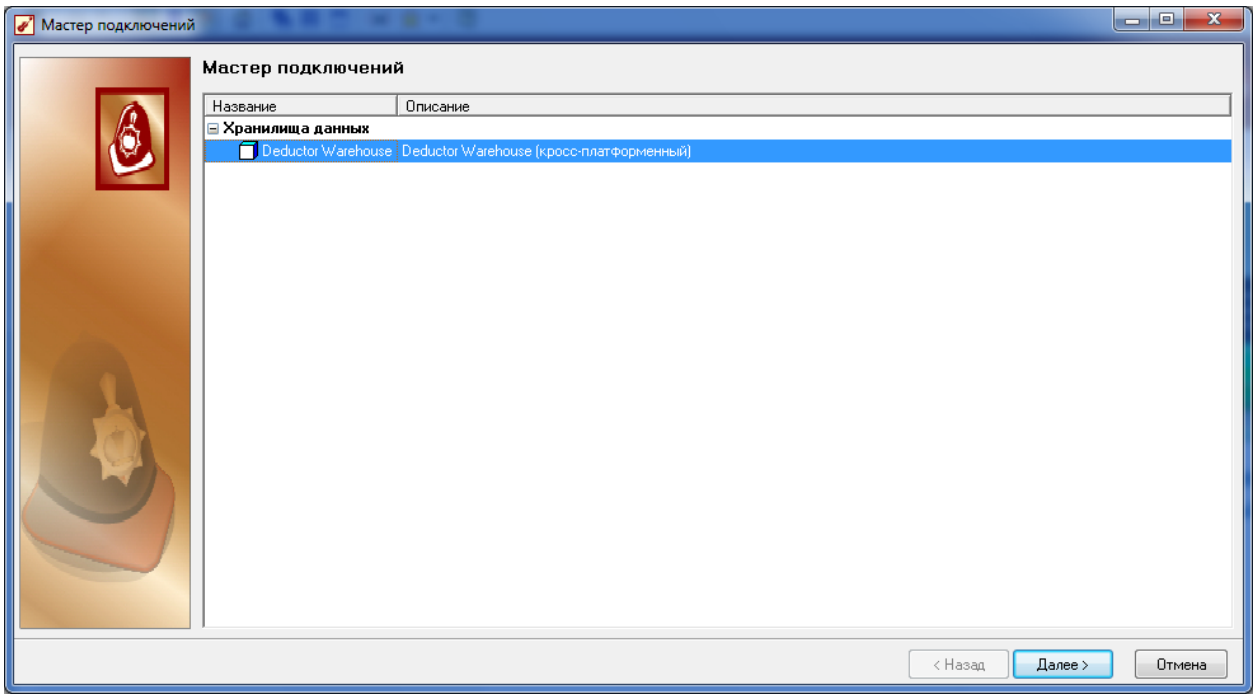
Запустить приложение Deductor Studio. Главное окно программы разделено на две части: панель сцена-риев (слева) и рабочую область (справа). В панели сценариев в виде иерархического дерева отображается последовательность операций по аналитической обработке данных начиная с импорта их из заданного источника.

В рабочей области отображаются таблицы с данными и результаты их аналитической обработки, графики и диаграммы, визуализаторы для представления аналитических моделей в виде карт, графиков, деревьев и т.д. В верхней части окна расположены панель инструментов и строка меню. Для

создания нового ХД требуется выполнить следующую последовательность действий.

1. Переходим на вкладку «Подключение» в меню «Вид» (или с помощью кнопки на панели инструментов). В результате вместо панели «Сценарии» в левой части окна откроется панель «Подключения», в которой следует воспользоваться кнопкой «Мастер подключений».

2. На 1-м шаге «Мастера подключений» выбрать тип объекта –Deductor Warehouse, после чего нажать «Далее».



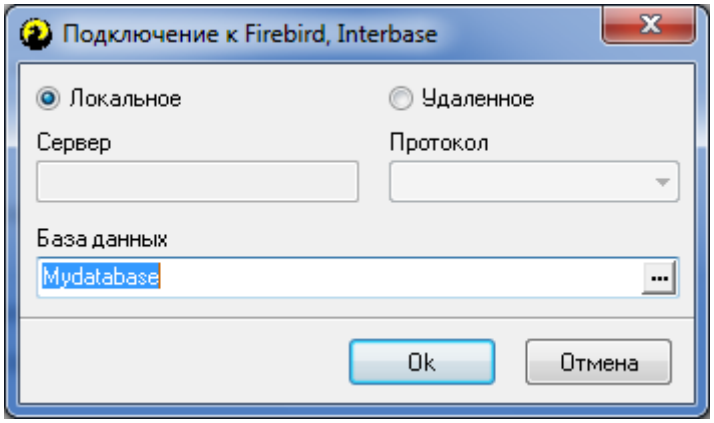
На 2-м шаге требуется определить параметры ХД

Тип базы данных – Firebird (будет выбрано по умолчанию).

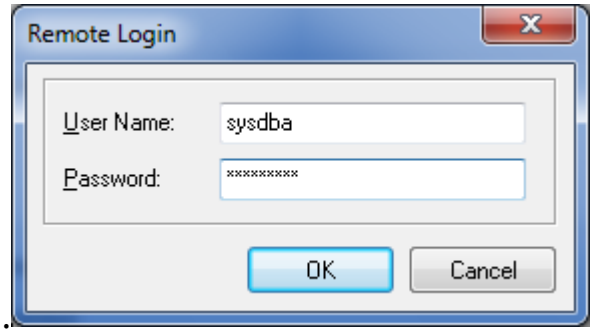
База данных – выбрать файл ХД. Для этого воспользоваться кнопкой «…», которая откроет окно выбора файла базы данных для ХД.

Выберите расположение нового ХД – локальное и нажмите на кнопку «…» в правой части строки «База данных». В результате откроется стандартное окно выбора файлов Windows. Если файл ХД уже существует (его можно узнать по расширению *.GDB), то его следует выбрать. В противном случае потребуется указать имя нового файла, например

mydatawarehouse.gdb, и нажать «Открыть». После этого окно выбора файла закроется, а указанное в нём имя файла отобразится в строке «База данных».



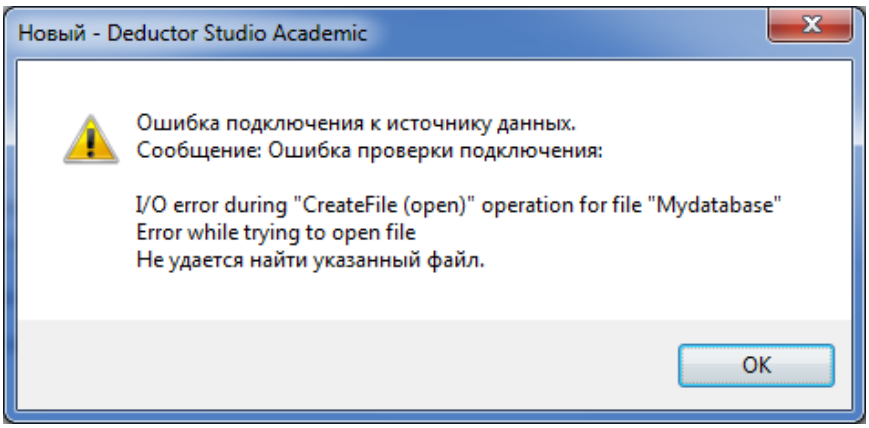
Логин/Пароль. Укажите логин и пароль для доступа к ХД. Дляэтого воспользуйтесь кнопкой «…», которая откроет окно.



Рекомендуется воспользоваться служебным логином sysdba и паролем masterkey.

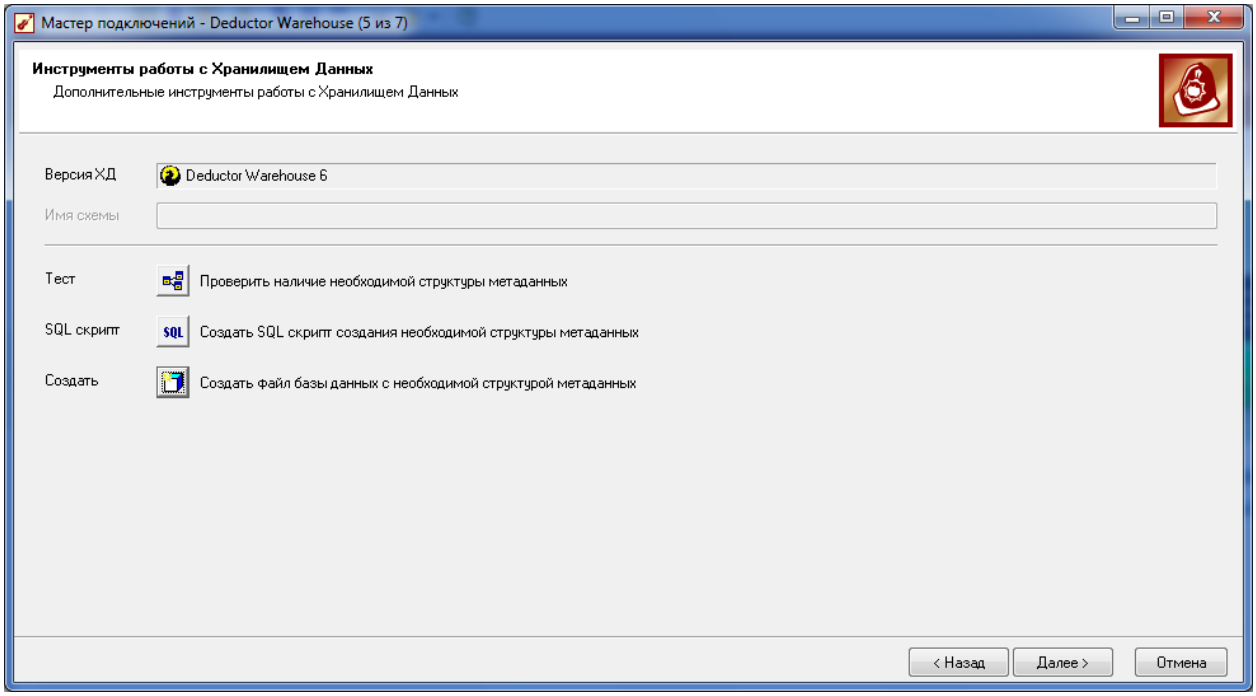
Список «Кодировка» и флажки раздела «Параметры» оставьте без изменений.

Тест подключения – кнопка позволяет проверить успешность подключения к существующему файлу ХД. В нашем случае, поскольку файл ХД с указанным именем еще не создан, попытка протестировать соединение приведёт к появлению сообщения об ошибке

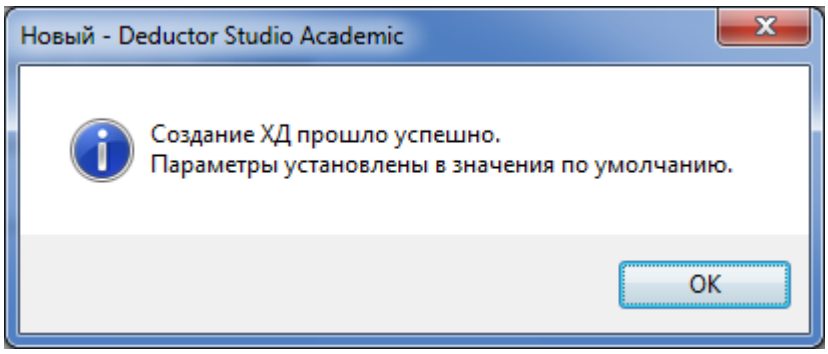


Чтобы создать новый файл ХД, перейдём на следующий шаг Мастера подключений, нажав копку «Далее».

На 3-м шаге «Мастера подключений» можно создать новый файл ХД с именем, заданным на 2-м шаге. Для этого достаточно воспользоваться кнопкой «Создать».

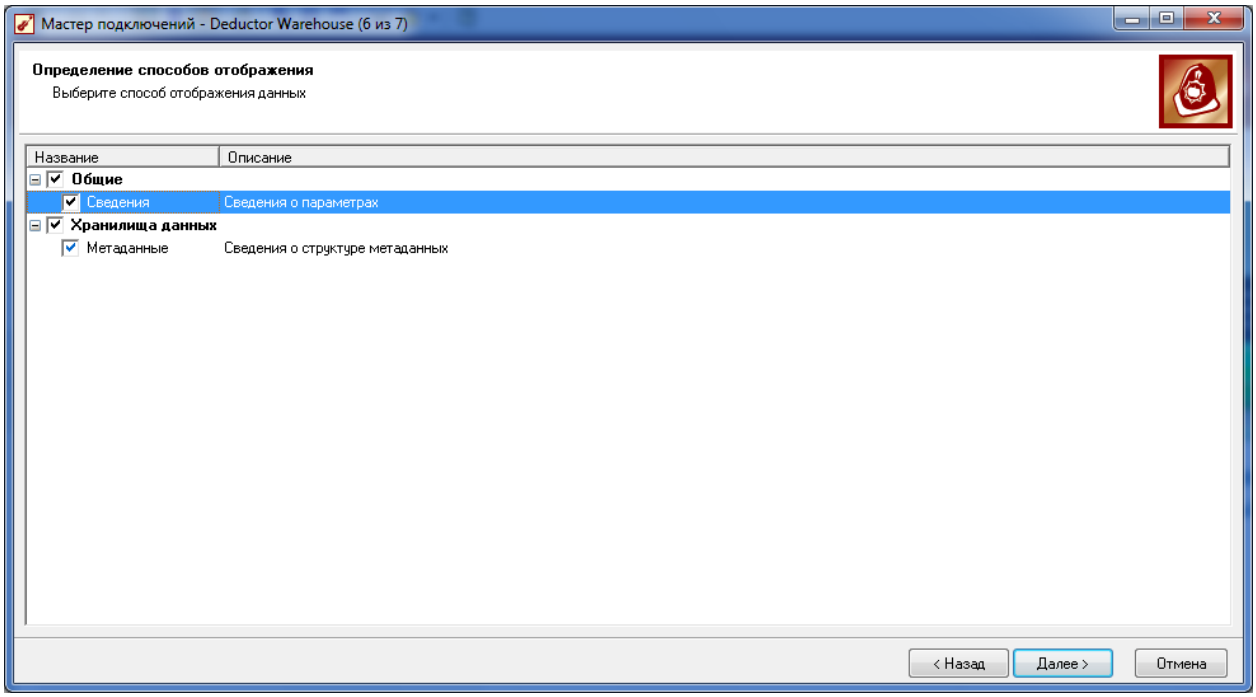


В случае успеха появится сообщение



Для проверки результатов создания структуры метаданных рекомендуется воспользоваться кнопкой «Тест». Если тест прошёл успешно, значит, файл ХД создан и готов к работе. После этого можно перейти к следующему шагу «Мастера подключений».

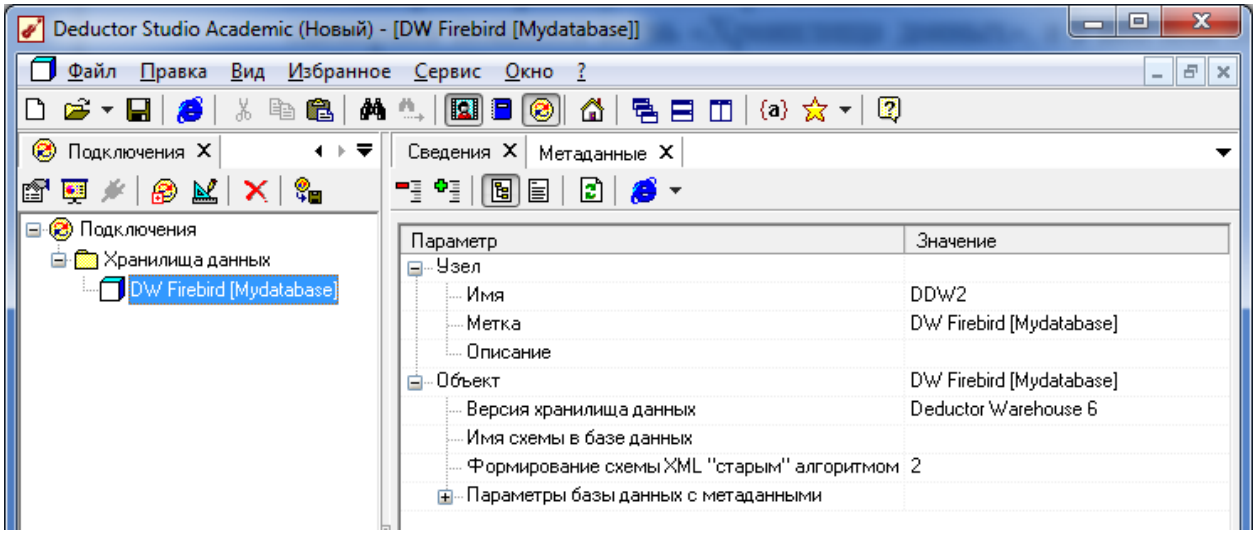
На 4-м шаге «Мастера» требуется выбрать способы визуализации результатов работы.



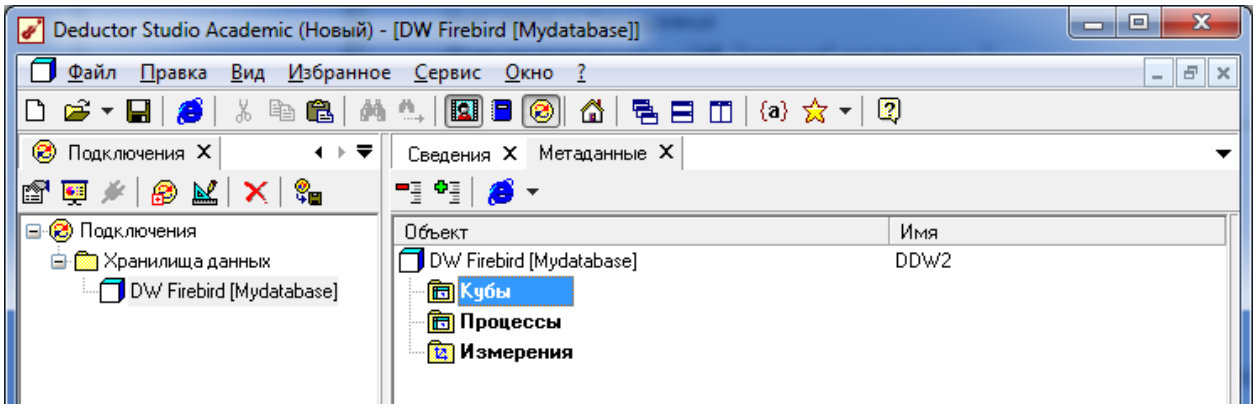
Можно выбрать сведения о параметрах нового ХД и о структуре его метаданных.

Настройки следующего, последнего, шага «Мастера подключений» носят служебный характер и их можно опустить. Кнопка «Готово» завершает работу «Мастера подключений».

В результате в панели сценариев появится ветвь «Хранилища данных», а в ней имя созданного файла ХД



В рабочей области окна программы будет создано две вкладки: «Сведения» и «Метаданные». Первая вкладка будет содержать информацию о новом ХД. Вкладка «Метаданные» отражает объекты метаданных, созданные по умолчанию

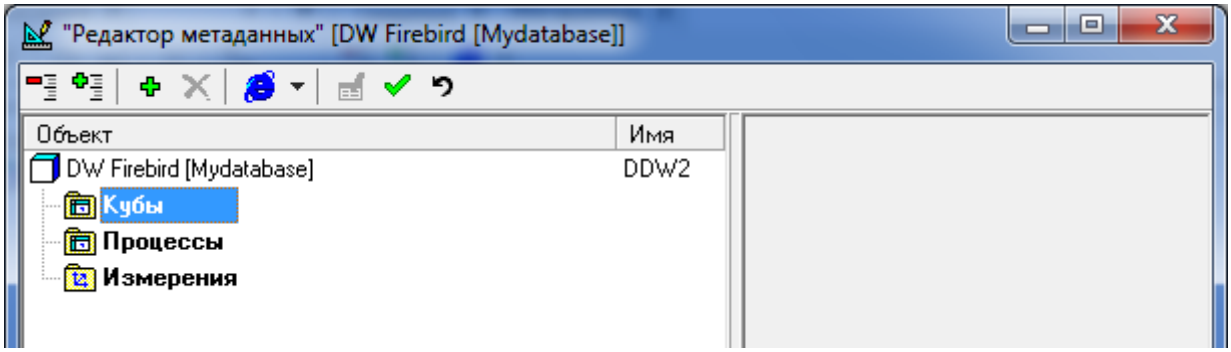


Таким образом, хранилище успешно создано и готово к работе.

Следующим этапом работы с ХД является создание структуры метаданных для загрузки таблиц с данными. Для этих целей используется редактор метаданных, открываемый с помощью кнопки на панели «Подключения».

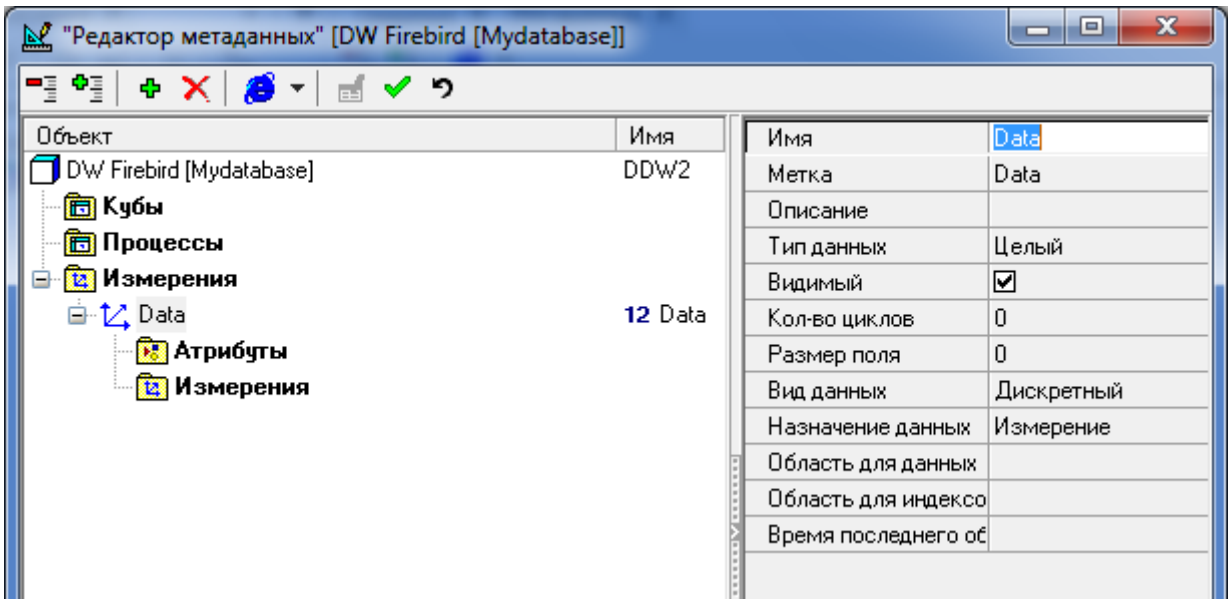
Оно содержит две области. В левой области в виде древовидной иерархической структуры отображаются объекты метаданных – кубы,

процессы и измерения. Начать следует с создания измерений. Для этого нужно воспользоваться кнопкой «Разрешить редактировать» на панели инструментов редактора метаданных После этого станет доступной кнопка для добавления новых элементов в многомерную модель.



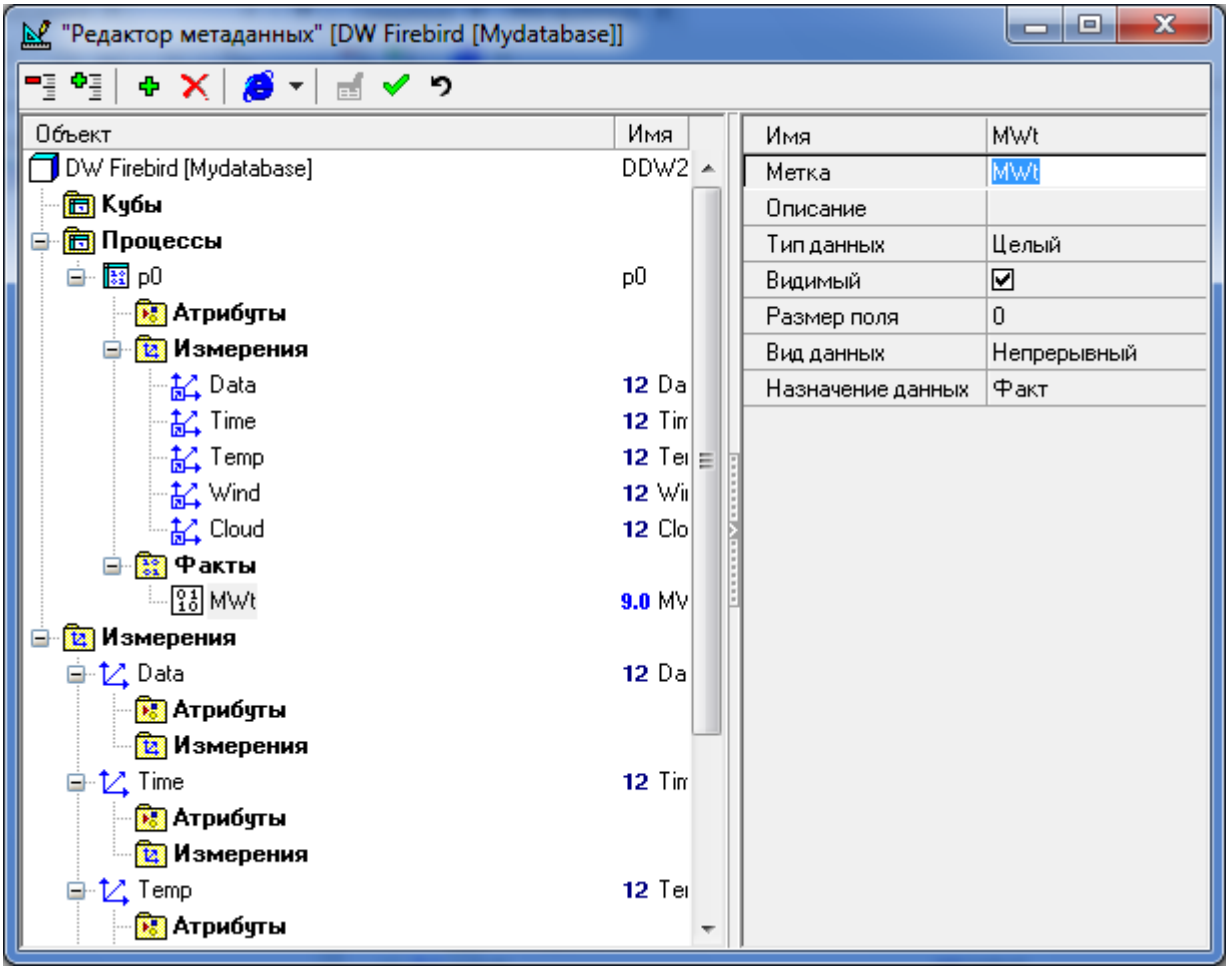
Чтобы создать измерения, выделим пункт «Измерения» и воспользуемся кнопкой «Добавить». Создать первое измерение Data с параметрами:

- имя – Data
- метка – Data
- тип данных – целый.



Проделав аналогичные действия для создания всех остальных измерений, взяв параметры из задания, сохраним изменение структуры ХД с помощью кнопки «Принять изменения».

После того как все измерения и ссылки на измерения созданы, приступаем к формированию процесса. Назовем его p0 и добавим в него ссылки на созданные измерения. Для создания процесса следует выделить пункт «Процесс» и нажать «Добавить». Добавление в процесс измерений и фактов производится аналогично созданию измерений (отличие только в том, что ранее созданные измерения надо просто выбрать из списка, повторно настраивать их параметры нет необходимости).



Таким образом, структура метаданных ХД создана и можно переходить к этапу загрузки данных.

Загрузка данных из внешних источников в хранилище производится в два этапа:

- файлы с данными импортируются в аналитическое приложение;

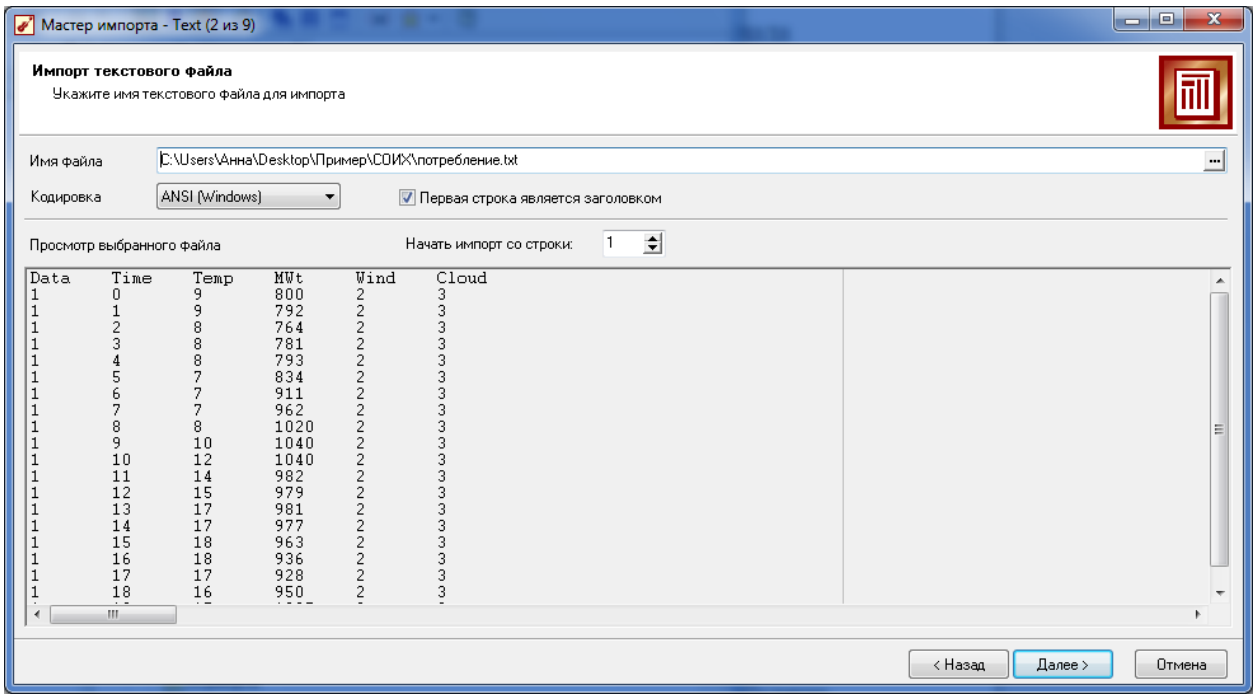
- загрузка данных в процессы и измерения ХД.

Импорт файлов в аналитическое приложение. Для начала этой процедуры следует перейти на панель сценариев с помощью кнопки и произвести импорт в аналитическое приложение файлов с таблицами, которые нужно загрузить в ХД. Для этого предусмотрен «Мастер импорта», который вызывается кнопкой и содержит следующие шаги.

1. Выбор типа файла, в котором содержатся импортируемые таблицы. В версии Deductor Academic доступны для импорта только текстовые файлы с разделителем, в которых столбцы таблицы разделены однотипными символами-разделителями – пробелом, табуляцией, точкой с запятой и т.д.

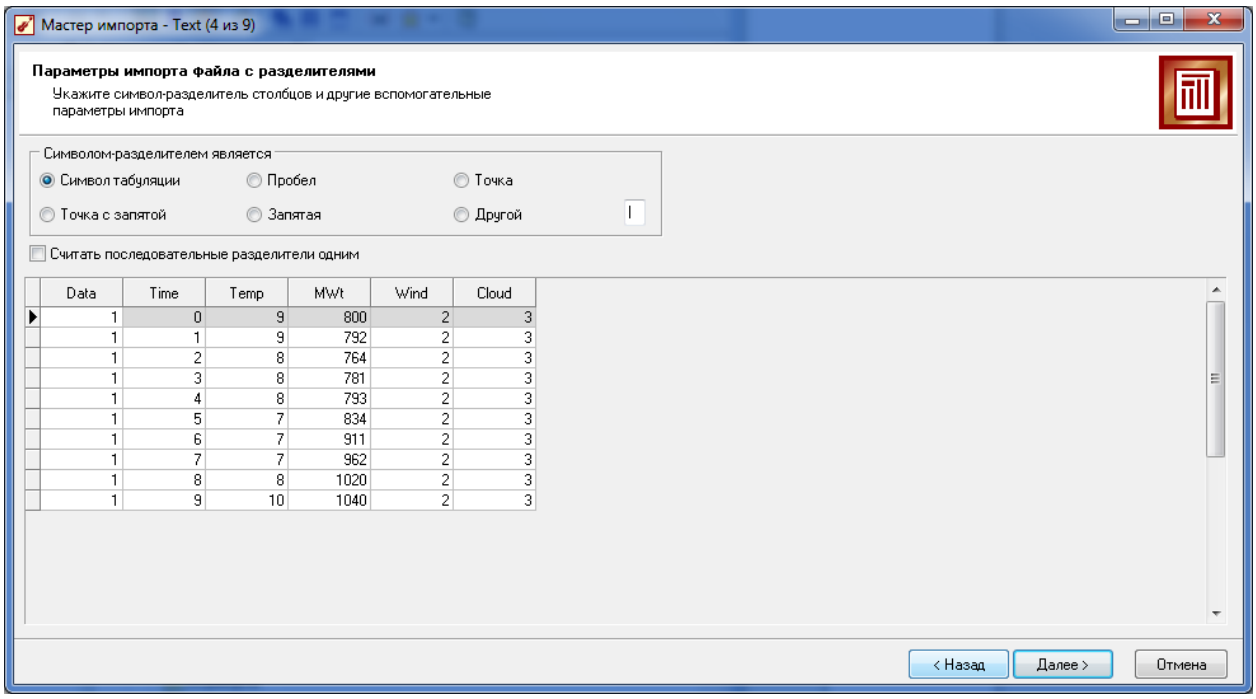
В списке следует выделить строку Text и нажать «Далее».

2. На 2-м шаге требуется задать имя файла и параметры импорта. Настройка параметров импорта источника данных Имя файла и путь можно указать вручную или воспользоваться кнопкой «…», которая вызывает стандартное окно открытия файлов Windows и позволяет выбрать нужный файл. Если файл выбран верно, то в нижней части окна откроется область просмотра, где можно убедиться в правильности параметров импорта. Если первая строка файла содержит заголовки столбцов таблицы, следует установить соответствующий флажок. Когда импорт следует начать не с первой строки, а с какой-то другой, требуется установить нужное значение в поле «Начать импорт со строки». После выбора файла в нижней части окна откроется предварительный просмотр его содержимого.



3. На 3-м шаге настраиваются форматы данных импортируемого файла: с разделителями или фиксированной шириной столбцов, выбирается используемый символ разделитель целой и дробной частей чисел, компонентов даты и компонентов времени (можно узнать из предварительного просмотра на предыдущем шаге), а также выбрать формат даты и времени. Неправильная настройка этих параметров может привести к искажению данных при загрузке и потребовать их перенастройки

4. На 4-ма шаге «Мастера импорта» требуется выбрать используемый символ-разделитель столбцов таблицы. Вид данных в секции предварительного просмотра позволяет определить нужный разделитель: если он выбран правильно, то структура таблицы будет регулярной, в противном случае она может быть искажена



5. На 5-м шаге Мастера импорта требуется определить параметры столбцов импортируемой таблицы Для настройки следует выделить имя столбца в списке слева и задать параметры в правой части окна:

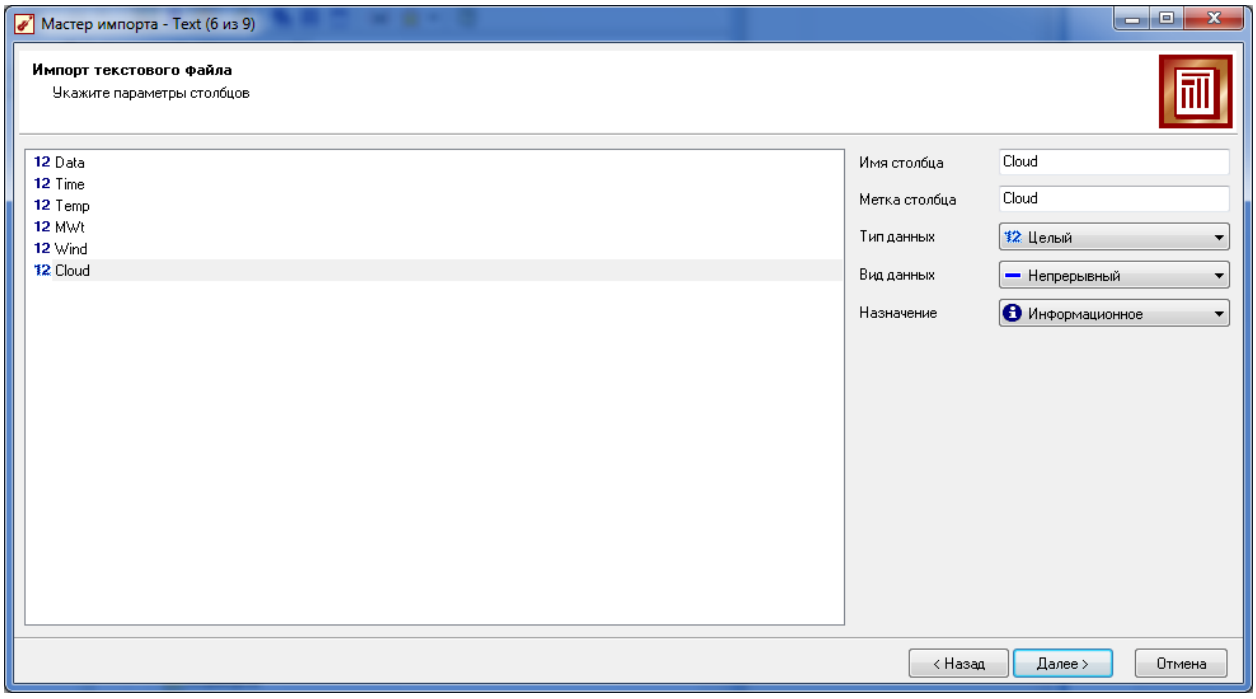
- имя столбца – уникальный идентификатор (только латинские буквы). Если его не задать, система определит имя по умолчанию;

- метка столбца – произвольная метка, отражающая смысл данных в столбце;

- тип данных – тип данных в столбце: логический, дата/время, вещественный, целый и строковый. Выбранный тип обязательно должен соответствовать типу измерения или факта в структуре метаданных, в которые будут загружаться соответствующие столбцы. В противном случае возникнет ошибка;

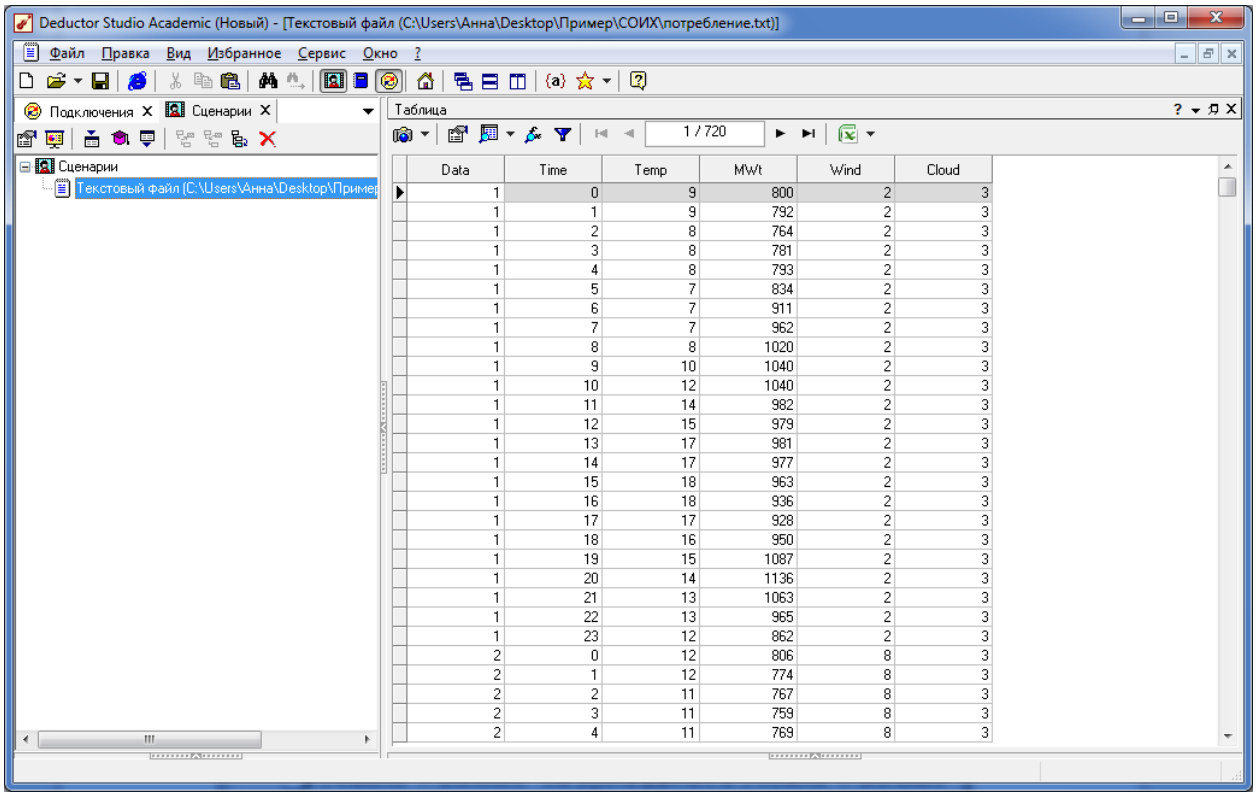
- вид данных – непрерывный или дискретный. Непрерывный вид используется только для столбцов вещественного или целого типов, дискретный – для логического, строкового и целого типов. Выбранный вид данных должен соответствовать виду, определенному для соответствующих измерений и фактов в структуре метаданных ХД;

- назначение – выбирается назначение столбца, как оно было определено в структуре метаданных: измерение, атрибут или факт.



6. На следующем шаге запускается процесс загрузки и контролируется по прогресс-индикатору. Если прогресс-индикатор долгое время не показывает продвижение загрузки, возможно, возникала ошибка (например, разорвалось сетевое соединение). В этом случае следует остановить процесс загрузки, устранить проблему и запустить загрузку повторно. После загрузки 100 % данных выдается сообщение об успешном завершении.

7. Последние два шага «Мастера импорта» в данном случае можно пройти без изменения их настроек. После успешного завершения загрузки в панели «Сценарии» появляется ссылка на загруженный файл, выделение которой открывает в рабочей области окна программы таблицу данных из файла.

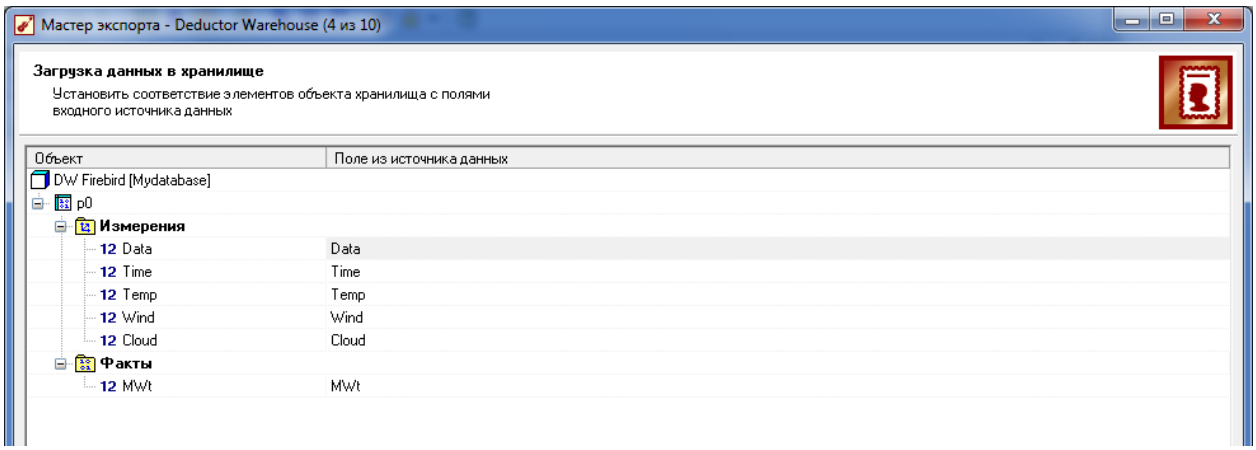


Data	Time	Temp	MWt	Wind	Cloud
1	0	9	800	2	3
1	1	9	792	2	3
1	2	8	764	2	3
1	3	8	781	2	3
1	4	8	793	2	3
1	5	7	834	2	3
1	6	7	911	2	3
1	7	7	962	2	3
1	8	8	1020	2	3
1	9	10	1040	2	3
1	10	12	1040	2	3
1	11	14	982	2	3
1	12	15	979	2	3
1	13	17	981	2	3
1	14	17	977	2	3
1	15	18	963	2	3
1	16	18	936	2	3
1	17	17	928	2	3
1	18	16	950	2	3
1	19	15	1087	2	3
1	20	14	1136	2	3
1	21	13	1063	2	3
1	22	13	965	2	3
1	23	12	862	2	3
2	0	12	806	8	3
2	1	12	774	8	3
2	2	11	767	8	3
2	3	11	759	8	3
2	4	11	769	8	3

Загрузка измерений и фактов процесса в ХД. Для загрузки в ХД следует выбрать в панели сценариев ссылку на файл, столбцы которого следует переместить в структуру метаданных, и нажать копку «Мастер экспорта». Далее следует выполнить следующие шаги.

1. На 1-м шаге Мастера выбрать ХД, в которое будет производиться загрузка

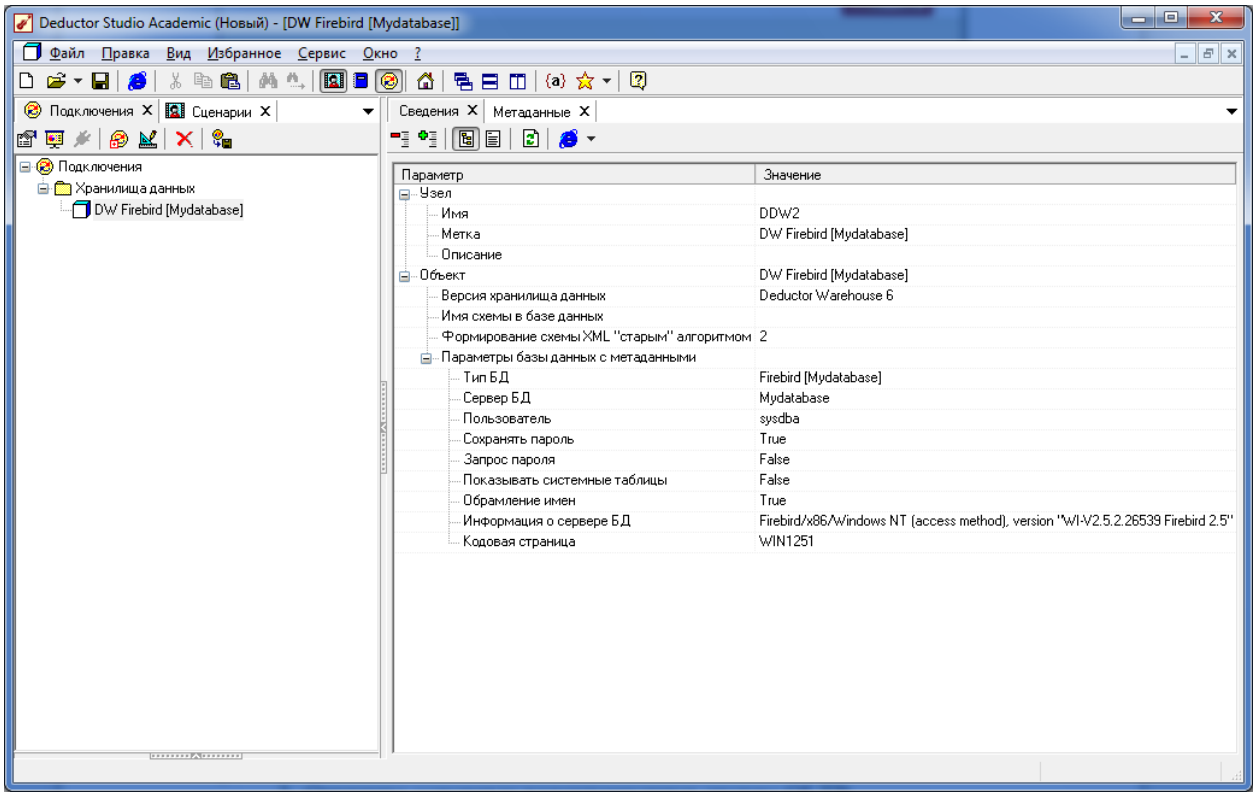
2. На 3-м шаге требуется установить соответствие между столбцами загружаемой таблицы и объектами структуры метаданных в ХД. В этом окне для каждого объекта метаданных должен быть указан столбец из загружаемой таблицы. Если поле не указано, это значит, что параметры объекта метаданных не соответствуют параметрам загрузки столбца в источнике (тип или вид данных, назначение). В этом случае следует перейти в редактор метаданных и произвести в их структуре соответствующие изменения либо изменить настройки в источнике данных.



После того как будет достигнуто полное соответствие между столбцами таблицы и объектами метаданных, можно будет переходить к их загрузке.

3. На 7-м шаге определить способ агрегирования числовых значений (если необходимо)

4. Запустить процесс загрузки в ХД, контролируя его ход с помощью прогресс-индиктора



В основе идеи OLAP лежит многомерная модель данных в виде так называемых гиперкубов или OLAP-кубов. Грани такого куба опираются на оси измерений многомерной модели, а внутри куба расположены факты.

Элементами многомерной модели данных в гиперкубе являются ячейки и срезы. Ячейки содержат отдельные факты бизнес-процессов, представляемых в кубе. Срез куба представляет собой множество ячеек, содержащих факты, связанные с определенным значением измерения. Таким образом, извлекая из куба нужные срезы, можно оперативно получать нужную информацию, в требуемом разрезе и наиболее удобном для восприятия виде. По срезам могут вычисляться агрегированные значения (суммы, количества и т.д.).

Чтобы приступить к построению OLAP-куба, нужно выделить узел источника данных в панели «Сценарии» и с помощью кнопки открыть «Мастер визуализации», где в списке доступных представлений данных отметить пункт «Куб».

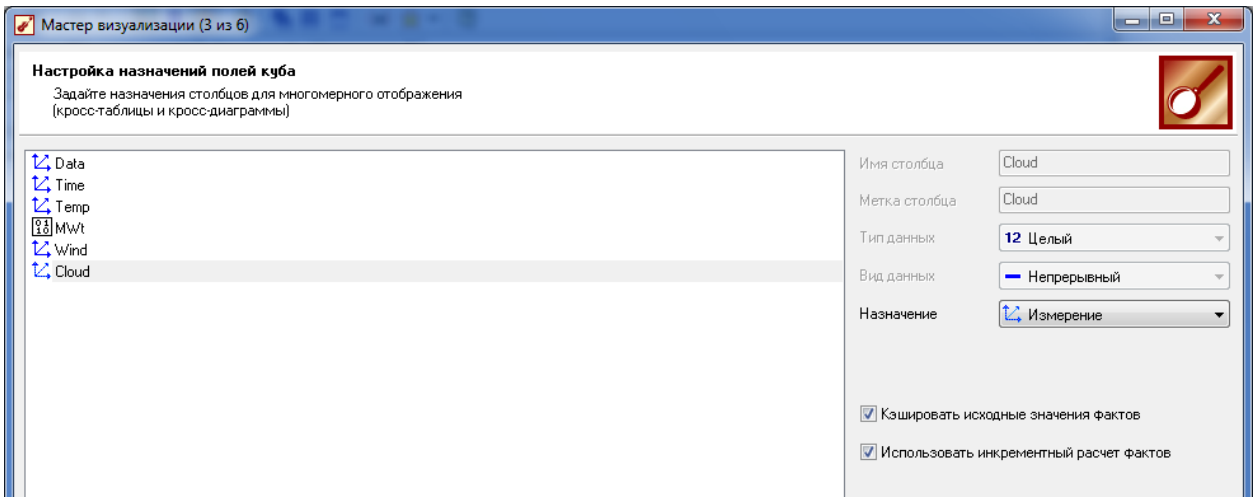
На 3-м шаге «Мастера визуализации» требуется выбрать столбцы источника данных, которые будут использоваться при построении куба, и определить их назначение. Могут быть установлены следующие назначения полей:

- измерение – поле будет использовано в качестве измерения (все поля строкового типа и типа дата/время по умолчанию устанавливаются как измерения);

- факт – поле будет использоваться как факт (все поля числового типа по умолчанию устанавливаются как факты);

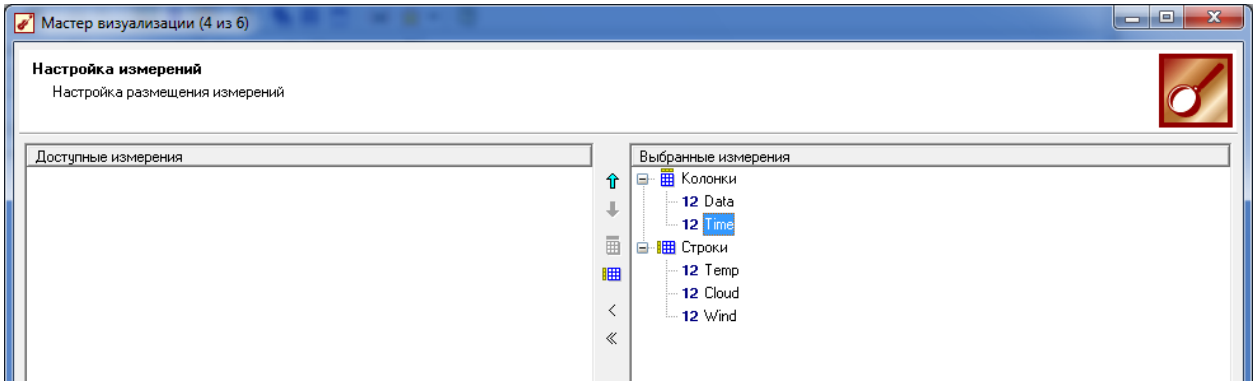
- неиспользуемое – поле не будет использоваться при построении OLAP-куба;

- информационное – поле будет отображаться в итоговых таблицах, но в куб встраиваться не будет.



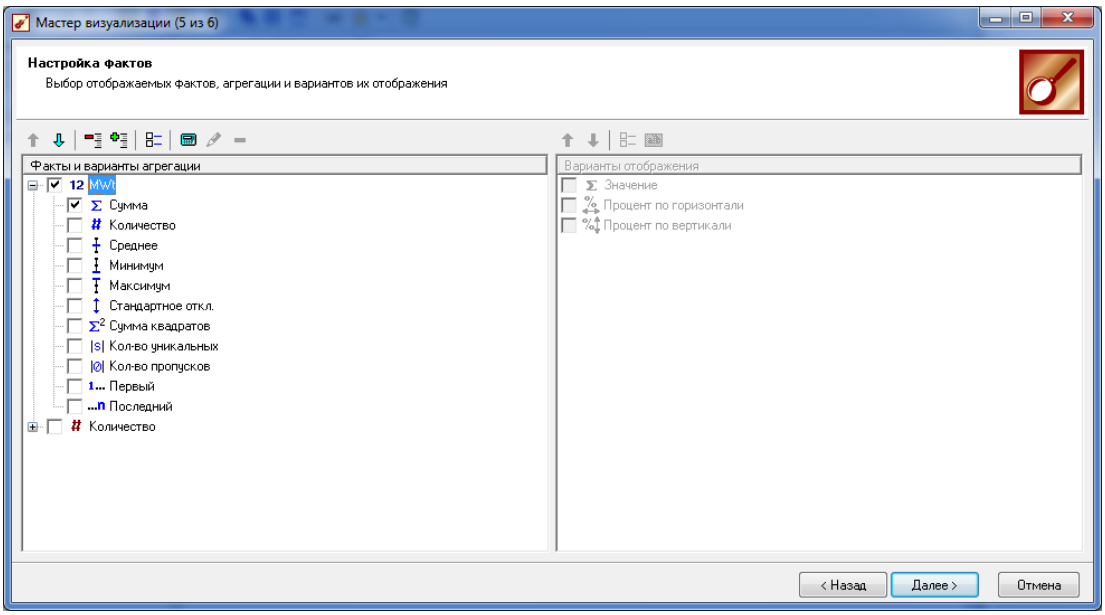
На 4-м шаге нужно выбрать способ представления измерений: в строках или колонках

Чтобы выбрать способ размещения измерения, нужно «перетащить» его значок из секции «Доступные измерения» на соответствующий значок «Колонки» или «Строки» в секции «Выбранные измерения».

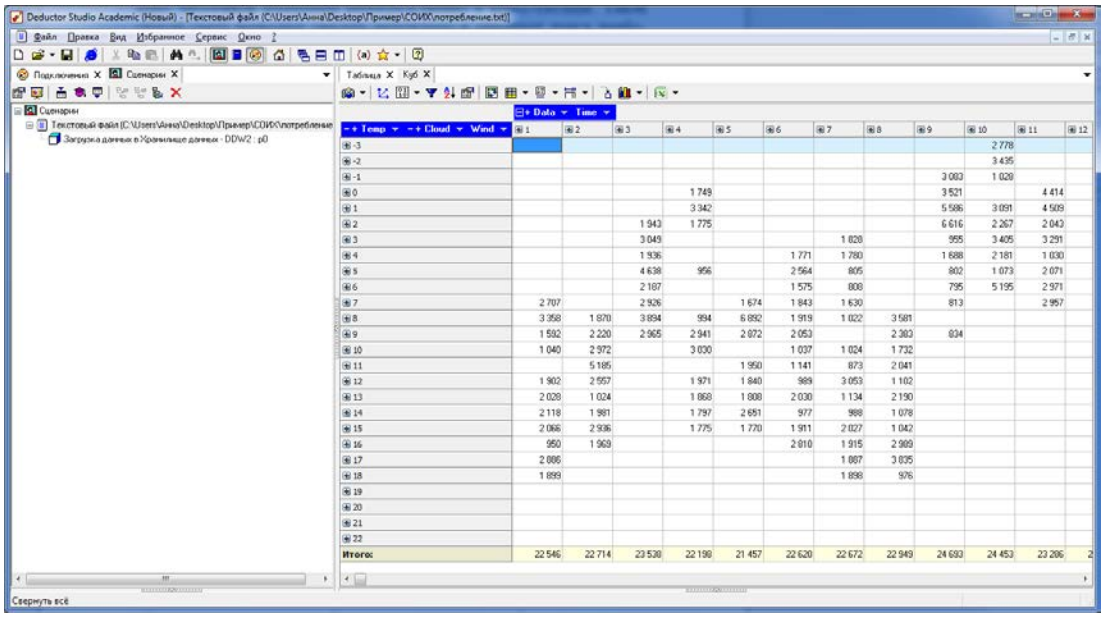


На 5-м шаге требуется выполнить настройку фактов: выбрать, какие факты будут использоваться при построении куба и какие функции агрегации к ним должны быть применены (если это необходимо).

Чтобы факты из списка «Факты и варианты агрегации» использовались в OLAP-кубе, нужно установить флажки слева от имени факта.



После завершения работы Мастера визуализации откроется кросс-таблица. Несложно увидеть, что изначально в кросс-таблице представлены только агрегаты по каждому факту. Значения измерений и фактов скрыты. Логика здесь понятна: система «не знает», в каком именно разрезе данные интересуют аналитика, на какие вопросы он хочет получить ответы с их помощью, поэтому включает в таблицу минимум информации и предоставляет возможность пользователю, манипулируя структурой таблицы, самому выбрать наиболее информативное представление.



Контрольные вопросы

1. Что такое хранилище данных, какие требования к нему предъявляются?

2. Опишите основные характеристики систем OLTP.

3. Каковы причины появления и развития концепции ХД?

4. Перечислите основные элементы логической структуры ХД, поясните их смысл, приведите примеры.

Лабораторная работа 2. Проектирование реляционных баз данных

Теоретические сведения

Процесс проектирования базы данных является итерационным - допускающим возврат к предыдущим этапам для пересмотра ранее принятых решений и включает следующие этапы:

1.Выделение сущностей и связей между ними.

2.Построение диаграмм ER-типа с учетом всех сущностей и их связей.

3.Формирование набора предварительных отношений с указанием пред полагаемого первичного ключа для каждого отношения и использованием диаграмм ER-типа.

4.Добавление неключевых атрибутов в отношения.

5.Приведение предварительных отношений к нормальной форме Бойса Кодда, например, с помощью метода нормальных форм.

6.Пересмотр ER-диаграмм в следующих случаях:

- некоторые отношения не приводятся к нормальной форме Бойса - Кодда,
- некоторым атрибутам не находится логически обоснованных мест в предварительных отношениях.

После преобразования ER-диаграмм осуществляется повторное выполнение предыдущих этапов проектирования (возврат к этапу 1).

На **первом этапе** выполняется выделение сущностей и связей между ними.

Одним из узловых этапов проектирования является этап формирования отношений. Рассмотрим процесс формирования предварительных отношений, составляющих первичный вариант схемы БД.

В рассмотренных выше примерах связь ВЕДЕТ всегда соединяет две сущности и поэтому является *бинарной.* Сформулированные ниже правила формирования отношений из диаграмм ER-типа распространяются именно на

бинарные связи. Поэтому, когда речь идет о связях, слово «бинарные» далее опускается.

На **<u>втором этапе</u>** выполняется построение диаграмм ER-типа с учетом всех сущностей и их связей.

Правила формирования отношений

Основными понятиями метода сущность-связь являются следующие:

- сущность
- атрибут сущности,
- ключ сущности,
- связь между сущностями,
- степень связи,
- класс принадлежности экземпляров сущности,
- диаграммы ER-экземпляров,
- диаграммы ER-типа.

Сущность представляет собой объект, информация о котором хранится в БД. Экземпляры сущности отличаются друг от друга и однозначно идентифицируются.

Атрибут представляет собой свойство сущности. Это понятие аналогично понятию атрибута в отношении. Так, атрибутами сущности ПРЕПОДАВАТЕЛЬ может быть его Фамилия, Должность, Стаж (преподавательский) и т. д.

Ключ сущности - атрибут или набор атрибутов, используемый для идентификации экземпляра сущности. Как видно из определения, понятие ключа сущности аналогично понятию ключа отношения.

Связь двух или более сущностей - предполагает зависимость между атрибутами и этих сущностей. Название связи обычно представляется *глаголом.* Примерами связей между сущностями являются следующие: ПРЕПОДАВАТЕЛЬ *ВЕДЕТ* ДИСЦИПЛИНУ (Иванов ВЕДЕТ «Базы данных»), ПРЕПОДАВАТЕЛЬ *ПРЕПОДАЕТ-В* ГРУППЕ (Иванов

ПРЕПОДАЕТ-В 256 группе), ПРЕПОДАВАЛТЕЛЬ *РАБОТАЕТ-НА* КАФЕДРЕ (Иванов РАБОТАЕТ-НА25 кафедре).

Приведенные определения сущности и связи не полностью формализованы, но приемлемы для практики. Следует иметь в виду, что в результате проектирования могут быть получены несколько вариантов одной БД. Так, два разных проектировщика, рассматривая одну и ту же проблему с разных точек зрения, могут получить различные наборы сущностей и связей. При этом оба варианта могут быть рабочими, а выбор лучшею из них будет результатом личных предпочтений.

С целью повышения наглядности и удобства проектирования для представления сущностей, экземпляров сущностей и связей между ними используются следующие графические средства:

• диаграммы ER-экземпляров,

• диаграммы ER-типа, или ER-диаграммы.

На рис. 1 приведена диаграмма ER-экземпляров для сущностей ПРЕПОДАВАТЕЛЬ и ДИСЦИПЛИНА со связью *ВЕДЕТ*.

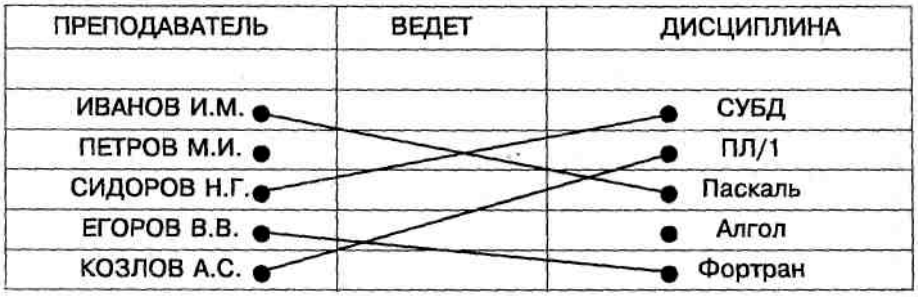


Рисунок 1 – Диаграмма ER-экземпляров

На **третьем этапе** выполняется Формирование набора предварительных отношений с указанием пред полагаемого первичного ключа для каждого отношения и использованием диаграмм ER-типа.

Диаграмма ER-экземпляров показывает, какую конкретно дисциплину (СУБД, ПЛ/1 и т.д.) ведет каждый из преподавателей. На рис. 2 представлена

диаграмма ER-типа, соответствующая рассмотренной диаграмме ER-экземпляров.

Рисунок 2 – Диаграмма ER-типа

На начальном этапе проектирования БД выделяются атрибуты, составляющие ключи сущностей.

На основе анализа диаграмм ER-типа формируются отношения проектируемой БД. При этом учитывается *степень связи сущностей* и *класс их принадлежности,* которые, в свою очередь, определяются на основе анализа диаграмм ER-экземпляров соответствующих сущностей.

Степень связи является характеристикой связи между сущностями, которая может быть типа: 1:1, 1:M,M:1,M:M.

Класс принадлежности (KII) сущности может быть: *обязательным* и *необязательным.*

Класс принадлежности сущности является *обязательным,,* если все экземпляры этой сущности обязательно участвуют в рассматриваемой связи, в противном случае класс принадлежности сущности является *необязательным.*

Варьируя классом принадлежности сущностей для каждого из названных типов связи, можно получить несколько вариантов диаграмм ER-типа. Рассмотрим примеры некоторых из них.

Пример 1. Связи типа 1:1 и необязательный класс принадлежности.

В приведенной на рис. 11 диаграмме степень связи между сущностями 1:1, класс принадлежности обеих сущностей необязательный. Действительно, из рисунка видно следующее:

•каждый преподаватель ведет не более одной дисциплины, а каждая дисциплина ведется не более чем одним преподавателем (степень связи 1:1);

•некоторые преподаватели не ведут ни одной дисциплины и имеются дисциплины, которые не ведет ни один из преподавателей (класс принадлежности обеих сущностей необязательный).

Пример 2. Связи типа 1:1 и обязательный класс принадлежности.

На рис. 3 приведены диаграммы, у которых степень связи между сущностями 1:1, а класс принадлежности обеих сущностей обязательный.

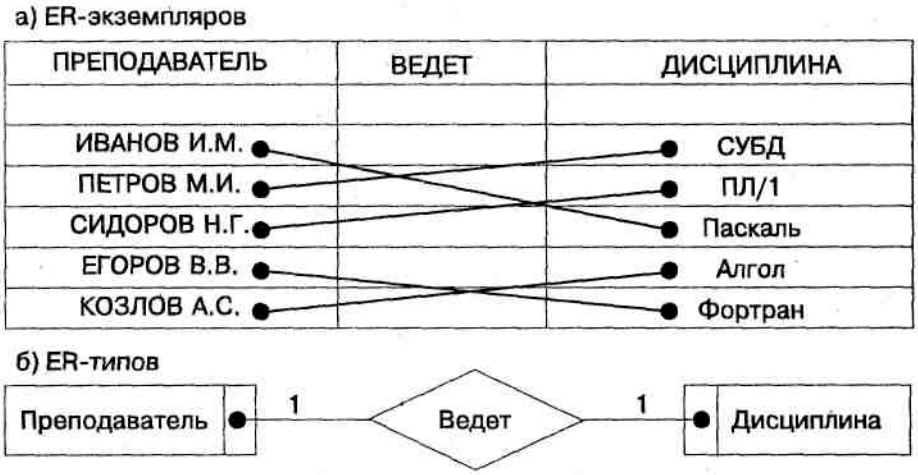


Рис. 3. Диаграммы для связи 1:1 и обязательным КП обеих сущностей

В этом случае каждый преподаватель ведет одну дисциплину и каждая дисциплина ведется одним преподавателем.

Возможны два промежуточных варианта с необязательным классом принадлежности одной из сущностей.

Под каждым блоком, соответствующим некоторой сущности, указывается ее ключ, выделяемый подчеркиванием. Многоточие за ключевыми атрибутами означает, что возможны другие атрибуты сущности, но ни один из них не может быть частью ее ключа. Эти атрибуты выявляются после формирования отношений.

На практике степень связи и класс принадлежности сущностей при проектировании БД определяется спецификой предметной области. Рассмотрим примеры вариантов со степенью связи 1:M или M:1.

Пример 3. Связи типа 1.M.

Каждый преподаватель может вести несколько дисциплин, но каждая дисциплина ведется одним преподавателем.

Пример 4. Связи типа M:1.

Каждый преподаватель может вести одну дисциплину, но каждую дисциплину могут вести несколько преподавателей.

Примеры с типом связи 1:M или M:1 могут иметь ряд вариантов, отличающихся классом принадлежности одной или обеих сущностей. Обозначим обязательный класс принадлежности символом «О», а необязательный - символом «Н», тогда варианты для связи типа 1:M условно можно представить как: О-О, О-Н, Н-О, Н-Н. Для связи типа M.i также имеются 4 аналогичных варианта.

Пример 5. Связи типа 1:M вариант Н-О.

Каждый преподаватель может вести несколько дисциплин или ни одной, но каждая дисциплина ведется одним преподавателем (рис. 4).

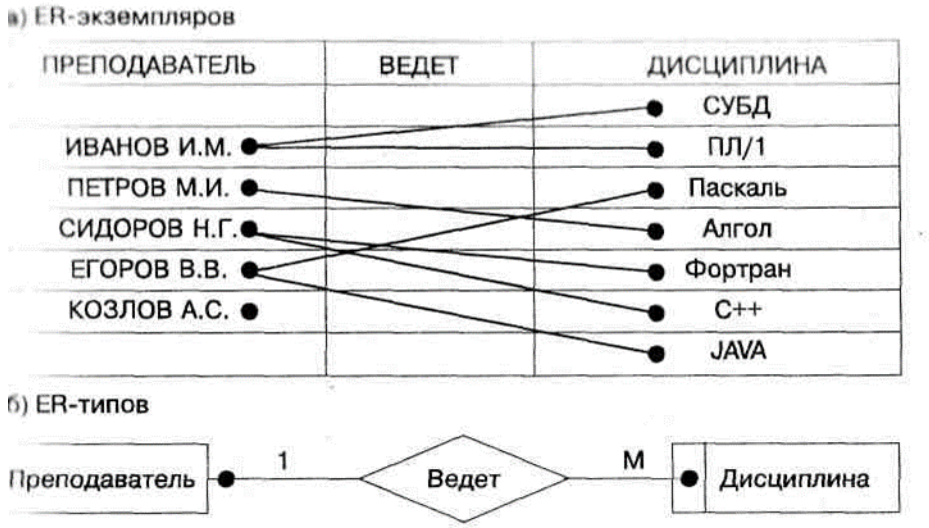


Рисунок 4 - Диаграммы для связи типа 1:M варианта Н-О

По аналогии легко составить диаграммы и для остальных вариантов.

Пример 6. Связи типа M:M.

Каждый преподаватель может вести несколько дисциплин, а каждая дисциплина может вестись несколькими преподавателями.

Как и в случае других типов связей, для связи типа M:M возможны 4 варианта, отличающиеся классом принадлежности сущностей.

Пример 7. Связи типа M:M и вариант класса принадлежности O-H.

Допустим, что каждый преподаватель ведет не менее одной дисциплины, а дисциплина может вестись более чем одним преподавателем, есть и такие дисциплины, которые никто не ведет. Соответствующие этому случаю диаграммы приведены на рис.5.

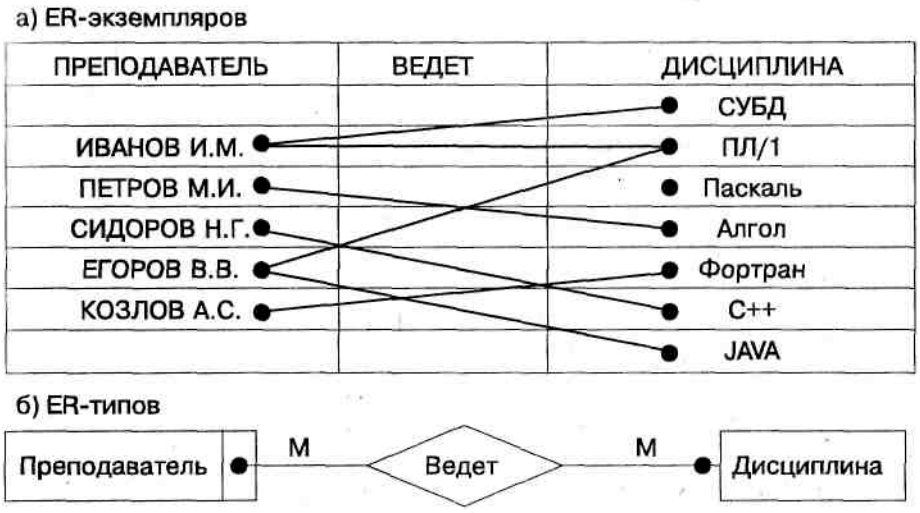


Рисунок 5 - Диаграммы для связи типа M : M и варианта O-H

На **четвертом** этапе проектирования необходимо сформулировать исходные отношения (или одно отношение), в которые войдут все необходимые для хранения данные (т.е. добавить неключевые описательные атрибуты).

На **пятом** этапе необходимо выполнить нормализацию отношений. Нормализация осуществляется постепенно через приведение к первой, второй и третьей нормальным формам.

Отношение находится в первой нормальной форме (сокращённо 1НФ), если все его атрибуты атомарны, то есть если ни один из его атрибутов нельзя разделить на более простые атрибуты, которые соответствуют каким-то другим свойствам описываемой сущности.

Отношение находится во второй нормальной форме (сокращённо 2НФ) тогда и только тогда, когда оно находится в первой нормальной форме и каждый его неключевой атрибут неприводимо зависим от первичного ключа.

Отношение находится в третьей нормальной формк, если оно находится во второй нормальной форме и в нем нет транзитивных зависимостей атрибутов от возможных ключей, т.е. каждый непервичный атрибут нетранзитивно зависит от каждого возможного ключа отношения.

На **шестом** этапе необходимо построить реляционную модель (ER-модель) базы данных. Для этого необходимо выполнить следующее:

1) Определить сущности модели на основании выделенных отношений. Сущность - это класс однотипных объектов, информация о которых должна быть учтена в модели.

2) Определить атрибуты сущностей. Атрибут сущности - это именованная характеристика, являющаяся свойством сущности.

3) Определить типы связей между отношениями. Связь - это некоторая ассоциация между двумя сущностями. Одна сущность может быть связана с другой сущностью или сама с собою. Связи позволяют по одной сущности находить другие сущности, связанные с нею. Каждая связь может иметь один из следующих типов связи:

-Связь типа один-к-одному означает, что один экземпляр первой сущности (левой) связан с одним экземпляром второй сущности (правой). Связь один-к-одному чаще всего свидетельствует о том, что на самом деле мы имеем всего одну сущность, неправильно разделенную на две.

- Связь типа один-ко-многим означает, что один экземпляр первой сущности (левой) связан с несколькими экземплярами второй сущности (правой). Это наиболее часто используемый тип связи. Левая сущность (со стороны "один") называется родительской, правая (со стороны "много") - дочерней.

- Связь типа много-ко-многим означает, что каждый экземпляр первой сущности может быть связан с несколькими экземплярами второй сущности,

и каждый экземпляр второй сущности может быть связан с несколькими экземплярами первой сущности. Тип связи много-ко-многим является временным типом связи, допустимым на ранних этапах разработки модели. В дальнейшем этот тип связи должен быть заменен двумя связями типа один-ко-многим путем создания промежуточной сущности.

4) Определить ключевые атрибуты и атрибуты, по которым осуществляется связь между сущностями. Ключ сущности - это неизбыточный набор атрибутов, значения которых в совокупности являются уникальными для каждого экземпляра сущности. Неизбыточность заключается в том, что удаление любого атрибута из ключа нарушается его уникальность.

Задание для выполнения

1. С помощью онлайн-сервиса draw.io или программы MS Visio создать диаграмму сущность-связь для предметной области «Телефонная книга».

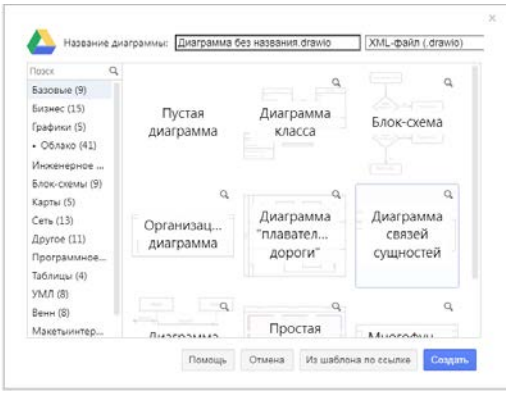
1.1 Определить сущности: Контактное лицо, Телефонный номер.

1.2 Определить связь между сущностями:

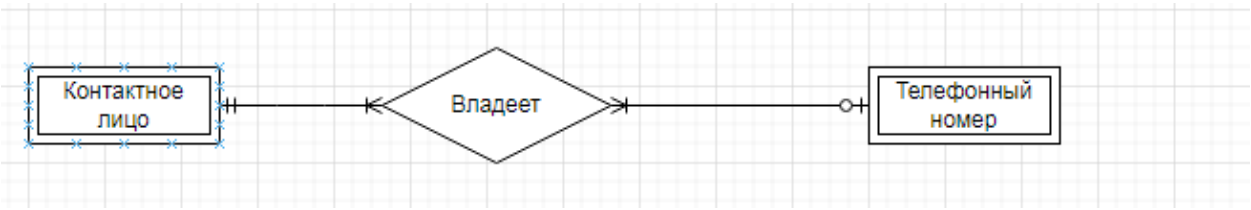
- Тип связи: 1:M (одно контактное лицо может иметь несколько телефонных номеров, у одного телефонного номера только один владелец;

- Вариант связи: H-O (у контактного лица может не быть телефона, но не может быть телефона без владельца).

1.3 При работе в draw.io выберите шаблон «сущность-связь»:



Постройте диаграмму следующего вида:



2. Получить у преподавателя в качестве индивидуального задания предметную область для проектирования

2.1 Определить сущности для предметной области (не менее 10).

2.2 Определить и описать связи между сущностями

2.3 Составить множественные ER-диаграммы и единую ER-диаграмму, связывающую все сущности

2.4 Определить первичные ключи для каждой сущности

3. Дополнить сущности неключевыми описательными атрибутами

3.1 Для каждой сущности выполнить нормализацию

3.2 Пересмотреть ER-модель БД

4. Построить ER-модель в программе VISIO или с помощью онлайн-сервиса draw.io

5. Выполнить описание таблиц БД с указанием ключей (первичных, внешних) и типов данных полей. Для определения типов данных полей использовать типы данных MySQL.

Указания по технике безопасности

Техника безопасности при выполнении лабораторной работы совпадает

с общепринятой для пользователей персональных компьютеров: самостоятельно не производить ремонт персонального компьютера, установку и удаление программного обеспечения; в случае неисправности персонального компьютера сообщить об этом обслуживающему персоналу лаборатории (оператору, администратору); соблюдать правила техники безопасности при работе с электрооборудованием; не касаться электрических розеток металлическими предметами; рабочее место пользователя персонального компьютера должно содержаться в чистоте; не разрешается возле персонального компьютера принимать пищу, напитки.

Защита работы

Перед выполнением лабораторной работы каждый студент получает индивидуальное задание. Защита лабораторной работы происходит только после выполнения индивидуального задания и оформления отчета по работе. При защите лабораторной работы студент отвечает на контрольные вопросы, приведенные в конце, и поясняет выполненное индивидуальное задание. Ход защиты лабораторной работы контролируется преподавателем.

Требования к отчету

Отчет должен содержать:

1. Скриншоты выполнения задания 1.
2. Описание выполнения задания 2 в соответствии с примером

Пример выполнения задания

Предметная область: необходимо создать базу данных для учета оборудования учебного заведения.

С помощью проектируемой базы данных решается учетная задача по учету оборудования.

Для ведения учета оборудования учебного заведения необходимо иметь следующую информацию (таблица 1):

Таблица 1 – Хранимая информация

Информация	Описание
Инвентарный номер	Инвентарный номер оборудования
Наименование	Наименование оборудования
Вид оборудования	Вид оборудования
Описание оборудования	Описание (краткое)
Год приобретения	Год приобретения оборудования
Состояние	Состояние на текущий момент
Стоимость оборудования	Стоимость на момент заполнения
Помещение	Помещение, в котором установлено оборудование
Тип помещения	Тип помещения (лабораторное, лекционное, служебное)
МОЛ	Материально ответственное лицо: сотрудник, ответственный за помещение (ФИО, должность)
Кафедра	Кафедра, которой принадлежит помещение
Заведующий кафедрой	Заведующий кафедрой (ФИО, должность)

Можно выделить следующие сущности:

1) Оборудование;
2) Сотрудник;
3) Помещения.

Каждый вид информации описывается отношением. Отношение представляет собой таблицу, поля которой являются атрибутами отношения, а строки – записями.

Отношения связаны между собой определенными зависимостями.

1. Отношение «Оборудование» связано с отношением «Помещение» следующим образом (рис. 1):

Рисунок 1 – Связь между отношениями «Оборудование» и «Помещение»

2. Отношение «Оборудование» связано с отношением «Сотрудники» следующим образом (рис. 2)

Рисунок 2 – Связь между отношениями «Сотрудник» и «Помещение»

ER-диаграмма системы:

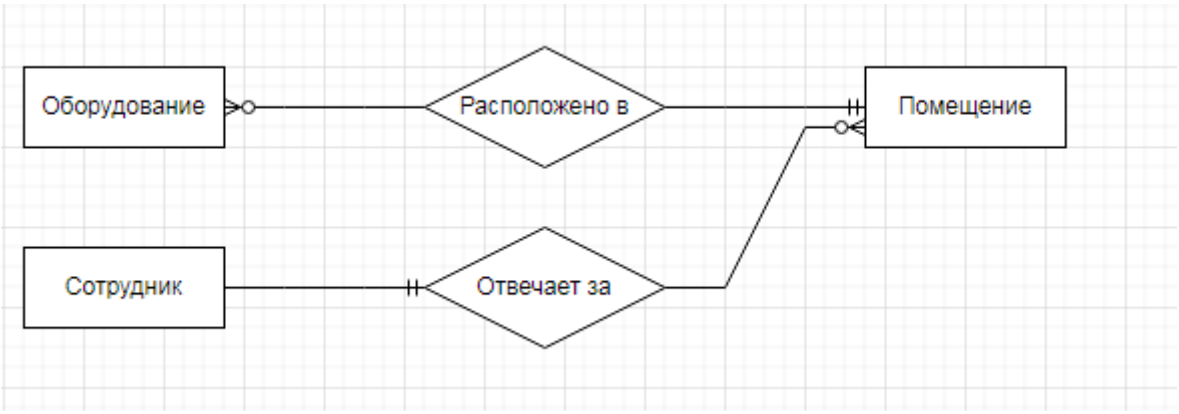


Рисунок 3 – ER-диаграмма системы

Таким образом, получаем следующие отношения (таблицы 2-4).

Таблица 2 – Отношение «Оборудование»

Наименование атрибута	Описание
Инвентарный номер	Инвентарный номер оборудования
Наименование	Наименование оборудования
Вид оборудования	Вид оборудования
Описание оборудования	Описание (краткое)
Год приобретения	Год приобретения оборудования
Состояние	Состояние на текущий момент
Стоимость оборудования	Стоимость на момент заполнения
Помещение	Помещение, в котором установлено оборудование

Таблица 3 – Отношение «Помещения»

Наименование атрибута	Тип данных

Номер	Номер помещения
Название	Название помещения
Тип помещения	Тип помещения (лабораторное, лекционное, служебное)
МОЛ	Код МОЛ
Кафедра	Кафедра, ответственная за помещение
Заведующий	Заведующий кафедрой

Таблица 4 – Отношение «Сотрудники»

Наименование атрибута	Описание
Код	Код ответственного лица
ФИО	ФИО
Должность	Должность

Отношение, сформулированное в таблице 4, не находится в первой нормальной форме, поскольку не все атрибуты отношения являются атомарными. Атрибут «МОЛ» содержат не одно, а несколько самостоятельных значений (фамилию, имя, отчество). Поэтому для приведения в первую нормальную форму исходное отношение должно быть преобразовано (таблица 5).

Таблица 5 – Отношение «Сотрудники» в 1НФ

Наименование атрибута	Описание
Код	Код ответственного лица
Фамилия	Фамилия
Имя	Имя
Отчество	Отчество
Должность	Должность

Отношение «Помещение» не находится в 3НФ, так как заведующий кафедрой не зависит от кода помещения, а зависит от кафедры. Поэтому отношение «Помещение» необходимо разбить на 2 отношения:

Таблица 8 – Отношение «Помещение» в 3НФ

Наименование атрибута	Описание
Код	Код помещения
Название	Название помещения
Тип помещения	Тип помещения (лабораторное, лекционное, служебное)
МОЛ	Код МОЛ
Кафедра	Кафедра, которой принадлежит помещение

Таблица 7 – Отношение «Кафедры» в 3НФ

Наименование атрибута	Описание
Код	Код кафедры
Название	Название кафедры
Заведующий	Код заведующего

Для каждого отношения определен ключ «Код», однозначно идентифицирующий каждый экземпляр отношения.

Для удобства ввода и обработки данных необходимо вынести в отдельный список виды оборудования. Поскольку определить состав списка заранее невозможно, его можно выделить в отдельную сущность «Виды оборудования».

Таким образом, получаем следующие сущности (таблицы 8-12).

Таблица 8 – Сущность «Оборудование»

Наименование атрибута	Тип данных
Инвентарный номер	Числовой
Наименование	Текстовый
Вид оборудования	Числовой
Описание оборудования	Текстовый
Год приобретения	Числовой
Состояние	Текстовый
Стоимость оборудования	Числовой
Помещение	Числовой

Таблица 9 – Сущность «Помещения»

Наименование атрибута	Тип данных
Номер	Числовой
Название	Текстовый
Тип помещения	Текстовый
МОЛ	Числовой
Кафедра	Числовой

Таблица 10 – Сущность «Кафедры»

Наименование атрибута	Тип данных
Код	Числовой
Название	Текстовый
Заведующий	Числовой

Таблица 11 – Сущность «Сотрудники»

Наименование атрибута	Тип данных
Код	Числовой
Фамилия МОЛ	Текстовый
Имя МОЛ	Текстовый
Отчество МОЛ	Текстовый
Должность МОЛ	Текстовый

Таблица 12 – Сущность «Виды оборудования»

Наименование атрибута	Тип данных
Код	Числовой
Наименование	Текстовый

Сущности «Помещения» и «Оборудование» связаны связью 1:M, то есть в одном помещении может содержаться несколько элементов оборудования. Связь осуществляется по атрибутам «Номер» и «Помещение».

Сущности «Кафедры» и «Помещения» связаны связью 1:M, то есть одной кафедре может принадлежать несколько помещений. Связь осуществляется по атрибутам «Код» и «Кафедра».

Сущности «Сотрудники» и «Помещения» связаны связью 1:M, то есть один сотрудник может быть ответственным за несколько помещений. Связь осуществляется по атрибутам «Код» и «МОЛ».

Сущности «Виды оборудования» и «Оборудование» связаны связью 1:M, то есть один вид может быть у нескольких единиц оборудования. Связь осуществляется по атрибутам «Код» и «Вид».

ER-модель данных базы данных представлена на рисунке 4.

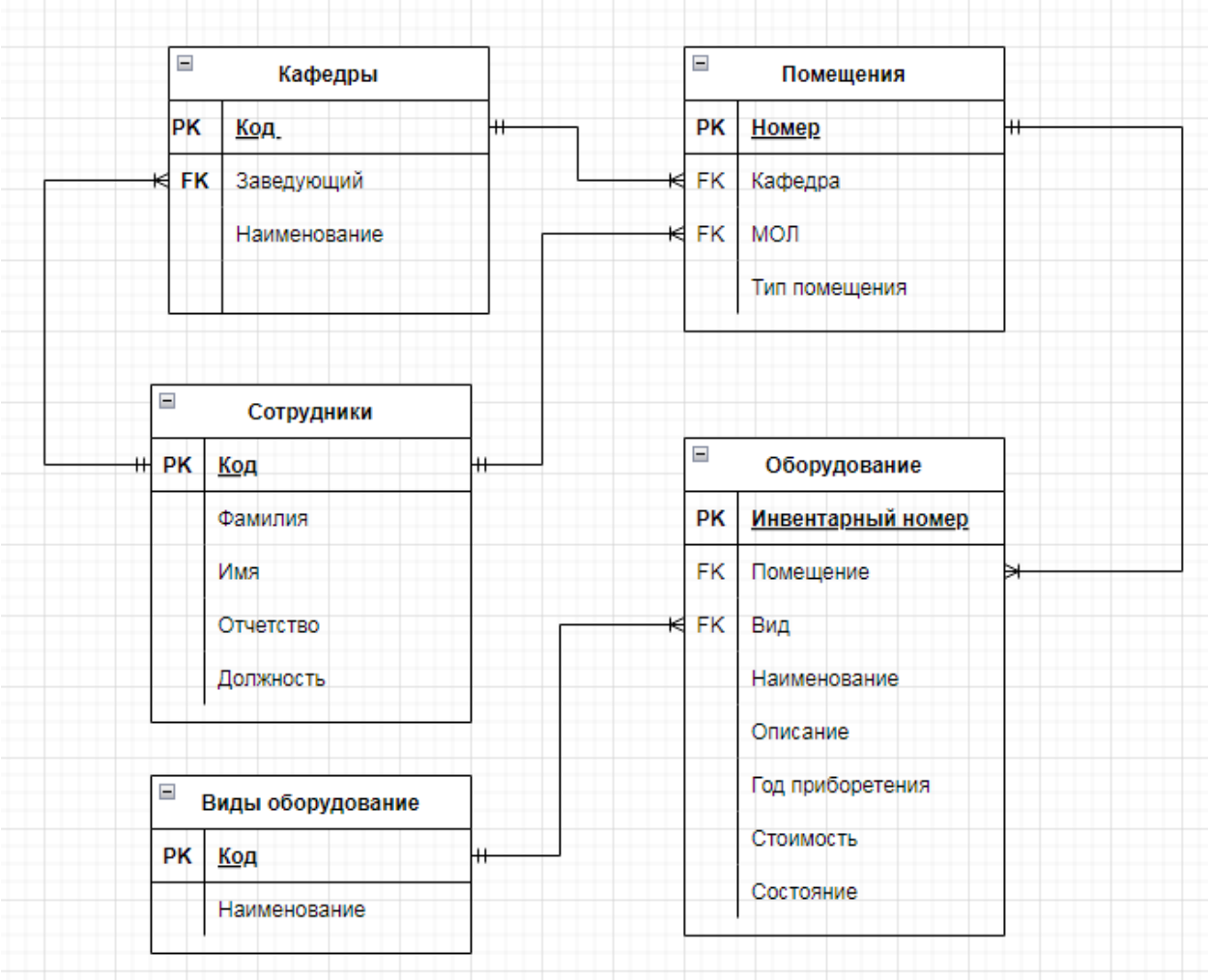


Рисунок 4 – ER-модель базы данных

Таким образом, в базе данных определены 5 взаимосвязанных таблиц, связи между таблицами – один ко многим, для каждой таблицы определены ключевые поля, таблицы нормализованы.

Контрольные вопросы

1. Что представляет собой ER-диаграмма?
2. Что такое атрибут сущности?

3. Что такое ключ сущности? Для чего предназначены ключи?
4. Какие виды ключей вы знаете?

Лабораторная работа 3. Использование языка SQL для создания и обработки баз данных

Теоретические сведения

Создание таблиц

Для создания таблиц применяется команда **CREATE TABLE**. С этой командой можно использовать ряд операторов, которые определяют столбцы таблицы и их атрибуты. И кроме того, можно использовать ряд операторов, которые определяют свойства таблицы в целом. Одна база данных может содержать до 2 миллиардов таблиц.

```
CREATE TABLE название_таблицы
(название_столбца1 тип_данных атрибуты_столбца1,
 название_столбца2 тип_данных атрибуты_столбца2,
 ..............................................
 название_столбцаN тип_данных атрибуты_столбцаN,
 атрибуты_таблицы
)
```

После команды CREATE TABLE идет название создаваемой таблицы. Имя таблицы выполняет роль ее идентификатора в базе данных, поэтому оно должно быть уникальным. Имя должно иметь длину не больше 128 символов. Имя может состоять из алфавитно-цифровых символов, а также символов $ и знака подчеркивания. Причем первым символом должна быть буква или знак подчеркивания.

Имя объекта не может включать пробелы и не может представлять одно из ключевых слов языка SQL. Если идентификатор все же содержит пробельные символы, то его следует заключать в кавычки. Если необходимо в качестве имени использовать ключевые слова, то эти слова помещаются в квадратные скобки.

После имени таблицы в скобках указываются параметры всех столбцов и в самом конце атрибуты, которые относятся ко всей таблице. Атрибуты столбцов и атрибуты таблицы являются необязательными компонентами, и их можно не указывать.

В самом просто виде команда CREATE TABLE должна содержать как минимум имя таблицы, имена и типы столбцов. Возможные типы данных приведены в **Приложении 1.**

Таблица может содержать от 1 до 1024 столбцов. Каждый столбец должен иметь уникальное в рамках текущей таблицы имя, и ему должен быть назначен тип данных.

С помощью выражения **PRIMARY KEY** столбец можно сделать первичным ключом.

```
CREATE TABLE Customers
(
    Id INT PRIMARY KEY,
    Age INT,
    FirstName NVARCHAR(20),
    LastName NVARCHAR(20),
    Email VARCHAR(30),
    Phone VARCHAR(20)
)
```

Если мы хотим, чтобы столбец имел только уникальные значения, то для него можно определить атрибут **UNIQUE.**

```
CREATE TABLE Customers
(
    Id INT PRIMARY KEY IDENTITY,
    Age INT,
    FirstName NVARCHAR(20),
    LastName NVARCHAR(20),
    Email VARCHAR(30) UNIQUE,
    Phone VARCHAR(20) UNIQUE
)
```

Чтобы указать, может ли столбец принимать значение NULL, при определении столбца ему можно задать атрибут **NULL** или **NOT NULL.**

Если этот атрибут явным образом не будет использован, то по умолчанию столбец будет допускать значение NULL. Исключением является тот случай, когда столбец выступает в роли первичного ключа - в этом случае по умолчанию столбец имеет значение NOT NULL.

```
CREATE TABLE Customers
(
    Id INT PRIMARY KEY IDENTITY,
    Age INT,
    FirstName NVARCHAR(20) NOT NULL,
    LastName NVARCHAR(20) NOT NULL,
    Email VARCHAR(30) UNIQUE,
    Phone VARCHAR(20) UNIQUE
)
```

Внешние ключи применяются для установки связи между таблицами. Внешний ключ устанавливается для столбцов из зависимой, подчиненной таблицы, и указывает на один из столбцов из главной таблицы. Хотя, как правило, внешний ключ указывает на первичный ключ из связанной главной таблицы, но это необязательно должно быть непременным условием. Внешний ключ также может указывать на какой-то другой столбец, который имеет уникальное значение.

Общий синтаксис установки внешнего ключа на уровне столбца:

```
[FOREIGN KEY] REFERENCES главная_таблица (столбец_главной_таблицы)
    [ON DELETE {CASCADE|NO ACTION}]
    [ON UPDATE {CASCADE|NO ACTION}]
```

ON DELETE и ON UPDATE

С помощью выражений **ON DELETE** и **ON UPDATE** можно установить действия, которые выполняться соответственно при удалении и изменении связанной строки из главной таблицы. И для определения действия мы можем использовать следующие опции:

- **CASCADE**: автоматически удаляет или изменяет строки из зависимой таблицы при удалении или изменении связанных строк в главной таблице.

- **NO ACTION**: предотвращает какие-либо действия в зависимой таблице при удалении или изменении связанных строк в главной таблице. То есть фактически какие-либо действия отсутствуют.
- **SET NULL**: при удалении связанной строки из главной таблицы устанавливает для столбца внешнего ключа значение NULL.
- **SET DEFAULT**: при удалении связанной строки из главной таблицы устанавливает для столбца внешнего ключа значение по умолчанию, которое задается с помощью атрибуты DEFAULT. Если для столбца не задано значение по умолчанию, то в качестве него применяется значение NULL.

Удаление таблиц

Для удаления таблиц используется команда **DROP TABLE**, которая имеет следующий синтаксис:

```
DROP TABLE table1 [, table2, ...]
```

Изменение таблицы

Возможно, в какой-то момент мы захотим изменить уже имеющуюся таблицу. Например, добавить или удалить столбцы, изменить тип столбцов, добавить или удалить ограничения. То есть потребуется изменить определение таблицы. Для изменения таблиц используется выражение **ALTER TABLE**.

Общий формальный синтаксис команды выглядит следующим образом:

```
ALTER TABLE название_таблицы [WITH CHECK | WITH NOCHECK]
{ ADD название_столбца тип_данных_столбца [атрибуты_столбца] |
  DROP COLUMN название_столбца |
  ALTER COLUMN название_столбца тип_данных_столбца [NULL|NOT NULL] |
  ADD [CONSTRAINT] определение_ограничения |
  DROP [CONSTRAINT] имя_ограничения}
```

Добавление данных. Команда Insert

Для добавления данных применяется команда **INSERT**, которая имеет следующий формальный синтаксис:

```
INSERT [INTO] имя_таблицы [(список_столбцов)] VALUES (значение1, значение2, ... значениеN)
```

Вначале идет выражение **INSERT INTO**, затем в скобках можно указать список столбцов через запятую, в которые надо добавлять данные, и в конце после слова **VALUES** скобках перечисляют добавляемые для столбцов значения.

Обновление данных. Команда UPDATE

Для изменения уже имеющихся строк в таблице применяется команда **UPDATE**. Она имеет следующий формальный синтаксис:

```
UPDATE имя_таблицы
SET столбец1 = значение1, столбец2 = значение2, ... столбецN = значениеN
[FROM выборка AS псевдоним_выборки]
[WHERE условие_обновления]
```

Удаление данных. Команда DELETE

Для удаления применяется команда **DELETE**:

```
DELETE [FROM] имя_таблицы
WHERE условие_удаления
```

Задание для выполнения

1 Создайте и заполните базу, спроектированную на предыдущем занятии.

2 Придумайте и реализуйте 10 любых запросов на выборку для БД, разработанной на предыдущих занятиях.

Указания по технике безопасности

Техника безопасности при выполнении лабораторной работы совпадает

с общепринятой для пользователей персональных компьютеров: самостоятельно не производить ремонт персонального компьютера, установку и удаление программного обеспечения; в случае неисправности персонального компьютера сообщить об этом обслуживающему персоналу лаборатории (оператору, администратору); соблюдать правила техники безопасности при работе с электрооборудованием; не касаться электрических розеток металлическими предметами; рабочее место пользователя персонального компьютера должно содержаться в чистоте; не разрешается возле персонального компьютера принимать пищу, напитки.

Защита работы

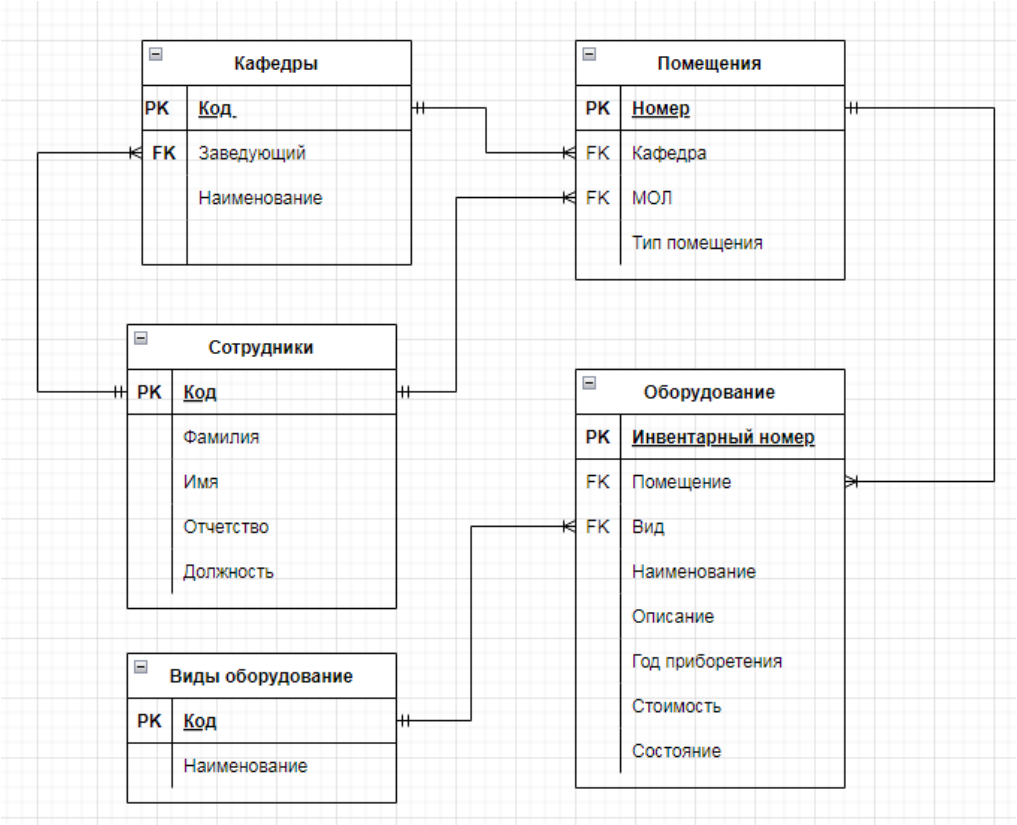
Перед выполнением лабораторной работы каждый студент получает индивидуальное задание. Защита лабораторной работы происходит только после выполнения индивидуального задания и оформления отчета по работе. При защите лабораторной работы студент отвечает на контрольные вопросы, приведенные в конце, и поясняет выполненное индивидуальное задание. Ход защиты лабораторной работы контролируется преподавателем.

Требования к отчету

Отчет должен содержать иллюстрации с подтверждением выполнения задания, пояснения к иллюстрациям, ответы на контрольные вопросы.

Пример выполнения задания

1. В рамках предыдущей работы была спроектирована база данных следующего вида:



С помощью онлайн-сервиса https://sqliteonline.com/ создадим эту базу данных.

1.1 Авторизуйтесь в онлайн-сервисе https://sqliteonline.com/

1.2 Зайдите в раздел SQLite

1.3 Создание базы начнем с таблиц, не имеющих внешних ключей.

Создание таблицы «Виды оборудования»:

```
1 CREATE TABLE VIDY (Kod INT PRIMARY KEY, Name VARCHAR(50))
2
```

После ввода текста запроса необходимо нажать кнопку RUN

Создание таблицы «Сотрудники»:

```
CREATE TABLE SOTR (Kod INT PRIMARY KEY, Familia VARCHAR(50), Imya VARCHAR(50), Otchestvo VARCHAR(50),Dolgnost VARCHAR(50))
```

1.4 Далее создаем таблицы, имеющие ссылки на уже созданные:

Создание таблицы «Кафедры»:

```
CREATE TABLE KAFEDRY (Kod INT PRIMARY KEY, Name VARCHAR(50), zaved INT REFERENCES SOTR (ID))
```

Атрибут Zaved ссылается на поле ***ID*** *таблицы* ***SOTR*** *с помощью конструкции zaved int* ***references SOTR (ID)***

Создание таблицы «Помещения»:

```
CREATE TABLE POMESCH (Nomer INT PRIMARY KEY, Tip VARCHAR(50),
                      Kafedra INT REFERENCES KAFEDRY (ID), MOL INT REFERENCES SOTR (ID))
```

Создание таблицы «Оборудование»:

```
CREATE TABLE OBORUD (InvNomer INT PRIMARY KEY, Name VARCHAR(50),
                     Pomecsh INT REFERENCES POMESCH (nomer), Vid INT REFERENCES VIDY (ID),
                   Opisan VARCHAR (255), God INT, Stoim DECIMAL (10,2), Sost VARCHAR (255))
```

1.5 Заполните таблицы в том же порядке, в каком их создавали:

Ввод данных в таблицу «Виды оборудования»:

```
INSERT INTO VIDY VALUES (1, "Мебель")
```

Проверьте введенные данные с помощью команды SELECT:

```
SELECT * FROM VIDY
```

Kod	Name
1	Мебель

Ввод данных в таблицу «Сотрудники»:

```
INSERT INTO SOTR VALUES (1, "Иванов",  "Иван", "Иванович")
```

Проверка:

```
SELECT * FROM SOTR
```

Kod	Familia	Imya	Otchestvo
1	Иванов	Иван	Иванович

Ввод данных в таблицу «Кафедры»:

```
INSERT INTO KAFEDRY VALUES (1, "АЭСиЭ",  1)
```

Ввод данных в таблицу «Помещения»:

```
INSERT INTO POMESCH (nomer, kafedra, mol) VALUES (502, 1,  1)
```

В данном случае заполнены не все поля таблицы, поэтому перечисляются заполняемые поря в порядке заполнения. Результат:

```
SELECT * FROM POMESCH
```

Nomer	Tip	Kafedra	MOL
502	NULL	1	1

1.6 Введите записи в каждую таблицу:

Виды оборудования:

Kod	Name
1	Мебель
2	Оргтехника
3	Осветительные приборы

Сотрудники:

Kod	Familia	Imya	Otchestvo
1	Иванов	Иван	Иванович
2	Кузнецов	Илья	Петрович
3	Семенова	Ольга	Петровна
4	Орлова	Ирина	Павловна
5	Соколов	Иван	Сергеевич

Кафедры:

Kod	Name	zaved
1	АЭСиЭ	1
2	ЗЧС	3
3	Математики	4

Помещения

Nomer	Tip	Kafedra	MOL
502	NULL	1	1
503	Лекционная	1	1
511	Лекционная	2	4
415	Лаборатория	1	2
416	Лаборатория	3	5

Оборудование

InvNomer	Name	Pomecsh	Vid	Opisan	God	Stoim	Sost
123	Стол	511	1	Новый	2015	1400	Удовлетворительное
124	Стул	415	1	Старый	2012	1000	Сломан
125	ПК	511	2	Учебный компьютер	2020	21000	Рабочий
126	ПК	511	2	Учебный компьютер	2020	21000	Рабочий
127	ПК	511	2	ПК Преподавателя	2021	24000	Рабочий
131	Стул	503	1	Стул преподавателя	2012	1000	Рабочий

1.7 Сохраните БД с помощью пункта меню File-SaveDB

1.8 Удалите таблицы из БД. *Удалять необходимо в порядке, обратном созданию!*

```
1 DROP TABLE OBORUD
2
3
```

Контрольные вопросы

1. Какая команда используется для создания таблицы БД?
2. Какая команда используется для выборки данных из БД?
3. Какая конструкция используется для связи таблиц?
4. Чем отличаются операторы DROP и DELETE?

Лабораторная работа 4. Выборка данных с помощью SQL

Теоретические сведения

Выборка данных

Для выборки информации из одной или более таблиц базы данных используется команда SELECT. Это исключительно мощная команда, способная выполнять действия, эквивалентные операторам реляционной алгебры selection, projection и join, причем в пределах одной выполняемой команды. Общий формат команды SELECT имеет вид:

SELECT [DISTINCT | ALL] {* | [*column* [AS *new_name*]] [, …]}

FROM *table_name* [*alias*] [, …]

[WHERE *condition*]

[GROUP BY *column_list*] [HAVING *condition*]

[ORDER BY *column_list*]

Здесь параметр *column* представляет собой имя столбца или выражение из нескольких имен. Параметр *table_name* является именем существующих в базе данных таблицы или представления, к котрым необходимо получить доступ. Необязательный параметр *alias* – это сокращение, устанавливаемое для имени таблицы *table_name.* Обработка элементов команды SELECT выполняется в следующей последовательности:

FROM Определяются имена используемой таблицы или нескольких таблиц

WHERE Выполняется фильтрация строк объекта в соответствии с заданными условиями

GROUP BYОбразуются группы строк, имеющих одно и то же значение в указанном столбце

HAVING Фильтруются группы строк объекта в соответствии с указанным условием

SELECT Устанавливается, какие столбцы должны присутствовать в выходных данных

ORDER BY Определяется упорядоченность результатов выполнения оператора

Порядок предложений и фраз в команде SELECT *не может* быть изменен. Только два ключевых слова команды – SELECT и FROM – являются обязательными. Команда SELECT является замкнутой: результатом запроса к таблице является друга таблица.

Если требуются данные из более чем 1 таблицы базы данных, используется условие *соединения*. Строки 1 таблицы соединяются со стоками другой согласно общим значениям в соответствующих столбцах – столбцах первичных и внешних ключей.

Имеется 2 основных типа соединений:

- эквисоединения;
- прочие виды соединения (не-эквисоединения).

К дополнительным методам соединения относятся:

- внешние соединения;
- соединения с собой;
- операторы множеств;

Для отображения данных из 2 или более связанных таблиц необходимо задать простое условие соединения в предложении WHERE

SELECT *таблица.столбец, таблица.столбец*

FROM *таблица1, таблица2*

WHERE *таблица1.столбец1=таблица2.столбец2*

где *таблица.столбец* – таблица и столбец, из которых производится выборка данных;

таблица1.столбец1=таблица2.столбец2 - условие, соединяющее таблицы (или задающее их взаимосвязь)

Для упрощения чтения команд SELECT, предполагающих соединение таблиц, и расширения возможностей доступа к базе данных следует указывать имена таблиц перед именами столбцов. Если более, чем в 1

таблице, есть столбцы с одинаковыми именами, указание имени перед столбцом обязательно. Минимальное количество условий соединения таблиц равно количеству соединяемых таблиц минус 1. Следовательно, для соединения 4 таблиц требуется по крайней мере 3 условия.

Различение столбцов таблиц с помощью имен таблиц может отнимать очень много времени – особенно, если имена у таблиц длинные. Поэтому вместо имен используются псевдонимы таблиц. Псевдонимы таблиц действительны только для данной команды SELECT. Если псевдоним таблицы создан, перед ссылкой на столбец следует указывать его, а не имя таблицы.

По умолчанию, все строки таблицы рассматриваются как одна группа. Для разбиения таблицы на меньшие группы строк используется предложение GROUP BY. Кроме того, для отбора возвращаемых групп используется предложение HAVING.

Синтаксис:

SELECT *столбец, групповая функция*
FROM *таблица*
[WHERE *условие*]
[GROUP BY *выражение_группирования*]
[HAVING *условие_группы*]
[ORDER BY *столбец*]

где

выражение_группирования задает столбец, на основе значения которого группируются строки,

условие_группы включает в выходной результат только те группы, для которых заданное условие истинно.

групповая функция – одна из нижеследующих:

Функция	**Описание**
AVG(DISTINCT/ALL n)	Среднее значение n

COUNT(DISTINCT/ALL/*выражение*/*)	Количество строк только с определенными результатами вычисления *выражения*. По * подсчитываются все строки, включая повторяющиеся значения и строки с неопределенным значением
MAX(DISTINCT/ALL/*выражение*)	Максимальное значение *выражения*
MIN(DISTINCT/ALL/*выражение*)	Минимальное значение *выражения*
STDDEV(DISTINCT/ALL/*n*)	Стандартное отклонение n
SUM(DISTINCT/ALL/*n*)	Сумма значений n
VARIANCE(DISTINCT/ALL/*n*)	Дисперсия n

Если задано слово DISTINCT, функция учитывает только неповторяющиеся значения, при наличии слова ALL (этот вариант принимается по умолчанию) рассматриваются все значения.

Все групповые функции, кроме COUNT, игнорируют неопределенные значения. Для подстановки значения вместо неопределенного используют NVL.

Например, для определения суммы каждого заказа (без учета скидки) можно воспользоваться следующим запросом:

```
SELECT SUM(Цена* Количество), КодЗаказа
FROM Заказано
GROUP BY КодЗаказа;
```

Этот запрос можно дополнить условием, чтобы выводились только те заказы, общая сумма которых более 10000р:

```
SELECT SUM(Цена* Количество), КодЗаказа
FROM Заказано
GROUP BY КодЗаказа
HAVING SUM(Цена* Количество)>10000
```

В случае использования группировки, в предложении SELECT должны быть только поля, участвующие в группировке или в групповом выражении.

Задание для выполнения

1. Придумайте и создайте 10 запросов с группировкой для БД, созданной на предыдущих занятиях.

Указания по технике безопасности

Техника безопасности при выполнении лабораторной работы совпадает с общепринятой для пользователей персональных компьютеров: самостоятельно не производить ремонт персонального компьютера, установку и удаление программного обеспечения; в случае неисправности персонального компьютера сообщить об этом обслуживающему персоналу лаборатории (оператору, администратору); соблюдать правила техники безопасности при работе с электрооборудованием; не касаться электрических розеток металлическими предметами; рабочее место пользователя персонального компьютера должно содержаться в чистоте; не разрешается возле персонального компьютера принимать пищу, напитки.

Защита работы

Перед выполнением лабораторной работы каждый студент получает индивидуальное задание. Защита лабораторной работы происходит только после выполнения индивидуального задания и оформления отчета по работе. При защите лабораторной работы студент отвечает на контрольные вопросы, приведенные в конце, и поясняет выполненное индивидуальное задание. Ход защиты лабораторной работы контролируется преподавателем.

Требования к отчету

Отчет должен содержать иллюстрации с подтверждением выполнения задания, пояснения к иллюстрациям, ответы на контрольные вопросы.

Пример выполнения задания

1 Откройте сохраненную базу с помощью пункта меню File-OpenDB

2 Покажите список кафедр с фамилиями заведующих: *(пример соединения таблиц)*

```
SELECT KAFEDRY.Kod, KAFEDRY.Name, SOTR.Familia
FROM KAFEDRY,SOTR
WHERE KAFEDRY.zaved=SOTR.Kod
```

Kod	Name	Familia
1	АЭСиЭ	Иванов
2	ЗЧС	Семенова
3	Математики	Орлова

3 Покажите все оборудование, установленное в помещении 511: *(пример выборки по условию)*

```
SELECT OBORUD.InvNomer, OBORUD.Name, OBORUD.Vid, OBORUD.Pomecsh
FROM OBORUD
WHERE OBORUD.Pomecsh=511
```

InvNomer	Name	Vid	Pomecsh
123	Стол	1	511
125	ПК	2	511
126	ПК	2	511
127	ПК	2	511

4 Покажите все оборудование с указанием его вида, установленное в помещении 511: *(пример выборки по условию с соединением таблиц)*

```
SELECT OBORUD.InvNomer, OBORUD.Name, VIDY.Name, OBORUD.Pomecsh
FROM OBORUD, VIDY
WHERE OBORUD.Pomecsh=511
AND OBORUD.Vid=VIDY.Kod
```

InvNomer	Name	Name	Pomecsh
123	Стол	Мебель	511
125	ПК	Оргтехника	511
126	ПК	Оргтехника	511
127	ПК	Оргтехника	511

Покажите все оборудование с указанием его вида в столбце «Вид», установленное в помещении 511: *(пример выборки по условию с соединением таблиц и переименованием столбца)*

```
SELECT OBORUD.InvNomer, OBORUD.Name, VIDY.Name AS "ВИД", OBORUD.Pomecsh
FROM OBORUD, VIDY
WHERE OBORUD.Pomecsh=511
AND OBORUD.Vid=VIDY.Kod
```

InvNomer	Name	ВИД	Pomecsh
123	Стол	Мебель	511
125	ПК	Оргтехника	511
126	ПК	Оргтехника	511
127	ПК	Оргтехника	511

5. Покажите количество оборудования на каждой кафедре:

```
SELECT KAFEDRY.Name, COUNT(OBORUD.InvNomer) AS "Количество"
FROM OBORUD, POMESCH, KAFEDRY
WHERE OBORUD.Pomecsh=POMESCH.Nomer AND POMESCH.Kafedra=KAFEDRY.Kod
GROUP BY KAFEDRY.Name
```

Name	Количество
АЭСиЭ	2
ЗЧС	4

6. Покажите, на каких кафедрах количество оборудования более 3:

```
SELECT KAFEDRY.Name, COUNT(OBORUD.InvNomer) AS "Количество"
FROM OBORUD, POMESCH, KAFEDRY
WHERE OBORUD.Pomecsh=POMESCH.Nomer AND POMESCH.Kafedra=KAFEDRY.Kod
GROUP BY KAFEDRY.Name
HAVING COUNT(OBORUD.InvNomer)>3
```

Name	Количество
ЗЧС	4

Контрольные вопросы

1. Перечислите агрегатный функции
2. Чем отличаются конструкции WHERE и HAVING?
3. Как используется функция GOUP BY?

Лабораторная работа 5. Использование БД в документах HTML

Теоретические сведения

Сценарий или *скрипт* — программа на специализированном языке программирования, выполняющаяся в режиме интерпретации.

Сценарий — это специальная программа, написанная на особом языке программирования (скриптовом языке, языке сценариев), расширяющая функциональные возможности web-страницы по сравнению с обычным HTML-документом.

Согласно классическому определению, сценарий (script) — это последовательность команд (программа), которая интерпретируется и обрабатывается другой программой. Это означает, что для написания сценария достаточно текстового редактора, в то время как для создания программы требуется другая программа (по крайней мере, компилятор). Иными словами, сценарии пишутся легче и быстрее, чем программы, создаваемые на более структурированных языках, таких как C и C++ и др.

Однако для выполнения сценария требуется больше времени, чем это необходимо для скомпилированной программы. Здесь каждая команда сценария сначала обрабатывается другой программой (а значит, эту дополнительную программу нужно еще загрузить и выполнить многие другие команды), а не непосредственно процессором. Впрочем, этот недостаток с успехом компенсируется тем, что для создания сценария нужен всего лишь бесплатный текстовый редактор, в то время как компилятор для языка C++ стоит сотни долларов.

Языки программирования, посредством которых кодируются сценарии, во многом похожи на современные объектно-ориентированные, но имеют и значительные отличия, вызванные их спецификой. В настоящее время для разработки сценариев используются два языка — JavaScript, предложенный инженерами компании Netscape Communications,

и Visual Basic Script, разработанный на базе языка Visual Basic специалистами фирмы Microsoft. Последний чаще обозначается как VBScript или иногда VBS — в противоположность другой разновидности Visual Basic`a — VBA (Visual Basic for Applications). Не следует путать JavaScript и Java: хотя синтаксис JavaScript напоминает Java, это два разных языка.

В настоящее время JavaScript поддерживается всеми популярными браузерами: Internet Explorer, Mozilla Firefox, Safari, Google Chrome и Opera.

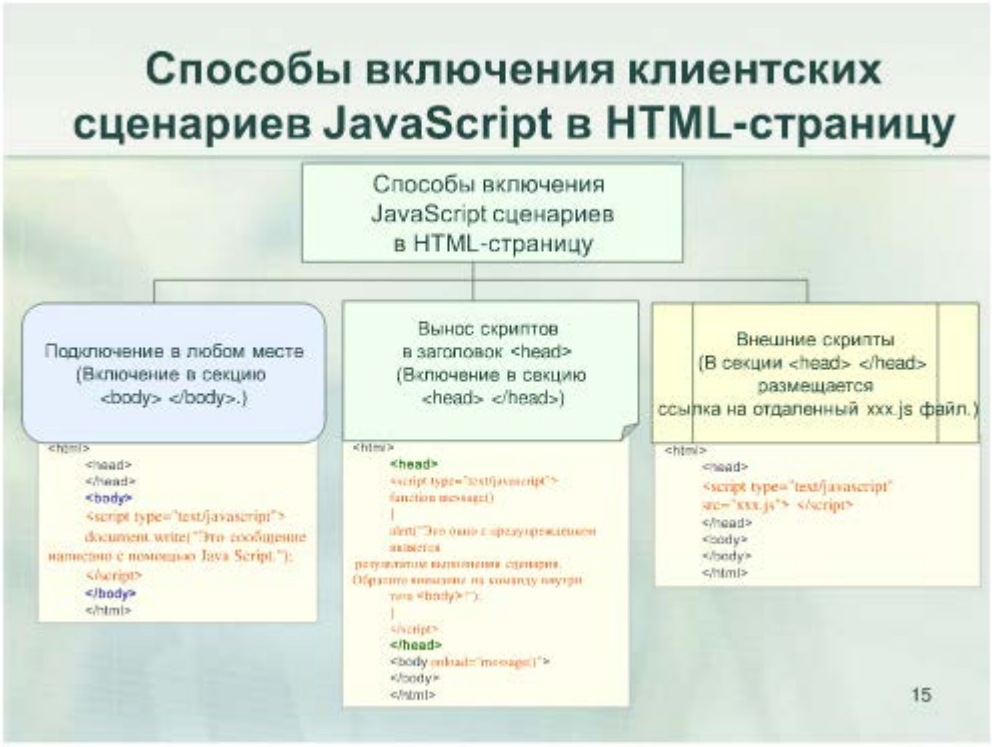
Существует три базовых способа встраивания java-скриптов в web-страницу (см. рис):

1. Расположение внутри тега. Задание скрипта внутри какого-нибудь тега. Например, внутри ссылки будет указана функция вида onclick= «javascript».

2. Отделение от разметки. Задание скрипта отдельно от тегов, например в начале HTML- документа.

3.Вынесение в отдельный файл.

Также можно написать скрипт в отдельном файле, а затем указать к нему путь из web-страницы



Когда браузер читает HTML-страницу и видит <script>, он первым делом читает и выполняет код, а только потом продолжает читать страницу далее.

Тег <script> сообщает браузеру о том, что внутри находится выполняемый скрипт. Атрибут type говорит о том, что это javascript.

Задание для выполнения

Выполнить задания из примера выполнения работы

Указания по технике безопасности

Техника безопасности при выполнении лабораторной работы совпадает с общепринятой для пользователей персональных компьютеров: самостоятельно не производить ремонт персонального компьютера, установку и удаление программного обеспечения; в случае неисправности персонального компьютера сообщить об этом обслуживающему персоналу лаборатории (оператору, администратору); соблюдать правила техники

безопасности при работе с электрооборудованием; не касаться электрических розеток металлическими предметами; рабочее место пользователя персонального компьютера должно содержаться в чистоте; не разрешается возле персонального компьютера принимать пищу, напитки.

Защита работы

Перед выполнением лабораторной работы каждый студент получает индивидуальное задание. Защита лабораторной работы происходит только после выполнения индивидуального задания и оформления отчета по работе. При защите лабораторной работы студент отвечает на контрольные вопросы, приведенные в конце, и поясняет выполненное индивидуальное задание. Ход защиты лабораторной работы контролируется преподавателем.

Требования к отчету

Отчет должен содержать иллюстрации с подтверждением выполнения задания, пояснения к иллюстрациям, ответы на контрольные вопросы.

Пример выполнения задания

1.Вывести на экран сведения о браузере, используемом для загрузки документа.

Текст программы:

```
<HTML>     // заключает в себя весь текст HTML-документа.
<HEAD>    // дескриптор, указывающий на начало заголовка документа.
<TITLE>Браузер, используемый для загрузки документа</TITLE> // заголовок.
</HEAD>   // дескриптор, указывающий на конец заголовка документа.
<BODY>   // задает начало и конец тела HTML-документа.
 <H1>Вывод на экран сведений о браузере</H1> // заголовок в теле документа.
<HR> // линия.
<SCRIPT LANGUAGE="JavaScript">    //дескриптор описания языка сценария.
document.write('Используемый браузер версии '+ navigator.appVersion)
document.write("называется <B>" + navigator.appName +'<B/>.') // вывод
названия и версии браузера.
</SCRIPT>          // закрытие дескриптора SCRIPT.
</BODY>            // закрытие дескриптора BODY.
```

```
</HTML>          // конец HTML-документа
```

2. Создайте HTML-код, который сразу после загрузки страницы отображал бы диалоговое окно с предупреждением «Загрузка страницы завершена». Другое диалоговое окно должно отображаться при щелчке пользователя на кнопке формы.

Текст программы:

```
<HTML>     // заключает в себя весь текст HTML-документа.
<HEAD>     // дескриптор, указывающий на начало заголовка документа.
<TITLE>Отображение диалогового окна сразу после загрузки
страницы</TITLE>          // заголовок.
<SCRIPT LANGUAGE="JavaScript">   //дескриптор описания языка сценария

function f1(){
alert(" Загрузка страницы завершена!")} // функция, при вызове которой, выводится
диалоговое окно с сообщением о завершении загрузки страницы.

function f2(){
alert (" Клик по странице! ")} // функция, при вызове которой, выводится диалоговое
окно с сообщением.

</SCRIPT>  // закрытие дескриптора SCRIPT

</HEAD>   // закрытие дескриптора HEAD
<BODY onLoad="f1()"> // открытие дескриптора тела документа в обработчике
событий onLoad, операторы сценария начинают выполняться сразу после загрузки
документа.
<FORM> // дескриптор объекта формы.
<INPUT TYPE='button'VALUE="Нажмите"onClick='f2()'> // создание кнопки,
при щелчке на которой в обработчике событий onClick вызывается функция f1().
</FORM>    //закрытие дескриптора объекта формы.
</BODY>    // закрытие дескриптора тела документа
</HTML>   // конец HTML-документа
```

3. Создайте HTML-документ, который содержит 2 поля ввода для чисел, 1 поле для вывода результата и объект кнопка для выполнения арифметической операции вычитания.

Текст программы:

```
<HTML>      // заключает в себя весь текст HTML-документа.
```

```
  <HEAD>   // дескриптор, указывающий на начало заголовка документа.
<TITLE>Вычитание</TITLE> // заголовок.
  <SCRIPT LANGUAGE="JavaScript">  //дескриптор описания языка сценария
  function vych(form){         //функция, которая производит арифметическую  операцию вычитания
var a=parseFloat(form.entry1.value) // объявление и инициализация переменной первого числа.
var b=parseFloat(form.entry2.value) // объявление и инициализация переменной второго числа.
   var s=a-b  // вычитание из первого числа второго.
form.entry3.value=s  // присвоение результата третей переменной
}
function obnovlenie(form){    //функция обновления.
form.entry1.value=""  //присвоение полям пустые строки
form.entry2.value=""
form.entry3.value=""
}
</SCRIPT> // закрытие дескриптора SCRIPT
</HEAD> // закрытие дескриптора HEAD
<BODY> // открытие дескриптора тела документа
<FORM>   //открытие дескриптора объекта формы.
Введите числа
Число1:<INPUT TYPE='text'NAME="entry1"> //создание текстового поля для первого числа.
Число2:<INPUT TYPE='text'NAME="entry2"> // создание текстового поля для первого числа.
<INPUT TYPE='button' VALUE="Вычесть" onClick='vych(this.form)'>
//создание кнопки, при щелчке на которой в обработчике событий onClick вызывается функция вычитания.
Результат:<INPUT TYPE='text'NAME="entry3">// создание текстового поля для результата вычитания
   <INPUT TYPE='button' VALUE="Обновить" onClick='obnovlenie(this.form)'>
</FORM>  //закрытие дескриптора объекта формы.
</BODY>  // закрытие дескриптора тела документа
</HTML> //конец HTML-документа
```

4. Создайте Web-страницу, на которой пользователь мог бы ввести шифр специальности и после щелчка на кнопке получить данные с указанием проходного бала и количество поступивших. Сделайте так, чтобы эта информация выводилась на станице в отдельных полях (не менее 15 записей).

Текст программы:

```
<HTML>  // заключает в себя весь текст HTML-документа.
<HEAD>  // дескриптор, указывающий на начало заголовка документа.
<SCRIPT LANGUAGE="JavaScript"> //дескриптор описания языка сценария.
var schifr=new Array(16) // описание массива schifr, выделение места в памяти.
  schifr[0]="220201" //описание первого элемента массива.
  schifr[1]="220202" // описание второго элемента массива и т. д.
  schifr[2]="220203"
  schifr[3]="220204"
  schifr[4]="220205"
  schifr[5]="220206"
  schifr[6]="220207"
  schifr[7]="220208"
  schifr[8]="220209"
  schifr[9]="220210"
  schifr[10]="220211"
  schifr[11]="220212"
  schifr[12]="220213"
  schifr[13]="220214"
  schifr[14]="220215"
  schifr[15]="220216"

var ball=new Array(16) //описание второго массива ball,выделение места в памяти.
  ball[0]="175" // описание первого элемента этого массива
  ball[1]="180"
  ball[2]="185"
  ball[3]="175"
  ball[4]="189"
  ball[5]="190"
  ball[6]="220"
  ball[7]="180"
  ball[8]="158"
  ball[9]="290"
  ball[10]="195"
  ball[11]="240"
  ball[12]="195"
  ball[13]="175"
  ball[14]="175"
  ball[15]="175" //описание последнего элемента этого массива
var kol=new Array(16) //описание третьего массива kol,выделение места в памяти.
  kol[0]="12"  // описание первого элемента массива
  kol[1]="13"
  kol[2]="14"
```

```
  kol[3]="15"
  kol[4]="16"
  kol[5]="17"
  kol[6]="18"
  kol[7]="19"
  kol[8]="20"
  kol[9]="21"
  kol[10]="22"
  kol[11]="23"
  kol[12]="24"
  kol[13]="25"
  kol[14]="26"
  kol[15]="27" // описание последнего элемента массива

function f1(){     //функция находит элемент совпадающий с введенным.
    var selectSchifr=document.entryForm.entry.value //присвоение более короткого имени объекту document.entryForm.entry.value
    for(var i=0;i<schifr.length;i++) //в цикле
    {
    if(schifr[i]==selectSchifr) //если введенный шифр совпадает с одним из элементов массива.
    {
    break // выход из цикла
    }
    }
  var msg=schifr[i]+" проходной балл: "+ball[i]+' количество поступивших '+kol[i] //описание переменной, при выводе которой будут выводиться шифр, балл и количество поступивших
document.entryForm.output.value=msg // присвоение результата переменной output
}

</SCRIPT>  // закрытие дескриптора SCRIPT
</HEAD>  //.закрытие дескриптора HEAD
<BODY>  // открытие дескриптора тела документа
<FORM NAME='entryForm'> // открытие дескриптора объекта формы.
<INPUT TYPE='text'NAME="entry"> // создание текстового поля для ввода шифра
<INPUT TYPE='button'VALUE="Указать данные"onClick='f1()'>// создание кнопки, при щелчке на которой в обработчике событий onClick вызывается функция f1().
<BR> // с новой строки
<INPUT TYPE='text'size="70"NAME='output'> // создание текстового поля с размером 70 для вывода результата.
</FORM>  //закрытие дескриптора объекта формы.
</BODY>  //закрытие дескриптора объекта формы.
</BODY>  // закрытие дескриптора тела документа
  </HTML> // конец HTML-документа
```

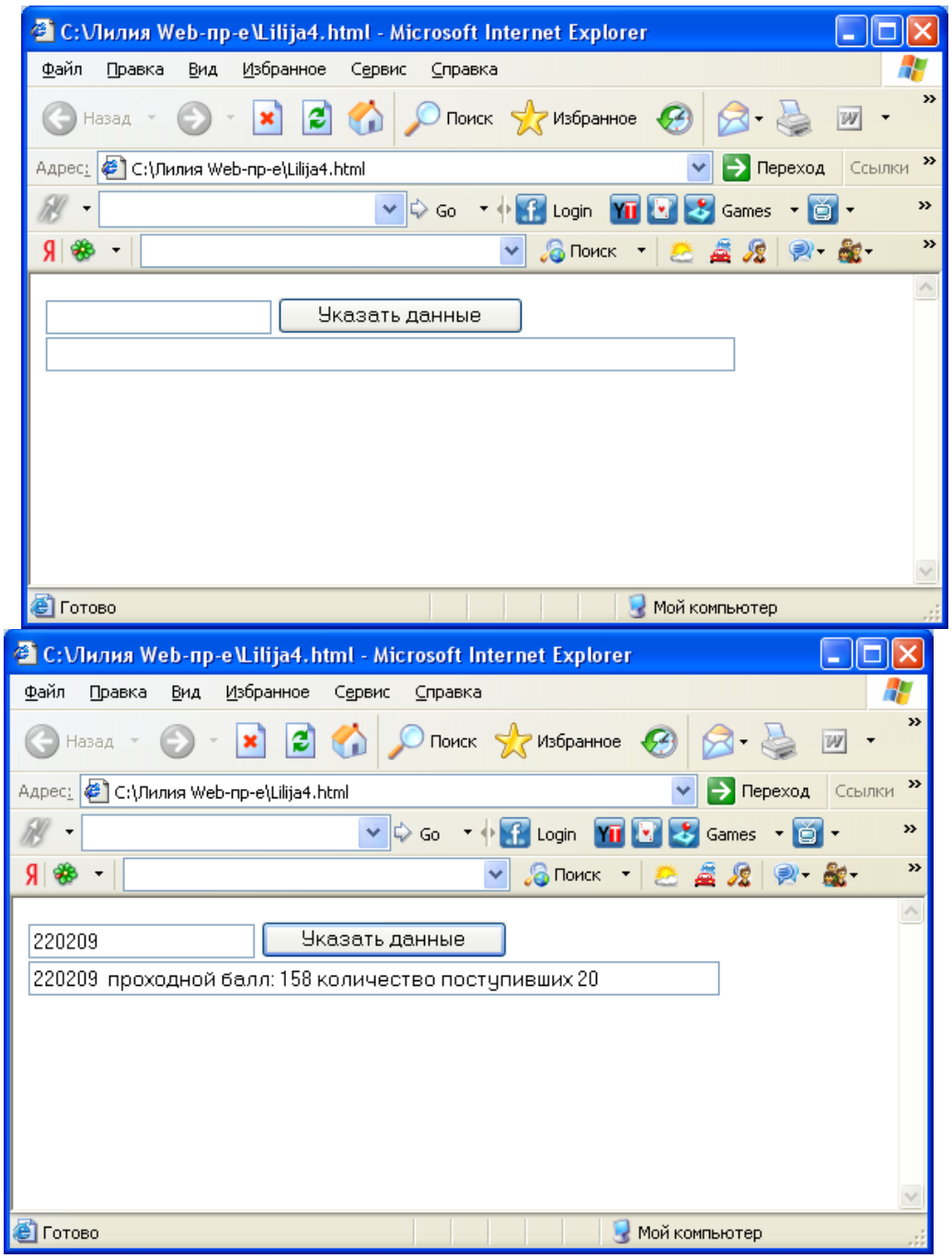


Рис.7,8. Ввод шифра, вывод соответствующих баллов и количество поступивших.

5. Напишите HTML-код:

- Отображающий сообщение в стоке состояния, приветствующий нового посетителя Web-страницы;
- На уровне заголовка H1;
- Пользователю предлагается ввести имя в диалоговом окне, а после этого приветствовать его по имени. Приветствие должно отображаться в основной части документа;
- Сразу после загрузки должно автоматически выводиться диалоговое окно с адресом URL текущей страницы.

Текст программы:

```
<HTML>  // заключает в себя весь текст HTML-документа.
<HEAD> // начало заголовка документа
  <SCRIPT LANGUAGE='JavaScript'> //дескриптор описания языка сценария
function privetstvie(){ //функция приветствия.
 var name=prompt(" Как Вас зовут?") // генерирует диалоговое окно, в котором есть поле для ввода имени.
  return name //возврат имени в качестве параметра
}
function adres(){ // функция  выводящая в диалоговом окне адрес URL страницы.
 alert(" Адрес страницы "+ location.href)
}
   window.status="Привет!!!!!!" //вывод текста в строке состояния.
</SCRIPT>  //закрытие дескриптора SCRIPT.
</HEAD> //закрытие дескриптора HEAD
<BODY onLoad="adres()"> // выполнение операторов функции adres() сразу после загрузки документа
<H1>Здравствуйте!</H1> //заголовок в теле документа
<SCRIPT LANGUAGE="JavaScript">   // дескриптор описания языка сценария.
document.write(" Добро пожаловать, "+privetstvie()+" !") //вывод приветствия
</SCRIPT> //закрытие дескриптора SCRIPT
</BODY>  // закрытие дескриптора тела документа
</HTML> // конец HTML-документ
```

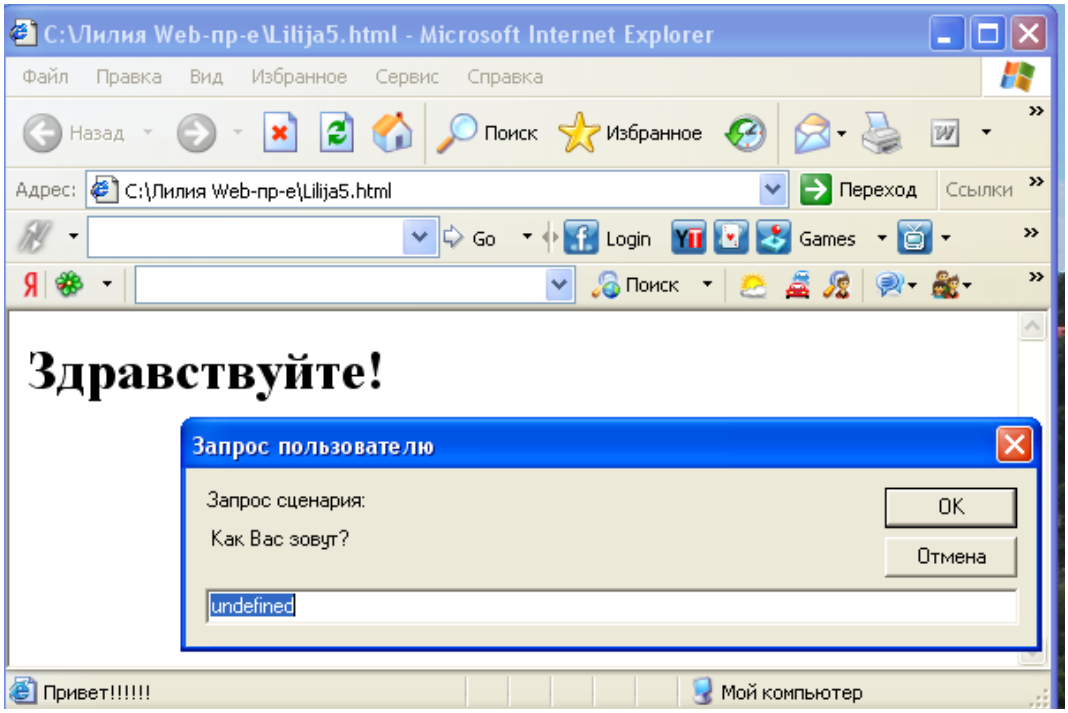


Рис.9. Приветствие в строке состояния и на уровне H1.

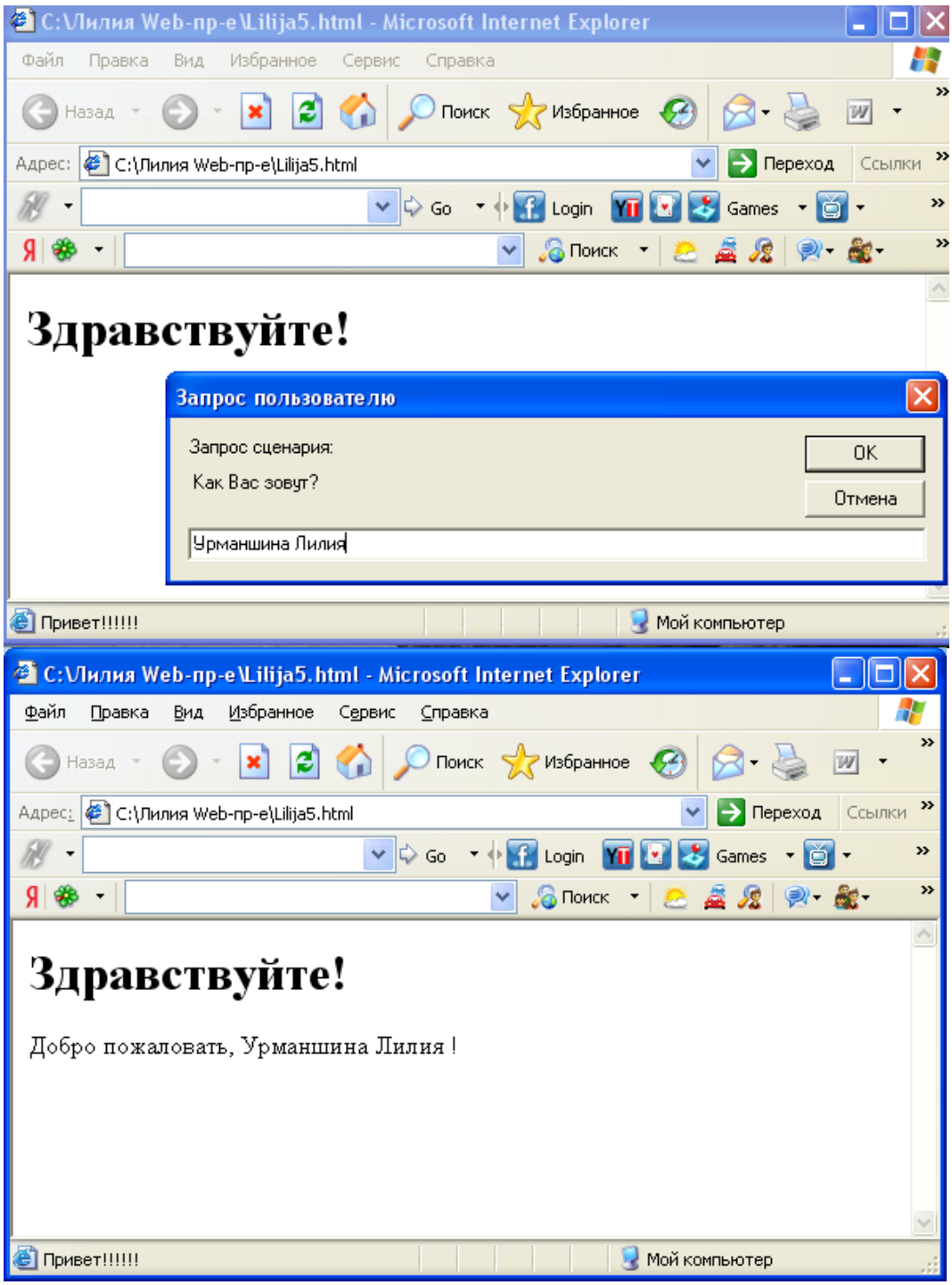


Рис.10,11. Ввод имени, приветствие по имени.

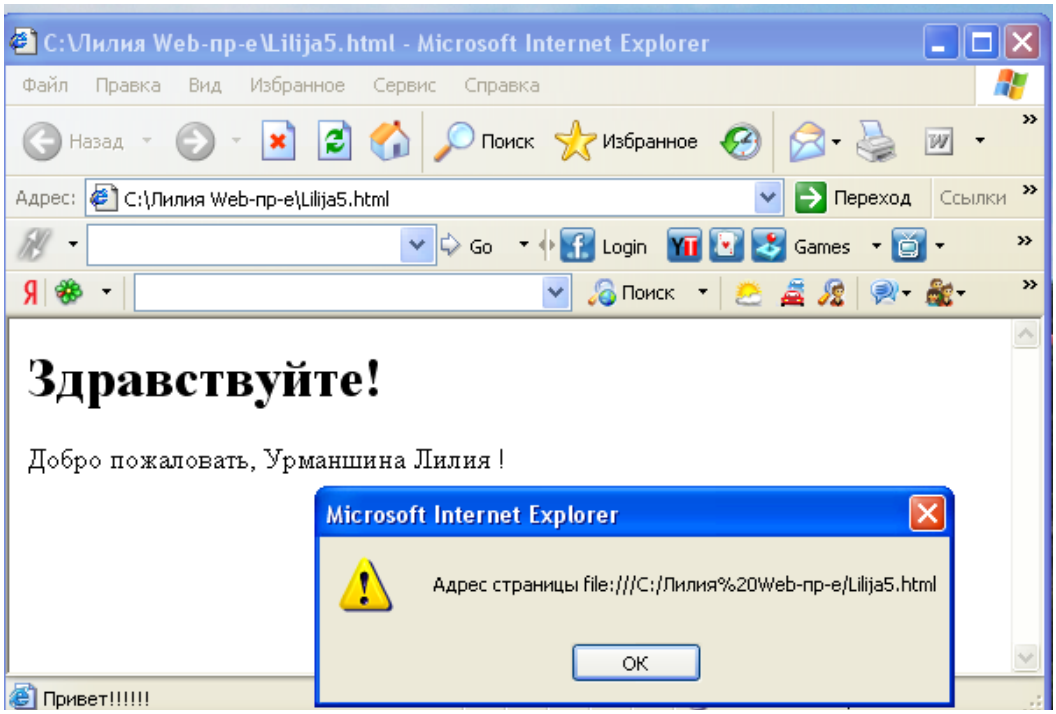


Рис.12. Вывод адреса URL текущей страницы

Контрольные вопросы

1. Для чего используются java-скрипты?
2. Каким образом выполняется встраивание java-скриптов в web-страницу?
3. Каким образом можно разместить БД на странице сайта?

Приложение 1. Индивидуальные задания

1. Проектирование и реализация базы данных электрооборудования подстанции.
2. Проектирование и реализация базы данных электрооборудования линии электропередачи.
3. Проектирование и реализация базы данных ремонтов оборудования подстанции.
4. Проектирование и реализация базы данных ремонтов оборудования линии электропередачи.
5. Проектирование и реализация базы данных потребителей энергорайона.
6. Проектирование и реализация хранилища данных оборудования ЛЭП
7. Проектирование и реализации базы данных показаний счетчиков потребителей энергорайона
8. Проектирование и реализации базы данных плавок гололеда
9. Проектирование и реализации базы данных плановых отключений
10. Проектирование и реализации базы данных внеплановых отключений
11. Проектирование и реализации базы данных механических характеристик ЛЭП
12. Проектирование и реализации базы данных изоляторов ЛЭП
13. Проектирование и реализации базы данных трансформаторов
14. Проектирование и реализации базы данных солнечных станций
15. Проектирование и реализации базы данных оборудования солнечных станций
16. Проектирование и реализации базы данных выработки солнечных станций
17. Проектирование и реализации базы данных ВЭС

18. Проектирование и реализации базы данных оборудования ВЭС
19. Проектирование и реализации базы данных выработки ВЭС
20. Проектирование и реализации базы данных ГЭС
21. Проектирование и реализации базы данных оборудования ГЭС
22. Проектирование и реализации базы данных выработки ГЭС
23. Проектирование и реализации базы данных ремонтов оборудования солнечных станций
24 Проектирование и реализации базы данных ремонтов оборудования ВЭС
25. Проектирование и реализации базы данных ремонтов оборудования СЭС
26. Проектирование и реализации базы данных состояния оборудования солнечных станций
27 Проектирование и реализации базы данных состояния оборудования ВЭС
28 Проектирование и реализации базы данных состояния оборудования СЭС
29. Проектирование и реализации хранилища данных показаний счетчиков потребителей энергорайона

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

по организации самостоятельной работы студента
по дисциплине
«Системы обработки и хранения данных в электроэнергетике**»**

Направление подготовки 13.03.02 «Электроэнергетика и электротехника»
Направленность (профиль):
«Цифровые технологии в электроэнергетике»
Форма обучения: очная

Ставрополь, 2025

СОДЕРЖАНИЕ

Введение

Самостоятельная работа является внеаудиторной и предназначена для самостоятельного изучения определенных тем дисциплины по материалам, рекомендованным преподавателем, а также в порядке подготовки к практическим занятиям.

Цель дисциплины «Системы обработки и хранения данных в электроэнергетике» – формирование согласно ООП ВО набора компетенций будущего бакалавра по направлению 13.03.02 «Электроэнергетика и электротехника».

Для достижения этой цели в процессе изучения дисциплины решаются следующие задачи: ознакомление с основными понятиями и терминами изучаемой дисциплины, которыми будущий специалист будет оперировать в своей практической деятельности.

1 Общая характеристика самостоятельной работы студента при изучении дисциплины

Основная задача высшего образования заключается в формировании творческой личности специалиста, способного к саморазвитию, самообразованию, инновационной деятельности. Решение этой задачи вряд ли возможно только путем передачи знаний в готовом виде от преподавателя к студенту. Для решения этой задачи в учебные планы всех специальностей включена самостоятельная работа, составляющая 50% учебного времени студента.

В широком смысле под самостоятельной работой студентов следует понимать совокупность всей самостоятельной деятельности студентов, как в учебной аудитории, так и вне ее, в контакте с преподавателем и в его отсутствие.

В учебном процессе выделяют два вида самостоятельной работы:

– аудиторная;

– внеаудиторная.

Аудиторная самостоятельная работа по дисциплине выполняется на учебных занятиях под непосредственным руководством преподавателя и по его заданию.

Внеаудиторная самостоятельная работа выполняется студентом по заданию преподавателя, но без его непосредственного участия.

Содержание внеаудиторной самостоятельной работы определяется в соответствии с рекомендуемыми видами заданий согласно примерной и рабочей программ учебной дисциплины. Видами заданий для внеаудиторной самостоятельной работы по дисциплине «Системы обработки и хранения данных в электроэнергетике» являются:

– для овладения знаниями: чтение текста (учебника, первоисточника, дополнительной литературы), составление плана текста, графическое изображение структуры текста, конспектирование текста, выписки из текста, работа со словарями и справочниками, ознакомление с нормативными

документами, учебно-исследовательская работа, использование аудио- и видеозаписей, компьютерной техники и Интернета и др.;

– для закрепления и систематизации знаний: обработка текста, повторная работа над учебным материалом (учебника, первоисточника, дополнительной литературы, аудио и видеозаписей, составление плана, составление таблиц для систематизации учебною материала, ответ на контрольные вопросы, заполнение рабочей тетради, аналитическая обработка текста (аннотирование, рецензирование, реферирование, конспект-анализ и др);

– для закрепления навыков – подготовка к лабораторным работам, оформление отчетов по лабораторным работам, выполнение дополнительных заданий, выданных преподавателем.

Самостоятельная работа может осуществляться индивидуально или группами студентов в зависимости от цели, объема, конкретной тематики самостоятельной работы, уровня сложности, уровня умений студентов. Контроль результатов внеаудиторной самостоятельной работы студентов может осуществляться в пределах времени, отведенного на обязательные учебные занятия по дисциплине и внеаудиторную самостоятельную работу студентов по дисциплине, может проходить в письменной, устной или смешанной форме.

Следует с первых дней обучения в вузе приучать себя к целесообразному распределению занятий по месяцам и неделям семестра. При этом очень важна равномерная работа как основное условие успешных занятий в вузе.

Высшей ступенью обучения является приобретение навыков самостоятельной творческой работы, а творческому применению знаний нужно учиться параллельно с их приобретением.

Задачами самостоятельной работы студентов по дисциплине «Системы обработки и хранения данных в электроэнергетике» являются:

– формирования готовности студентов к поиску, обработке и применению информации для решения профессиональных задач;

– систематизация и закрепление полученных теоретических знаний и практических умений студентов;

– развитие познавательных способностей и активности студентов, творческой инициативы, самостоятельности, ответственности и организованности;

– формирование самостоятельности мышления, способностей к саморазвитию, самосовершенствованию и самореализации;

– выработка навыков эффективной самостоятельной профессиональной деятельности.

2 План-график выполнения самостоятельной работы

Общий объём самостоятельной работы в рамках изучения дисциплины «Системы обработки и хранения данных в электроэнергетике» составляет 72 академических часа.

В структуре самостоятельной работы предусмотрено самостоятельное изучение теоретического материала, подготовка к лабораторным занятиям.

3 Методические рекомендации по изучению теоретического материала

Большая часть самостоятельной работы студента состоит в изучении литературы. Одна из задач студента – научиться самостоятельно работать с книгой, а это требует определенных затрат энергии и времени. Самостоятельная работа с книгой может быть успешной, если текст не только прочитан, но и законспектирован. Существует несколько форм записей, но любая форма записи не даст нужного результата, если не будет пробуждать мысли того, кто ее ведет, если отсутствует активная работа ума и формирование своих выводов из прочитанного.

Выбор формы записи зависит от индивидуальных особенностей человека, его образованности и опыта. При этом не меньшую роль играет назначение записей, то есть то, какие задачи ставит перед собой человек (для самообразования, для выступления на семинаре, для использования в будущем).

Введение записей мобилизует наряду со зрительной памятью, также и моторную память. Кроме того, у человека, систематически ведущего записи изучаемой литературы, создается свой фонд материалов для быстрого повторения и мобилизации накопленных знаний.

Все записи должны быть убористыми и компактными. Интервалы между строками должны быть достаточными, чтобы вписывать дополнения. Рекомендуется вести записи ручкой, а карандашом или ручкой другого цвета пользоваться для отметок и выделений при последующей работе. Полезно также датировать записи.

Записи могут носить различный характер: план, выписки, тезисы, аннотирование, конспектирование.

1. План - наиболее краткая формой записи. Это перечень вопросов, рассматриваемых в книге или статье. План обычно раскрывает структуру произведения, логику автора, способствует лучшей ориентации в содержании.

Так, составленным планом можно воспользоваться, чтобы вспомнить прочитанное или быстро отыскать в книге нужное место. Представление об основных пунктах плана дает оглавление книги, поэтому во многих случаях наименования глав и разделов можно использовать в качестве пунктов. Составление плана приучает логически мыслить, вырабатывать умение сжато и последовательно излагать суть вопроса в письменной и устной форме.

Существует два способа составления плана: работа над ним по ходу чтения и составление плана после ознакомления с произведением. При этом план получается более последовательным и стройным.

Трудность составления плана состоит в том, что надо выяснить для себя, прежде всего, построение изучаемого текста, ход мыслей автора и лишь затем изложить содержание работы кратко и ясно. При всей своей краткости план дает представление о содержании прочитанного. Следует учесть, что форма плана не исключает цитирования отдельных мет и обобщений. Различают простой и развернутый план. В отличие от простого плана развернутый план не только содержит перечисление вопросов, но и раскрывает основные идей произведения, может включать выдержки из него, схемы, таблицы. Планом, особенно развернутым, необходимо пользоваться при написании выступления или статьи.

В целом развернутый план дает гораздо большее представление о произведении, его основных идеях, задачах, которые в нем решаются. Он может включать положения, замечания, собственные мысли студента.

Важно знать, что составление планов помогает вырабатывать способность к отвлеченному, абстрактному мышлению, но наибольшую пользу составление плана даст подготовленным лицам, которые бывают достаточно лишь взглянуть на перечень основных вопросов, чтобы воспроизвести содержание прочитанного.

2. Тезисы – более сложная и совершенная форма записи, чем составление плана.

Это сжатое изложение основных мыслей прочитанного произведения или подготовляемого вступления. Особенностью тезисов является их утвердительный характер.

В них сосредотачивается самое главное, только выводы и обобщения, в них меньше доказательств, иллюстрации и пояснений. Тезисы не должны повторять дословно текст, но в ряде мест могут быть близки к нему, воспроизводя некоторые характерные выражения автора, важные для понимания хода его мыслей. Составление тезисов помогает глубже понять основные идеи произведения, выделить главное в нем; приучают сжато, точно и четко сформулировать свои мысли, повышает культуру речи и

письма. При составлении тезисов учитывают следующее. Прежде всего, если произведение небольшое, необходимо внимательно изучать его в целом, если большое – изучать по главам и разделам. Затем, когда будут ясны основные идеи, кратко и последовательно излагать их в виде пунктов.

Различают простые и сложные, развернутые тезисы. Если записывают только утверждение чего – либо, такой тезис называют простым, а сложным тезисом будет выражение главной мысли, содержащее, кроме утверждения, еще и краткое ее доказательство.

Часто тезисы формулируются самим автором как выводы и обобщения в заключении книги или разделах книги. Нередко тезисы выделяются в тексте другим шрифтом.

Рекомендуется делать тезисные записи своими словами, причем можно записывать один абзац за другим, учитывая смысловую связь между ними. Но в большинстве случаев следует составлять сводный тезис, сложный по форме. При этом объединяется несколько утверждений, тесно связанных между собой.

Тезисы по содержанию очень близки к конспекту, но конспект носит более описательный характер, и его положения не столь категоричны, как в тезисах. Кроме того, конспект представляет собой более полную форму записи.

Следует отметить, что различие между формами записей условно, но в любой форме запись – важнейшая часть самостоятельной работы с книгой.

3. Выписки. Это записи текста из книги: теоретических положений, статистических данных, имеющих по мнению читателя важное значение.

Достоинство выписок состоит в точности воспроизведения текста книги, удобстве пользования записями при последующей работе, в накоплении обобщений и фактического материала. Выписки полезны для повторения, освежения в памяти прочитанного, для быстрой мобилизации своих знаний, когда необходимо в короткий срок вспомнить материал. Выписки выделяют из текста самое главное и тем самым помогают глубже

понять его. Без них трудно обойтись при подготовке доклада, реферата, выступления. Выписки следует рассматривать как составную часть тезисов и конспектов.

Выписывать текст можно и по ходу чтения и после его завершения. В последнем случае надо замечать места, которые потом будут выписаны. Необходимо каждую выписку снабжать ссылкой на источник с указанием соответствующей станицы. Это нужно, чтобы в последствии можно было быстро найти в книге соответствующее место. Целесообразно выписывать из текста только такие места, в которых содержится самое главное, суть вопроса. Выписки должны быть ориентированы на изучение произведения в целом, а не отдельных мест, поскольку положения, вырванные из общего контакта, понимаются нередко совсем не так, как этого хотел автор. Иначе говоря, отдельно взятые, лишенные пояснений выдержки могут быть не поняты или поняты неправильно.

Выписки бывают дословные (цитаты) и «свободные», когда мысли автора излагаются своими словами. Следует учесть, что большие отрывки, которые трудно цитировать, целесообразнее в краткой форме переложить своими словами, но «яркие» и важные места лучше выписывать дословно. Каждую цитату следует заключать в кавычки. Если ее берут из середины предложения, то после вводных кавычек ставят три точки. Ставят их и в конце цитаты, если из предложения опущены последние слова.

Следует знать, что какого-либо единого метода выписок, годного для всех случаев, не существует, поскольку у каждого человека свои особенности мышления и восприятия, свои подход к теме. Все это влияет на содержание и характер выписок.

Выписки рекомендуется хранить в картотеке, конвертах или папках, на которых следует обозначить общую тему.

4. Аннотация – еще одна форма записи, являющаяся кратким обобщением содержания книги. Ею удобно пользоваться, если имеется

намерение вернуться к изучаемому произведению. Аннотация может быть необходима и для того, чтобы не забыть о нем.

Для составления аннотации надо сначала полностью прочитать и глубоко продумать произведение. При всей своей краткости аннотация может содержать отдельные фрагменты авторского текста, а не только оценку книги или статьи.

5. Резюме очень близко к аннотации. Это запись, являющаяся краткой оценкой прочитанного материала. Различие между ними состоит в том, что аннотация сжато характеризует произведение в целом, а резюме концентрирует внимание на его выводах, главных итогах.

6. Конспект – наиболее совершенная и наиболее сложная форма записи. Слово «конспект» происходит от латинского «conspectus», что означает «обзор, изложение». В правильно составленном конспекте обычно выделено самое основное в изучаемом тексте, сосредоточено внимание на наиболее существенном, в кратких и четких формулировках обобщены важные теоретические положения.

Конспект представляет собой относительно подробное, последовательное изложение содержания прочитанного. На первых порах целесообразно в записях ближе держаться тексту, прибегая зачастую к прямому цитированию автора. В дальнейшем, по мере выработки навыков конспектирования, записи будут носить более свободный и сжатый характер.

Конспект книги обычно ведется в тетради. В самом начале конспекта указывается фамилия автора, полное название произведения, издательство, год и место издания. При цитировании обязательная ссылка на страницу книги. Если цитата взята из собрания сочинений, то необходимо указать соответствующий том. Следует помнить, что четкая ссылка на источник – непременное правило конспектирования. Если конспектируется статья, то указывается, где и когда она была напечатана.

Конспект подразделяется на части в соответствии с заранее продуманным планом. Пункты плана записываются в тексте или на полях

конспекта. Писать его рекомендуется четко и разборчиво, так как небрежная запись с течением времени становиться малопонятной для ее автора. Существует правило: конспект, составленный для себя, должен быть по возможности написан так, чтобы его легко прочитал и кто-либо другой.

Формы конспекта могут быть разными и зависят от его целевого назначения (изучение материала в целом или под определенным углом зрения, подготовка к докладу, выступлению на занятии и т.д.), а также от характера произведения (монография, статья, документ и т.п.). Если речь идет просто об изложении содержания работы, текст конспекта может быть сплошным, с выделением особо важных положений подчеркиванием или различными значками.

В случае, когда не ограничиваются переложением содержания, а фиксируют в конспекте и свои собственные суждения по данному вопросу или дополняют конспект соответствующими материалами их других источников, следует отводить место для такого рода записей. Рекомендуется разделить страницы тетради пополам по вертикали и в левой части вести конспект произведения, а в правой свои дополнительные записи, совмещая их по содержанию.

Конспектирование в большей мере, чем другие виды записей, помогает вырабатывать навыки правильного изложения в письменной форме важные теоретических и практических вопросов, умение четко их формулировать и ясно излагать своими словами.

Таким образом, составление конспекта требует вдумчивой работы, затраты времени и труда. Зато во время конспектирования приобретаются знания, создается фонд записей.

Конспект может быть текстуальным или тематическим. В текстуальном конспекте сохраняется логика и структура изучаемого произведения, а запись ведется в соответствии с расположением материала в книге. За основу тематического конспекта берется не план произведения, а содержание какой-либо темы или проблемы.

Текстуальный конспект желательно начинать после того, как вся книга прочитана и продумана, но это, к сожалению, не всегда возможно. В первую очередь необходимо составить план произведения письменно или мысленно, поскольку в соответствии с этим планом строится дальнейшая работа. Конспект включает в себя тезисы, которые составляют его основу. Но, в отличие от тезисов, конспект содержит краткую запись не только выводов, но и доказательств, вплоть до фактического материала. Иначе говоря, конспект – это расширенные тезисы, дополненные рассуждениями и доказательствами, мыслями и соображениями составителя записи.

Как правило, конспект включает в себя и выписки, но в него могут войти отдельные места, цитируемые дословно, а также факты, примеры, цифры, таблицы и схемы, взятые из книги. Следует помнить, что работа над конспектом только тогда будет творческой, когда она не ограничена текстом изучаемого произведения. Нужно дополнять конспект данными из другими источников.

В конспекте необходимо выделять отдельные места текста в зависимости от их значимости. Можно пользоваться различными способами: подчеркиваниями, вопросительными и восклицательными знаками, репликами, краткими оценками, писать на полях своих конспектов слова: «важно», «очень важно», «верно», «характерно».

В конспект могут помещаться диаграммы, схемы, таблицы, которые придадут ему наглядность.

Составлению тематического конспекта предшествует тщательное изучение всей литературы, подобранной для раскрытия данной темы. Бывает, что какая-либо тема рассматривается в нескольких главах или в разных местах книги. А в конспекте весь материал, относящийся к теме, будет сосредоточен в одном месте. В плане конспекта рекомендуется делать пометки, к каким источникам (вплоть до страницы) придется обратиться для раскрытия вопросов. Тематический конспект составляется обычно для того, чтобы глубже изучить определенный вопрос, подготовиться к докладу,

лекции или выступлению на семинарском занятии. Такой конспект по содержанию приближается к реферату, докладу по избранной теме, особенно если включает и собственный вклад в изучение проблемы.

4 Методические указания по подготовке к лабораторным работам

В рамках самостоятельной работы предусмотрена подготовка студента к лабораторным работам. Лабораторные работы позволяют интегрировать теоретические знания и формировать практические умения и навыки студентов в процессе учебной деятельности.

Цели лабораторных занятий по дисциплине «Системы обработки и хранения данных в электроэнергетике:

1) закрепление теоретического материала путем систематического контроля за самостоятельной работой студентов;

2) формирование умений использования теоретических знаний в процессе выполнения лабораторных работ;

3) развитие аналитического мышления путем обобщения результатов лабораторных работ;

4) формирование навыков оформления результатов лабораторных работ в виде таблиц, графиков, выводов.

На лаюораторных занятиях осуществляются следующие формы работ со студентами: индивидуальная (оценка знаний, выполненных тестовых заданий, проверка рабочих тетрадей); групповая (выполнение заданий малыми группами по 2-4 человека); фронтальная (подведение итогов выполнения лабораторных работ).

Список рекомендуемой литературы

Основная литература:

8.1.1. Перечень основной литературы:

1. Цехановский, В. В. Управление данными : учебник / В. В. Цехановский, В. Д. Чертовской. — Санкт-Петербург : Лань, 2022. — 432 с. — ISBN 978-5-8114-1853-4. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/212084

2. Макшанов, А. В. Системы поддержки принятия решений : учебное пособие для вузов / А. В. Макшанов, А. Е. Журавлев, Л. Н. Тындыкарь. — 2-е изд., стер. — Санкт-Петербург : Лань, 2021. — 108 с. — ISBN 978-5-8114-8489-8. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/176903

3. Кузнецов, С. Д. Введение в модель данных SQL : учебное пособие / С. Д. Кузнецов. — 3-е изд. — Москва : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2021. — 350 с. — ISBN 978-5-4497-0873-1. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт].

4. Кузнецов, С. Д. Введение в реляционные базы данных : учебное пособие / С. Д. Кузнецов. — 3-е изд. — Москва : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2021. — 247 с. — ISBN 978-5-4497-0902-8. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт].

Дополнительная литература:

1. Волк, В. К. Базы данных. Проектирование, программирование, управление и администрирование / В. К. Волк. — 4-е изд., стер. — Санкт-Петербург : Лань, 2023. — 244 с. — ISBN 978-5-507-47243-7. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/346439

2. Савельев, А. О. Проектирование и разработка веб-приложений на основе технологий Microsoft : учебное пособие / А. О. Савельев, А. А.

Алексеев. — 4-е изд. — Москва : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2022. — 418 с. — ISBN 978-5-4497-1650-7. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]

3. Советов, Б. Я. Информационные технологии: теоретические основы : учебное пособие / Б. Я. Советов, В. В. Цехановский. — 2-е изд., стер. — Санкт-Петербург : Лань, 2022. — 444 с. — ISBN 978-5-8114-1912-8. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/209876

4. Сычев, А. В. Теория и практика разработки современных клиентских веб-приложений : учебное пособие / А. В. Сычев. — 3-е изд. — Москва : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2021. — 482 с. — ISBN 978-5-4497-0943-1. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]

Учебно-методическая литература:

1. Методические указания к лабораторным работам по дисциплине «Системы обработки и хранения данных в электроэнергетике», СКФУ, 2024.

Интернет-ресурсы:

1. http://catalog.ncfu.ru – электронные каталоги Ассоциации электронных библиотек учебных заведений и организаций СКФО.

2. http://www.biblioclub.ru – электронно-библиотечная система «Университетская библиотека онлайн».

3. http://www.iprbookshop.ru/ – ЭБС «IPRbooks».