ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ

СЫКТЫВКАРСКИЙ ЛЕСНОЙ ИНСТИТУТ – ФИЛИАЛ ГОСУДАРСТВЕННОГО ОБРАЗОВАТЕЛЬНОГО УЧРЕЖДЕНИЯ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ «САНКТ-ПЕТЕРБУРГСКАЯ ГОСУДАРСТВЕННАЯ ЛЕСОТЕХНИЧЕСКАЯ АКАДЕМИЯ ИМЕНИ С. М. КИРОВА»

КАФЕДРА ИНФОРМАЦИОННЫХ СИСТЕМ

СИСТЕМЫ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ И ЯЗЫК SQL

САМОСТОЯТЕЛЬНАЯ РАБОТА СТУДЕНТОВ

Методические указания
для подготовки дипломированных специалистов
по направлению 654700 «Информационные системы»
специальности 230201 «Информационные системы и технологии»

СЫКТЫВКАР 2008

УДК 004
ББК 73
С40

Рассмотрены и рекомендованы к печати кафедрой информационных систем Сыктывкарского лесного института 14 ноября 2006 г. (протокол № 8).

Утверждены к печати методической комиссией технологического факультета Сыктывкарского лесного института 29 декабря 2006 г. (протокол № 3).

Составитель:
А. Н. Кирпичев, доцент

С40 **СИСТЕМЫ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ И ЯЗЫК SQL** : САМОСТОЯТЕЛЬНАЯ РАБОТА СТУДЕНТОВ : метод. указания для подготовки дипломированных специалистов по направлению 654700 «Информационные системы» спец. 230201 «Информационные системы и технологии» / сост. А. Н. Кирпичев ; СЛИ. – Сыктывкар, 2008. – 20 с.

УДК 004
ББК 73

В издании приведены сведения о дисциплине «Системы управления базами данных и язык SQL», ее целях, задачах, содержании, месте в учебном процессе. Помещены рекомендации по самостоятельной подготовке студентов и контролю их знаний. Дан список рекомендуемой литературы.

Для студентов специальности 230201 «Информационные системы и технологии».

* * *

Учебное издание

Составитель Кирпичев Алексей Николаевич

СИСТЕМЫ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ И ЯЗЫК SQL

САМОСТОЯТЕЛЬНАЯ РАБОТА СТУДЕНТОВ

Методические указания для подготовки дипломированных специалистов по направлению 654700 «Информационные системы» специальности 230201 «Информационные системы и технологии»

Сыктывкарский лесной институт – филиал государственного образовательного учреждения высшего профессионального образования «Санкт-Петербургская государственная лесотехническая академия имени С. М. Кирова» (СЛИ)
167982, г. Сыктывкар, ул. Ленина, 39
institut@sfi.komi.com, www.sli.komi.com

Подписано в печать 14.01.08. Формат 60 × 90 1/16. Усл. печ. л. 1,2. Тираж 10. Заказ № .

Редакционно-издательский отдел СЛИ.
Отпечатано в типографии СЛИ

© А. Н. Кирпичев, составление, 2008
© СЛИ, 2008

ОГЛАВЛЕНИЕ

1. ЦЕЛИ И ЗАДАЧИ ДИСЦИПЛИНЫ, ЕЕ МЕСТО В УЧЕБНОМ ПРОЦЕССЕ 4
1.1. Цель преподавания дисциплины 4
1.2. Задачи изучения дисциплины 4
1.3. Перечень дисциплин и тем, усвоение которых студентами необходимо для изучения данной дисциплины 4
1.4. Дополнение к нормам государственного стандарта 4
2. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ 5
2.1. Самостоятельная работа и контроль успеваемости 5
2.2. Распределение часов по темам и видам занятий 6
3. РЕКОМЕНДАЦИИ ПО САМОСТОЯТЕЛЬНОЙ ПОДГОТОВКЕ СТУДЕНТОВ 6
3.1. Методические рекомендации по самостоятельному изучению тем 6
3.2. Методические рекомендации по самостоятельной подготовке к лабораторным работам 8
4. КОНТРОЛЬ ЗНАНИЙ СТУДЕНТОВ 10
4.1. Промежуточный контроль 10
4.2. Итоговый контроль 19
БИБЛИОГРАФИЧЕСКИЙ СПИСОК 20

1. ЦЕЛИ И ЗАДАЧИ ДИСЦИПЛИНЫ, ЕЕ МЕСТО В УЧЕБНОМ ПРОЦЕССЕ

1.1. Цель преподавания дисциплины

Целью преподавания дисциплины является освоение студентами практического использования языка структурированных запросов SQL.

1.2. Задачи изучения дисциплины

Задача изучения дисциплины состоит в том, чтобы студенты овладели основами теоретических и практических знаний в области написания запросов к реляционным базам данных и изучении существующих на рынке РСУБД.

В результате изучения дисциплины студент должен:

ЗНАТЬ теоретические основы реляционных баз данных (РБД), синтаксис предложений языка SQL, существующие на рынке РСУБД;

УМЕТЬ писать на языке SQL запросы средней и высокой степени сложности:

- выборку данных SELECT;
- обновление UPDATE;
- вставку INSERT;
- удаление DELETE;
- управление доступом GRANT, REVOKE;
- создание и изменение БД и реляционных таблиц CREATE | ALTER TABLE | DATEBASE;

БЫТЬ ОЗНАКОМЛЕНЫ:

- со средствами визуального проектирования SQL (QBE бланки);
- с механизмами управления доступом и безопасности на примере РСУБД MySQL.

1.3. Перечень дисциплин и тем, усвоение которых студентами необходимо для изучения данной дисциплины

Для полноценного усвоения учебного материала необходимо иметь прочные знания по предметам «Управление данными», «Алгебре» (раздел Реляционная алгебра), «Информатике» и «Математической логике».

1.4. Дополнение к нормам государственного стандарта

Язык SQL, история развития, составные части, предложение, метаязык описания синтаксиса, синтаксис предложений манипулирования данными, модификации и управления доступом, РСУБД, безопасность и санкционирование доступа, регулярные выражения.

2. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

Дисциплина «Системы управления базами данных и язык SQL» изучается в 3 семестре. Запланировано 18 лекционных и 16 лабораторных часов.

2.1. Самостоятельная работа и контроль успеваемости

Самостоятельная работа студента по видам учебных занятий распределяется следующим образом:

• проработка лекционного материала по конспекту и учебной литературе;

• подготовка к практическим и лабораторным занятиям.

Текущая успеваемость студентов контролируется опросами по результатам выполнения лабораторных работ и проведением контрольных и тестовых работ. Итоговая успеваемость определяется на зачете.

Очная форма обучения

Вид самостоятельной работы	Кол-во часов	Вид контроля успеваемости
1. Проработка лекционного материала по конспекту, учебной литературе и специализированным ресурсам Internet	9	КР, ТФП Зачет – 5 семестр
2. Подготовка к практическим занятиям	8	КО, КР, ТФП
3. Изучение средств администрирования СУБД Access	4	КР, ТФП Зачет – 5 семестр
4. Практическое использование предложений CREATE TABLE, CREATE DATEBASE, (ALTER и DROP)	7	КР, ТФП Зачет – 5 семестр
5. Подготовка к зачету	4	Зачет – 5 семестр
Всего часов	32	

Очно-заочная форма обучения

Вид самостоятельной работы	Кол-во часов	Вид контроля успеваемости
1. Проработка лекционного материала по конспекту, учебной литературе и специализированным ресурсам Интернет	5	КР, ТФП Зачет – 7 семестр
2. Подготовка к практическим занятиям	5	КО, КР, ТФП
3. Изучение средств администрирования СУБД Access	4	КР, ТФП Зачет – 7 семестр
4. Практическое использование предложений CREATE TABLE, CREATE DATEBASE, (ALTER и DROP)	7	КР, ТФП Зачет – 7 семестр
5. Проработка материала по темам, не рассмотренным на лекциях	8	КР, ТФП Зачет – 7 семестр
6. Выполнение 6 и 7 лабораторной работы	6	Зачет – 7 семестр
7. Подготовка к зачету	11	Зачет – 7 семестр
Всего часов	46	

Текущая успеваемость студентов контролируется защитой отчетов по лабораторным работам, проведением промежуточной аттестации в виде тестирования.

Требования к зачету: выполнение и защита лабораторных работ; положительная оценка за промежуточную тестовую работу; выполнение итоговой зачетной работы.

2.2. Распределение часов по темам и видам занятий

Очное форма обучения

Наименование темы дисциплины	Объем работы, ч				Вид контроля успеваемости
	лекции	ЛР	СР	всего	
1. Введение	2	0	1	3	КО, КР
2. Синтаксис предложения SELECT и практическое применение предложения	4	4	4	12	КО, КР
3. Предложения модификации данных	2	5	3,5	10,5	КО, КР
4. Предложения управления доступом	2	0	1	3	КО, КР
5. Использование вложенных запросов. Соединения и объединение	2	5	3,5	10,5	КО, КР
6. Создание таблиц и баз данных модификация таблиц	4	0	9	13	КО, КР
7. РСУБД MySQL. Использование регулярных выражений	2	2	2	6	КО, КР
8. Изучение средств администрирования СУБД Access	0	0	4	4	КО, КР
Подготовка к зачету	0	0	4	4	Зачет
Всего	18	16	32	66	

Очно-заочная форма обучения

Наименование темы дисциплины	Объем работы, ч				Вид контроля успеваемости
	лекции	ЛР	СР	всего	
1. Введение	2	0	1	3	КО, КР
2. Синтаксис предложения SELECT и практическое применение предложения	3	4	5	12	КО, КР
3. Предложения модификации данных	2	4	3	9	КО, КР
4. Предложения управления доступом	1	0	2	3	КО, КР
5. Использование вложенных запросов. Соединения и объединение	2	2	4	8	КО, КР
6. Создание таблиц и баз данных модификация таблиц	0	0	10	10	КО, КР
7. РСУБД MySQL. Использование регулярных выражений	0	0	6	6	КО, КР
8. Изучение средств администрирования СУБД Access	0	0	4	4	КО, КР
Подготовка к зачету	0	0	11	11	Зачет
Всего	10	10	46	66	

3. РЕКОМЕНДАЦИИ ПО САМОСТОЯТЕЛЬНОЙ ПОДГОТОВКЕ СТУДЕНТОВ

3.1. Методические рекомендации по самостоятельному изучению тем

Самостоятельная работа студентов по изучению отдельных тем дисциплины включает поиск учебных пособий по данному материалу, проработку и анализ теоретического материала, самоконтроль знаний по данной теме с помощью нижеприведенных контрольных вопросов и заданий.

Темы дисциплины	Контрольные вопросы и задания
1	*2*
1. Введение	**Знать:** ▪ 12 правил Кодда, относящиеся к РСУБД ▪ историю развития языка SQL ▪ расширения языка SQL ▪ стандарты SQL ▪ составляющие языка, назначение

Окончание таблицы

1	*2*
2. Синтаксис предложения SELECT и практическое применение предложения	**Знать:** ▪ мета язык описания синтаксиса ▪ состав и содержание предложения SELECT ▪ синтаксис предложения **Уметь использовать:** ▪ различные WHERE условия ▪ квалификаторы IN, AS. ▪ группировку данных и отбор групп. (GROUP BY и HAVING фразы). ▪ предложение для нескольких таблиц.
3. Предложения модификации данных	**Знать:** ▪ Особенности использования предложения INSERT при вставке неполных данных и при вставке нескольких записей. ▪ Синтаксис предложений модификации. **Уметь использовать предложения:** ▪ DELETE ▪ INSERT ▪ UPDATE
4. Предложения управления доступом	**Знать:** ▪ Понятие привилегии. ▪ Синтаксис и практическое применение предложений GRANT и REVOKE ▪ Инструментальные средства СУБД и хранение привилегий на примере СУБД MySQL **Уметь использовать:** ▪ Привилегии. ▪ Инструментальные средства СУБД и хранение привилегий на примере СУБД MySQL.
5. Использование вложенных запросов. Соединения и объединение	**Уметь использовать:** ▪ вложенные запросов в WHERE фразе предложений SELECT, UPDATE и DELETE. ▪ запросы при построении подзапросов в качестве элементов выражения. ▪ UNION и JOIN (LEFT, RIGHT, FULL, INNER).
6. Создание таблиц и баз данных модификация таблиц	**Знать:** ▪ Типы SQL-совместимых данных. ▪ Отличия char() и varchar(). ▪ Расширения по типам данных для различных РСУБД. ▪ Синтаксис CREATE TABLE, CREATE DATEBASE, ALTER TABLE, DROP TABLE, DROP DATEBASE. **Уметь использовать:** ▪ множества, логические типы, поля autoincrament, zerofield. ▪ предложения: CREATE TABLE, CREATE DATEBASE, ALTER TABLE, DROP TABLE, DROP DATEBASE. ▪ Значения по умолчанию.
7. РСУБД MySQL. Использование регулярных выражений	**Знать:** ▪ Общие понятия СУБД MySQL. ▪ Назначение система управления PHPMyAdmin. ▪ Понятие регулярного выражения. ▪ Синтаксис модификатора REGEX. **Уметь использовать:** ▪ Регулярные выражения в запросах.
8. Изучение средств администрирования СУБД Access	**Знать:** ▪ Средства администрирования СУБД. ▪ Способы задания привилегий. ▪ Способы защиты БД от несанкционированного доступа.

Рекомендуемая литература. Все темы: [2], [5], [7], [8], по теме 1: [4]; по теме 7: [3], [14], [15], [16].

3.2. Методические рекомендации по самостоятельной подготовке к лабораторным работам

Согласно учебному плану специальности, на проведение лабораторных работ отводится 16 часов по очной форме обучения и 10 часов по очно-заочной форме обучения.

Самостоятельная работа студентов по подготовке к лабораторным работам, оформлению отчетов и защите лабораторных работ включает проработку и анализ теоретического материала, описание проделанной экспериментальной работы с приложением таблиц, запросов, а и также самоконтроль знаний по теме лабораторной работы с помощью нижеприведенных контрольных вопросов и заданий.

Темы лабораторных работ	**Контрольные вопросы и задания**
1	***2***
1.Использование предложения SELECT для одной таблицы	▪ Что такое мета-язык описания синтаксиса? Для чего он нужен? ▪ Синтаксис предложения SELECT ▪ Правила написания WHERE фраз и WHERE условий ▪ Для чего необходимы агрегатные функции? Когда их можно применять? ▪ Особенности использования функции AVG() ▪ Функции работы с датой/временем ▪ Что может быть в качестве элемента Select? ▪ Как записываются несколько элементов Select? ▪ Каким образом соединяются несколько WHERE условий? ▪ Правила составления выражений ▪ Для чего нужен предикат LIKE? ▪ Специальные символы для предиката LIKE ▪ Когда используется предикат IN? ▪ Как записываются элементы списка (множества)? ▪ Где можно использовать предикат AS? ▪ Может ли быть в качестве элемента Select константа? ▪ Что такое NULL? ▪ Для чего нужны ключи и индексы? ▪ Как хранятся ключи и индексы?
2.Использование предложений модификации данных DELETE и UPDATE	▪ Что такое мета-язык описания синтаксиса? Для чего он нужен? ▪ Синтаксис предложения DELETE ▪ Синтаксис предложения UPDATE ▪ Правила написания WHERE фраз и WHERE условий ▪ Может ли отсутствовать WHERE условие? ▪ Что произойдет при выполнении запроса DELETE FROM Таблица; ? ▪ Для чего необходимы агрегатные функции? Когда их можно применять? ▪ Функции работы с датой/временем ▪ Как записываются несколько полей в фразе SET? ▪ Каким образом соединяются несколько WHERE условий? ▪ Правила составления выражений ▪ Для чего нужен предикат LIKE? ▪ Специальные символы для предиката LIKE ▪ Когда используется предикат IN? ▪ Как записываются элементы списка (множества)? ▪ Где можно использовать предикат AS? ▪ Может ли быть в качестве элемента Select константа? ▪ При каких условиях запрос на модификацию данных не произведет изменений?

Окончание таблицы

1	*2*
3. Использование предложения SELECT при работе с несколькими таблицами	▪ Что такое мета-язык описания синтаксиса? Для чего он нужен? ▪ Синтаксис предложения SELECT ▪ Что такое подзапрос? ▪ Где можно применять подзапросы? ▪ Как можно обойтись без подзапросов? ▪ Для чего необходим UNION? ▪ Для чего необходимы агрегатные функции? Когда их можно применять? ▪ Как записывать несколько таблиц после FROM? ▪ Как нужно указывать названия полей при выборке данных из нескольких таблиц? ▪ Что может быть в качестве элемента Select? ▪ Как записываются несколько элементов Select? ▪ Каким образом соединяются несколько WHERE условий? ▪ Правила составления выражений ▪ Для чего нужен предикат LIKE? ▪ Специальные символы для предиката LIKE ▪ Когда используется предикат IN? ▪ Как записываются элементы списка (множества)? ▪ Где можно использовать предикат AS? ▪ Может ли быть в качестве элемента Select константа? ▪ Как правильно можно записать элемент Select при выборке данных из двух таблиц, если в результате нужно показать только поля одной таблицы?
4. Выполнение сложных запросов на вставку данных и на выборку	▪ Что такое мета-язык описания синтаксиса? Для чего он нужен? ▪ Синтаксис предложения SELECT ▪ Синтаксис предложения INSERT ▪ Что такое подзапрос? ▪ Где можно применять подзапросы? ▪ Как можно обойтись без подзапросов? ▪ Для чего необходим UNION? ▪ Что такое соединение? ▪ Отличие полного соединения от правого и левого. Привести примеры. ▪ Для чего необходимы агрегатные функции? Когда их можно применять? ▪ Как записывать несколько таблиц после FROM? ▪ Как нужно указывать названия полей при выборке данных из нескольких таблиц? ▪ Когда используется предикат IN? ▪ Как записываются элементы списка (множества)? ▪ Где можно использовать предикат AS? ▪ Может ли быть в качестве элемента Select константа? ▪ Как правильно можно записать элемент Select при выборке данных из двух таблиц, если в результате нужно показать только поля одной таблицы? ▪ Правила объединения результатов двух и более запросов ▪ Правила работы с предложением INSERT.
5. Использование визуального инструментария РСУБД MySQL на примере PHPMyAdmin	▪ Что такое привилегия? ▪ Для чего нужен PHPMyAdmin? ▪ Особенность получения прав доступа с помощью PHPMyAdmin. ▪ Получение статистики работы СУБД ▪ Синтаксис предложений создания таблиц и баз данных ▪ Поддерживаемые языки и кодировки.

Рекомендуемая литература: [1], [6], [7], [8], [9], [10], [13], [14].

4. КОНТРОЛЬ ЗНАНИЙ СТУДЕНТОВ

4.1. Промежуточный контроль

Текущая успеваемость студентов контролируется выполнением, оформлением и защитой отчетов по лабораторным работам, промежуточной аттестацией в виде тестирования. Тесты промежуточной аттестации включают: теоретический материал, пройденный на лекциях, практический материал по лабораторным работам.

Варианты тестовых заданий для промежуточного контроля

1. Определите команду для вызова функции, создаваемой следующей командой:

```
CREATE FUNCTION user1.my_func3(
@r VARCHAR(20),@t FLOAT)
RETURNS TABLE
AS
RETURN (SELECT Название, Тариф
        FROM Город
        WHERE Регион=@r AND Тариф<=@t
        ORDER BY Название)
-SELECT Название
 FROM user1.my_func3('Поволжье',3.50)

-SELECT Название, Тариф
 FROM my_func3(@r='Поволжье',@t=3.50)

-SELECT Название, Тариф
 FROM user1.my_func3
 WHERE @r='Поволжье'AND @t=3.50

-SELECT *
 FROM user1.my_func3(3.50,'Поволжье')
```

2. Необходимо разработать функцию для вывода списка пассажиров, купивших билеты в определенный день на некоторый рейс. Функция создается командой:

```
-CREATE FUNCTION user1.my_func(
 @r INT,@d DATETIME)
 RETURNS TABLE
 AS
 RETURN (SELECT Билет.Фамилия_пассажира
         FROM Билет
         WHERE Билет.Дата_продажи=@d AND
         Билет.Номер_рейса=@r)

-CREATE FUNCTION.my_func(@r INT,@d DATETIME)
 RETURNS TABLE
 AS
 RETURN (SELECT Билет.Фамилия_пассажира
         FROM Билет
```

```
        WHERE Билет.Дата_продажи=? AND
        Билет.Номер_рейса=?)
```

```
- CREATE FUNCTION user1.my_func
  (DECLARE @r INT,@d DATETIME)
  RETURNS TABLE
  AS
  RETURN (SELECT Билет.Фамилия_пассажира
          FROM Билет
          WHERE Билет.Дата_продажи=@d AND
          Билет.Номер_рейса=@r)
```

```
- CREATE FUNCTION user1.my_func(@r INT,@d DATETIME)
  RETURNS TABLE
  AS
  DECLARE @r INT,@d DATETIME
  RETURN (SELECT Билет.Фамилия_пассажира
          FROM Билет
          WHERE Билет.Дата_продажи=@d AND
          Билет.Номер_рейса=@r)
```

3. Необходимо разработать функцию для получения списка книг заданной тематики, изданных тиражом менее указанного числа. Определите команду для ее вызова.

```
- SELECT Название
  FROM user1.my_func(‘детектив’,100)
```

```
- SELECT Название
  FROM my_func(@t=’детектив’,@k=100)
```

```
- SELECT Название
  FROM user1.my_func
  WHERE @t=’детектив’ AND @k=100
```

```
- SELECT *
  FROM user1.my_func(100,’детектив’)
```

4. Даны таблицы

```
CREATE TABLE Город
(Код_Города INT ,
Название VARCHAR(20) NOT NULL,
Тариф MONEY,
Регион VARCHAR(20))
```

```
CREATE TABLE Разговор
Код_Разговора INT ,
Код_Города INT NOT NULL,
Фамилия IVARCHAR(20),
Дата DATETIME NOT NULL,
Продолжительность INT NOT NULL)
```

Перечислить фамилии людей, которые не говорили с абонентами из Москвы.

```
- SELECT DISTINCT Разговор.Фамилия
  FROM Разговор
  WHERE Разговор.Фамилия NOT IN
   (SELECT Разговор.Фамилия
  FROM Разговор INNER JOIN Город ON
   Разговор.Код_Города = Город.Код_Города
  WHERE Город.Название="Москва")
```

```
- SELECT DISTINCT Разговор.Фамилия
  FROM Разговор,Город
  WHERE Разговор.Код_Города = Город.Код_Города
  AND Город.Название<>"Москва"

- SELECT DISTINCT Разговор.Фамилия
FROM Разговор INNER JOIN Город ON
  Разговор.Код_Города = Город.Код_Города
WHERE Город.Название<>"Москва"

- SELECT DISTINCT Разговор.Фамилия
  FROM Разговор
  WHERE NOT EXISTS (SELECT Разговор.Фамилия
  FROM Разговор INNER JOIN Город ON
   Разговор.Код_Города = Город.Код_Города
  WHERE Город.Название="Москва")
```

5. Даны таблицы **Автор** и **Книга**.

```
CREATE TABLE Автор
(Код_Автора INT ,
 Фамилия VARCHAR(50))
```

```
CREATE TABLE Книга
(Код_Книги INT,
   Название        VARCHAR(50) NOT NULL,
   Цена            MONEY,
   Издательство    VARCHAR(50) NOT NULL,
   Код_Автора      INT NOT NULL,
   Количество      INT)
```

Определить авторов, чьи книги проданы в наименьшем количестве.

```
- SELECT Автор.Фамилия, Книга.Количество
  FROM Книга INNER JOIN Автор ON
   Книга.Код_Автора = Автор.Код_Автора
  WHERE Книга.Количество=
  (SELECT Min(Книга.Количество)
  FROM Книга)

- SELECT Автор.Фамилия, Книга.Количество
  FROM Книга INNER JOIN Автор ON
   Книга.Код_Автора = Автор.Код_Автора
  WHERE Книга.Количество <=
  ALL(SELECT Книга.Количество
  FROM Книга)

- SELECT Автор.Фамилия, Книга.Количество
  FROM Книга INNER JOIN Автор ON
   Книга.Код_Автора = Автор.Код_Автора
  WHERE Книга.Количество =
   Min(Книга. Количество)

- SELECT Автор.Фамилия, Книга. Количество
  FROM Книга INNER JOIN Автор ON
   Книга.Код_Автора = Автор.Код_Автора
  WHERE Книга.Количество =
  Min(SELECT Книга.Количество
  FROM Книга)
```

6. Определить город Поволжья, с которым дольше других ведутся телефонные разговоры.

```
- SELECT Город.Название,
   Sum(Разговор.Продолжительность)
  FROM Город INNER JOIN Разговор ON
   Город.Код_Города = Разговор.Код_Города
  WHERE Город.Регион='Поволжье'
  GROUP BY Город.Название
  HAVING Sum(Разговор.Продолжительность)>=All
   (SELECT Sum(Разговор.Продолжительность)
   FROM Город INNER JOIN Разговор ON
   Город.Код_Города = Разговор.Код_Города
   WHERE Город.Регион='Поволжье'
   GROUP BY Город.Название)

- SELECT Город.Название,
   Sum(Разговор.Продолжительность) AS
   Кол_разговоров
  FROM Город INNER JOIN Разговор ON
   Город.Код_Города = Разговор.Код_Города
  WHERE Город.Регион='Поволжье'
  GROUP BY Город.Название
  HAVING Count(Разговор.Продолжительность)>=MAX
   (SELECT Sum(Разговор.Продолжительность)
   FROM Город INNER JOIN Разговор ON
   Город.Код_Города = Разговор.Код_Города
   WHERE Город.Регион='Поволжье'
   GROUP BY Город.Название)

- SELECT Город.Название,
   Sum(Разговор. Продолжительность) AS
   Кол_разговоров
  FROM Город INNER JOIN Разговор ON
   Город.Код_Города = Разговор.Код_Города
  WHERE Город.Регион='Поволжье'
  HAVING Sum(Разговор.Код_Разговора)>=ALL
   (SELECT Sum(Разговор.Продолжительность)
   FROM Город INNER JOIN Разговор ON
   Город.Код_Города = Разговор.Код_Города
   WHERE Город.Регион='Поволжье')

- SELECT Город.Название,
   Sum(Разговор. Продолжительность) AS
   Кол_разговоров
  FROM Город INNER JOIN Разговор ON
   Город.Код_Города = Разговор.Код_Города
  WHERE Город.Регион='Поволжье'
  AND Sum(Разговор.Продолжительность)>=ALL
   (SELECT Sum(Разговор.Продолжительность)
   FROM Город INNER JOIN Разговор
   ON Город.Код_Города = Разговор.Код_Города
   WHERE Город.Регион='Поволжье'
   GROUP BY Город.Название)
```

7. В базе данных basa_user зарегистрировано два пользователя: UserA и UserB. Какие действия предпримет система?

Basa_user Администратор	UserA	UserB
GRANT ALL TO UserB		CREATE TABLE tab (id_t INT PRIMARY KEY, nam_t VARCHAR(20))
INSERT INTO tab values (1,'aa') ?	INSERT INTO tab values (2,'bb') ?	INSERT INTO tab values (3,'cc') ?

- Выполнено Выполнено Отклонено
- Отклонено Выполнено Выполнено
- Отклонено Отклонено Выполнено
- Выполнено Выполнено Выполнено

8. Пусть номер страхового пенсионного свидетельства состоит из 11 цифр, разделенных знаками "–", например: 063-840-013-82. Укажите правильный тип данных для его описания.

- Пенс_свидетельство CHAR(14)
- Пенс.свид. CHAR(14)
- Пенс_свид FLOAT
- Pens_svid INT(11)

9. Какие из следующих описаний типов являются неправильными?

- DECIMAL(12,14)
- DECIMAL(8,8)
- DECIMAL(6,3)
- DECIMAL(40,20)

10. Какие из следующих строк можно внести в поле, описанное как VARCHAR(6)?

- ‘Привет’
- ‘Здравствуйте’
- Ура
- ‘123456’

11. Что представляют собой базы данных?

- аппаратные средства для хранения данных
- набор логически связанных данных
- программные средства управления данными
- компьютер с хранящимися на нем данными

12. Назовите отличительные черты реализации от стандарта языка.

- любая реализация языка является подмножеством стандарта
- стандарт – это подмножество любой реализации языка
- реализация является дополнением и усовершенствованием команд и опций стандарта языка
- реализация языка заключается в адаптации стандарта языка к конкретной СУБД

13. Даны таблицы

```
CREATE TABLE Блюдо
(Название_блюда VARCHAR(20) NOT NULL,
 Время_приготовления INT NOT NULL,
 Общая_калорийность INT NOT NULL,
 Номер_рецепта INT,
 Повар VARCHAR(20),
 Стоимость MONEY)
```

```
CREATE TABLE Компонент
(Название_компонента VARCHAR(20),
Калорийность INT NOT NULL,
     Вес FLOAT,
Белки INT,
Блюдо VARCHAR(20),
Углеводы INT,
Стоимость MONEY NOT NULL)
```

Рассчитать общую стоимость компонентов блюда и калорийность каждого компонента.

```
-SELECT Название_компонента,
  Стоимость*Вес AS Общ_Стоимость,
   Калорийность*Вес AS Общ_калорийность
FROM Компонент

-SELECT Название_компонента,
  Стоимость*Вес, Калорийность*Вес
FROM Компонент

-SELECT Название_компонента,
 Общ_Стоимость =Стоимость*Вес,
  Общ_калорийность =Калорийность*Вес
FROM Компонент

-SELECT Название_компонента,
 @s=Стоимость*Вес, @k=Калорийность*Вес
FROM Компонент
```

14. Даны таблицы

```
CREATE TABLE Автор
(Код_Автора INT ,
Фамилия VARCHAR(50) NULL,
Имя VARCHAR(50) NULL,
Отчество VARCHAR(50) NULL,
Пол VARCHAR(50) NOT NULL ,
Дата_рождения DATETIME ,
 Телефон CHAR(9))
```

```
CREATE TABLE Книга
 (Код_Книги INT,
Название VARCHAR(50) NOT NULL,
Цена MONEY,
Тематика VARCHAR(50) NOT NULL,
Издательство VARCHAR(50) NOT NULL,
Код_Автора INT NOT NULL,
Количество INT)
```

На какую сумму были проданы книги каждого автора?

```
-SELECT Автор.Фамилия,
 Sum(Книга.Цена*Книга.Количество) AS Стоимость
FROM Книга INNER JOIN Автор ON
 Книга.Код_Автора = Автор.Код_Автора
GROUP BY Автор.Фамилия

-SELECT Автор.Фамилия,
 Стоимость =Sum(Книга.Цена*Книга.Количество)
FROM Книга INNER JOIN Автор ON
 Книга.Код_Автора = Автор.Код_Автора
GROUP BY Автор.Фамилия

-SELECT Автор.Фамилия,
 Sum(Книга.Цена*Книга.Количество) AS Стоимость
FROM Книга INNER JOIN Автор ON
  Книга.Код_Автора = Автор.Код_Автора

-SELECT Автор.Фамилия,
 Sum(Цена* Количество) AS Стоимость
FROM Книга INNER JOIN Автор ON
 Книга.Код_Автора = Автор.Код_Автора
GROUP BY Автор.Фамилия
```

15. На какую сумму были проданы билеты на рейс до Москвы в день вылета?

```
- SELECT Sum(Билет.Стоимость) AS Общ_Стоимость,
  Билет.Дата_продажи
FROM Рейс INNER JOIN Билет ON
  Рейс.Номер_рейса = Билет.Номер_рейса
WHERE Билет.Дата_продажи=Рейс.Дата_вылета AND
  Рейс.Конечный_пункт='Москва'
GROUP BY Билет.Дата_продажи

- SELECT Sum(Билет.Стоимость) AS Общ_Стоимость,
  Билет.Дата_продажи
FROM Рейс INNER JOIN Билет ON
  Рейс.Номер_рейса = Билет.Номер_рейса
GROUP BY Билет.Дата_продажи
HAVING Билет.Дата_продажи=Рейс.Дата_вылета AND
  Рейс.Конечный_пункт='Москва'

- SELECT Sum(Билет.Стоимость) AS Общ_Стоимость
FROM Рейс INNER JOIN Билет ON
  Рейс.Номер_рейса = Билет.Номер_рейса
WHERE Билет.Дата_продажи=Рейс.Дата_вылета AND
  Рейс.Конечный_пункт='Москва'
GROUP BY Билет.Дата_продажи

- SELECT Билет.Дата_продажи,
  Общ_Стоимость =Sum(Билет.Стоимость)
FROM Рейс INNER JOIN Билет ON
  Рейс.Номер_рейса = Билет.Номер_рейса
WHERE Билет.Дата_продажи=Рейс.Дата_вылета AND
  Рейс.Конечный_пункт='Москва'
GROUP BY Билет.Дата_продажи
```

16. В базе данных в таблице Рейс необходимо хранить сведения о рейсах самолетов с указанием конечного пункта, даты вылета, продолжительности маршрута, количества мест в самолете и стоимости билета. Для описания данных используются следующие поля:

```
- Номер_рейса                  INT NOT NULL
  Конечный_пункт               VARCHAR(30)
  Дата_вылета                  DATETIME NOT NULL
  Продолжительность_маршрута   FLOAT
  Число_мест                   INT
  Стоимость_билета             MONEY

- Номер_рейса                  INT NOT NULL
  Конечный_пункт               VARCHAR(30)
  Дата_вылета                  DECIMAL(8,4) NOT NULL
  Продолжительность_маршрута   DATETIME
  Число_мест                   INT
  Стоимость билета             MONEY

- Номер_рейса                  INT NOT NULL
  Конечный_пункт               INT
  Дата_вылета                  DATETIME NOT NULL
  Продолжительность маршрута   DATETIME
```

```
Число_мест                    DECIMAL(3,6)
Стоимость_билета              INT

-Номер_рейса                  INT NOT NULL
Конечный_пункт                VARCHAR(30)
Дата_вылета                   DATETIME NOT NULL
Продолжительность_маршрута    DATETIME
Число_мест                    INT
Стоимость_билета              NUMERIC(3,4)
```

17. В таблицу Рейс необходимо добавить бортовой номер и увеличить длину поля Конечный_пункт.

```
-ALTER TABLE Рейс ADD борт CHAR(5)
 ALTER TABLE Рейс
      ALTER COLUMN Конечный_пункт VARCHAR(50)

-ALTER TABLE Рейс
      ADD COLUMN борт VARCHAR(3,1) NOT NULL
 ALTER TABLE Рейс DROP Конечный_пункт
 ALTER TABLE Рейс
      ADD COLUMN Конечный_пункт VARCHAR(50)

-ALTER TABLE Рейс ADD борт INT(5)
 ALTER TABLE Рейс
      ALTER Конечный пункт VARCHAR(50)

-ALTER TABLE Рейс
                 ADD COLUMN борт FLOAT(8,3)
 ALTER TABLE Рейс
      ALTER Конечный_пункт VARCHAR(50)
```

18.Создана таблица **Автор**. Изменить тип поля **Телефон** на символьный и заменить поле **Возраст** полем **Дата_рождения**.

```
 CREATE TABLE Автор
 (Фамилия   VARCHAR(50) NOT NULL,
 Пол        CHAR(3),
 Возраст    INT NOT NULL,
 Телефон    INT)
-ALTER TABLE Автор
            ALTER COLUMN Телефон CHAR(10)
ALTER TABLE Автор DROP Возраст
ALTER TABLE Автор
                 ADD Дата_рождения DATETIME

-ALTER TABLE Автор ALTER Телефон CHAR(10)
 ALTER TABLE Автор ALTER Возраст DATETIME

-ALTER TABLE Автор
                 ALTER COLUMN Телефон CHAR(10)
ALTER TABLE Автор DROP Возраст
ALTER TABLE Автор
            ADD COLUMN Дата_рождения DATETIME

-ALTER TABLE Автор ALTER Телефон CHAR(10)
ALTER TABLE Автор DROP Возраст
ALTER TABLE Автор
                 ADD Дата_рождения DATETIME
```

19. Даны таблицы **Город** и **Разговор**.

```
CREATE TABLE Город
(Код_Города CHAR(5),
Название VARCHAR(20) NOT NULL UNIQUE,
Тариф MONEY)
```

```
CREATE TABLE Разговор
(Дата DATETIME NOT NULL,
Время Суток VARCHAR(5),
Продолжительность INT NOT NULL)
```

В таблице Разговор значения поля ВремяСуток ограничить значениями из заданного списка: день, вечер, ночь, утро.

```
- ALTER TABLE Разговор
  ADD CONSTRAINT ch1 CHECK(ВремяСуток IN
  ('утро', 'день', 'вечер', 'ночь'))

- ALTER TABLE Разговор
  ADD CHECK(ВремяСуток IN (
  'утро', 'день', 'вечер', 'ночь'))

- ALTER TABLE Разговор
  ALTER ВремяСуток VARCHAR(20) CHECK(
  время_суток IN ('утро', 'день', 'вечер', 'ночь'))

- ALTER TABLE Разговор
  ADD CHECK(ВремяСуток ='утро' OR
   'день' OR 'вечер' OR ‘ночь’)
```

20. Даны таблицы **Автор** и **Книга**.

```
CREATE TABLE Автор
(Паспорт INT,
Фамилия VARCHAR(50) NULL,
Имя VARCHAR(50) NULL,
Отчество VARCHAR(50) NULL,
Пол CHAR(7),
Возраст INT ,
Телефон INT)
```

```
CREATE TABLE Книга
 (Название VARCHAR(50),
Цена MONEY,
Тематика VARCHAR(50),
Издательство VARCHAR(50),
Количество INT)
```

Для поля Пол таблицы Автор создать и применить правило для ввода только значений «Мужской» и «Женский».

```
- CREATE RULE sex AS @p IN ('Мужской','Женский')
  sp_bindrule 'sex', 'Автор.Пол'

- CREATE RULE sex AS @p='Мужской' OR @p='Женский'
  sp_bindrule 'sex', 'Автор.Пол'

- CREATE RULE sex AS @p='Мужской' OR @p='Женский'
  sp_bindrule 'sex', 'Пол'

- CREATE RULE sex (Автор.Пол IN (
  'Мужской','Женский'))
  sp_bindrule 'sex', 'Автор.Пол'
```

4.2. Итоговый контроль

Итоговый контроль включает зачет.

Требования к зачету:

1. Выполнение и защита лабораторных работ.
2. Положительная оценка за промежуточное тестирование.
3. Выполнение итоговой зачетной работы.

Вопросы к зачету

1. 12 правил Кодда, относящиеся к РСУБД
2. Реляционная СУБД
3. Архитектуры ИС
4. Модели данных
5. Примеры РСУБД и их отличительные особенности
6. История развития языка SQL
7. Расширения языка SQL, стандарты.
8. Составляющие языка, назначение.
9. Мета язык описания синтаксиса предложений
10. Состав и содержание предложения SELECT.
11. Синтаксис предложения SELECT
12. Примеры различных WHERE условий, использование квалификаторов IN, AS.
13. Группировка данных и отбор групп. (GROUP BY и HAVING фразы).
14. Использование предложения для нескольких таблиц.
15. Сортировка данных, ограничение вывода записей по числу
16. Предложение DELETE, INSERT, UPDATE: синтаксис и практическое применение.
17. Особенности использования предложения INSERT при вставке неполных данных и при вставке нескольких записей.
18. Понятие привилегии, работа с привилегиями.
19. Синтаксис и практическое применение предложений GRANT и REVOKE.
20. Инструментальные средства СУБД и хранение привилегий на примере СУБД MySQL
21. Использование вложенных запросов в WHERE фразе предложений SELECT, UPDATE и DELETE.
22. Использование запросов при построении подзапросов в качестве элементов выражения.
23. Использование UNION и JOIN (LEFT, RIGHT, FULL, INNER) примеры.
24. Типы SQL-совместимых данных. Отличия char() и varchar()
25. Использование множеств, логических типов, поля autoincrament, zerofield.
26. Расширения по типам данных для различных РСУБД.

27. Предложения CREATE TABLE, CREATE DATEBASE: синтаксис и примеры практическоко применения. Значения по умолчанию. Использование различных кодировок

28. ALTER TABLE, DROP TABLE, DROP DATEBASE: синтаксис и примеры практического применения

29. СУБД MySQL.

30. Система управления PHPMyAdmin

31. Понятие регулярного выражения.

32. Применение регулярных выражений в запросах. Модификатор REGEX.

33. Средства администрирования СУБД Access

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Упражнения по SQL. [Электронный ресурс] / 2000–2006. – Режим доступа: http: //sql-ex.ru. – Загл. с экрана.
2. CITFORUM.RU. Сервер информационных технологий. (Раздел базы данных) [Электронный ресурс]. – Режим доступа: http://www.citforum.ru. – Загл. с экрана.
3. MySQL.Ru. Лучший ресурс для разработчиков БД [Электронный ресурс] / 2001–2006. – Режим доступа: http://mysql.ru. – Загл. с экрана.
4. Русская Википедия [Электронный ресурс] / 1990–2006. – Режим доступа: http://ru.wikipedia.org
5. Все про SQL и клиент/серверные технологии. [Электронный ресурс]. – Режим доступа: http://www.sql.ru. – Загл. с экрана.
6. *Пачеко, К. А.* Использование SQL запроса и роль SQL в технологии создания клиент/сервер и работы с базой данных [Текст] / К. А. Пачеко. – СПб. : 2001.
7. *Хернандес, М. Дж.* SQL – запросы для простых смертных [Текст] : практ. руководство по манипулированию данными в SQL / М. Дж. Хернандес, Дж. Л. Вьескас ; пер. с англ. А. Головко. – М. : Лори, 2003.
8. Энциклопедия SQL [Текст] / Дж. Гроаф, П. Вайнберг. – СПб. : Питер, 2003.
9. SQL [Текст] : полное руководство : пер. с англ. – 2-е изд., перераб. и доп. – СПб. : BHV, 2001. – 816 с.
10. *Груббер, М.* Понимание SQL [Текст] / М. Грубер. – М. : Вильямс, 1993.
11. *Райордан, Р.* Основы реляционных баз данных [Текст] / Р. Райордан. – М. : Русская редакция, 2001.
12. *Коннолли, Т.* Базы данных. Проектирование, реализация и сопровождение. Теория и практика [Текст] / Т. Коннолли [и др.]. – М. : Вильямс, 2003.
13. *Пачеко, К. А.* Использованием SQL запроса и роль SQL в технологии создания технологии клиент/сервер и работы с базой данных [Текст] / К. А. Пачеко. – СПб. : 2001.
14. SourceForge phpMyAdmin project page [Электронный ресурс]. – Режим доступа: http://www.phpmyadmin.net. – Загл. с экрана.
15. MySQL AB Официальный сайт [Электронный ресурс] / 2000–2006. – Режим доступа: http://www.mysql.com. – Загл. с экрана.
16. Русское руководство phpMyAdmin. [Электронный ресурс]. – Режим доступа: http://www.phpmyadmin.ru. – Загл. с экрана.
17. *Базиян, М.* Использование Visual FoxPro 6 [Текст] : полное справочное руководство : пер. с англ. / М. Базиян [и др.]. – М. : Вильямс, 1999.
18. Офицальный сайт СУБД PostgreSQL [Электронный ресурс]. – Режим доступа: http://www.postgresql.org. – Загл. с экрана.