ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ

Федеральное государственное
образовательное бюджетное учреждение
высшего профессионального образования
«САНКТ-ПЕТЕРБУРГСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ
им. проф. М.А. БОНЧ-БРУЕВИЧА»

П.С. Зернов

БАЗЫ ДАННЫХ

Методические указания к лабораторным работам

СПб ГУТ)))

САНКТ-ПЕТЕРБУРГ
2012

УДК 004.65(075.8)
ББК 32.973.26–018.2я73
З58

Рецензент
кандидат технических наук, доцент СПбГЭТУ
А.Ю. Волков

Рекомендовано к печати
редакционно-издательским советом СПбГУТ

Зернов, П.С.
З58 Базы данных: методические указания к лабораторным работам / *П.С. Зернов.* – СПб. : Изд-во СПбГУТ, 2012. – 36 с.

Методические указания предназначены для студентов 3-го курса по направлению «Программная инженерия», профилю «Разработка программного обеспечения инфокоммуникационных сетей и систем», дисциплине «Базы данных». Содержат девять работ, составленных в порядке усвоения основных понятий языка SQL, понимания устройства СУБД «MySQL». Каждая работа содержит краткую постановку задачи, рекомендации по ее выполнению и контрольные вопросы.

УДК 004.65(075.8)
ББК 32.973.26–018.2я73

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

MySQL – система управления базами данных (СУБД) с открытым кодом. Это высокопроизводительная и масштабируемая СУБД с множеством программных интерфейсов. Она обладает огромными функциональными возможностями и подходит для решения самых разных задач.

Методические указания призваны помочь студентам, изучающим дисциплину «Базы данных», освоить его с практической стороны в ходе выполнения лабораторных работ. В первой и второй работах рассматриваются возможности языка SQL по конструированию запросов на выборку, обновление, удаление и добавление данных. Изучаются вопросы внутреннего и внешнего объединения таблиц. В работах с третьей по пятую исследуются возможности создания хранимых процедур, встроенных функции и транзакций. В работе номер шесть рассматриваются курсоры MySQL. Седьмая работа посвящена обеспечению целостности данных и триггерам. В работах номер восемь и девять студентам предстоит создать не только базу данных MySQL, но и разработать на языке PHP скрипт, позволяющий просматривать и редактировать данные через веб-браузер. Таким образом, в заключительных работах студентам предстоит применить весь спектр знаний, полученных в ходе изучения дисциплины, для создания веб-интерфейса к собственной базе данных MySQL.

ПОРЯДОК ВЫПОЛНЕНИЯ ЛАБОРАТОРНЫХ РАБОТ

1. Подготовка к лабораторной работе, посредством чтение лекционных материалов по соответствующей теме.

2. Выполнение очередной лабораторной работы на персональном компьютере (ПК) в классе согласно последовательности пунктов задания и оформление черновика.

3. Оформление отчета по результатам выполненной лабораторной работы согласно требованиям;

4. Защита лабораторной работы.

Отчет должен содержать следующие материалы:

- ☞ титульный лист с названием лабораторной работы, указанием группы и фамилией студента;
- ☞ цель работы;
- ☞ используемые в ходе работы SQL команды и результаты их выполнения в виде таблиц;
- ☞ выводы.

Для получения зачета студент должен продемонстрировать понимание теоретического материала и подкрепить свои знания ссылками на результаты в отчете лабораторной работы.

ЛАБОРАТОРНАЯ РАБОТА 1
SQL ЗАПРОСЫ В MYSQL

Цель работы – получение практических навыков создания запросов на выборку, обновление и удаление строк в таблице базы данных на языке SQL.

Задание

1. Создать базу данных «university» в программе-дизайнере MySQL Workbench.
2. В базе данных «university» создать таблицу «students» с полями:
   - ☞ id тип int – ключ (PK), счетчик (AI);
   - ☞ name тип varchar, ненулевое (NN);
   - ☞ d_id тип int.
3. Заполнить таблицу «students» произвольными записями (вкладка Inserts) - 5 строк (поле id следует заполнять нулями).
4. Сохранить созданную в программе-дизайнере схему базы данных на локальный компьютер.
5. Запустить генерацию базы данных на сервере MySQL (Пункт меню: Database->Forward Engineer. В опциях необходимо поставить галки против пунктов: DROP Objects Before Each CREATE Object и Generate INSERT Statements for Tables).
6. Подключиться к базе данных MySQL (команда mysql –u root –p).
7. Активизировать базу данных «university» (команда *use*).
8. Выполнить SQL команду: *SELECT * FROM students;* результаты записать в отчет.
9. Выполнить SQL команды:
   UPDATE students SET name = 'Ivan' WHERE id = 2;
   *SELECT * FROM students WHERE id = 2;* результаты записать в отчет.
10. Выполнить SQL команды:
    DELETE FROM students WHERE id = 2;
    *SELECT * FROM students;* результаты записать в отчет.
11. Проанализировать полученные результаты.

Справочные материалы

Для получения информации из таблиц базы данных используются запросы – SQL команды, начинающиеся с ключевого слова *SELECT*. Приведем пример наиболее простой команды, которая выводит все строки в таблице:

*SELECT * FROM <имя таблицы>;*

Для обозначения конца SQL запроса, по умолчанию в MySQL используется символ ‘;’. Не забывайте ставить его в конце каждого запроса. Если запрос состоит из нескольких строк, то переход на новую строку в

программе-клиенте командной строки MySQL осуществляется по нажатию клавиши Enter, а в конце последней строки также добавляется символ ‘;’.

Если необходимо выбрать из таблицы строки, которые удовлетворяют какому-либо критерию, то к SQL запросу добавляют команду *WHERE <условие отбора>.* Например, запрос:

*SELECT * FROM students WHERE name = 'Ivan';*

отобразит только те записи (строки) из таблицы students, в которых значение в столбце name соответствует имени 'Ivan'. Обратите внимание на то, что если в качестве условия отбора используется строковая константа, ее значение необходимо заключать в одинарные кавычки. Если в качестве условия задается число, то значение в кавычки не ставится.

Команда *UPDATE* позволяет установить новые значения в одной или нескольких строках таблицы. В упрощенном общем виде команду можно записать в следующем виде:

UPDATE <имя таблицы>
SET <имя столбца1>=<значение 1>
…
SET <имя столбцаN>=<значение N>
[WHERE <условие отбора>];

Обратите внимание, условие отбора помещено в квадратные скобки. Это означает, что команда *WHERE* не обязательно должна присутствовать в тексте SQL запроса. При наличии условия в запросе квадратные скобки не пишутся. Пример запроса на обновление строк:

UPDATE students SET name = 'Petr' WHERE name = 'Ivan';

После выполнения запроса во всех строках таблицы students, содержащих в столбце name значение ‘Ivan’, имя студента будет изменено на ‘Petr’.

Команда удаления строк в упрощенном общем виде выглядит следующим образом:

DELETE FROM <имя таблицы>
[WHERE <условие отбора>];

Если не указано условие, то после выполнения команды будут удалены все строки таблицы, иначе только соответствующие условию строки. Обратите внимание на то, что команда удаляет только строки таблицы, таблица из базы данных не удаляется.

Пример оформления отчета

Лабораторная работа 1

SQL запросы в MySQL

Цель работы – получение практических навыков создания запросов на выборку, обновление и удаление строк в таблице базы данных на языке SQL.

Результат выполнение команды: *SELECT * FROM students;*

id	**Name**	**d_id**
1	Egor	
2	Petr	
3	Nikolay	
4	Vasily	
5	Pavel	

Результат выполнение команд:
UPDATE students SET name = 'Ivan' WHERE id = 2;
*SELECT * FROM students WHERE id = 2;*

id	**Name**	**d_id**
1	Egor	
2	Ivan	
3	Nikolay	
4	Vasily	
5	Pavel	

Результат выполнение команд:
DELETE FROM students WHERE id = 2;
*SELECT * FROM students;*

id	**Name**	**d_id**
1	Egor	
3	Nikolay	
4	Vasily	
5	Pavel	

Выводы

Для извлечения информации из таблиц базы данных используются запросы, написанные на языке SQL. Команда *SELECT* создает запрос на выборку строк, при этом существует возможность указания условия их

отбора посредством инструкцией *WHERE*. Команда *UPDATE* осуществляет изменение/обновление данных, при этом если условие не указано, то изменения происходят во всех строках таблицы. Команда *DELETE* удаляет строки таблицы согласно условию. Если условие не указано, то таблица очищается, т.е. происходит удаление всех ее строк. Сама таблица не удаляется.

Контрольные вопросы

1. Какую команду нужно ввести в командной строке, чтобы получить доступ к MySQL? (Путь к каталогу bin, в который установлены исполняемые файлы MySQL, записан в системную переменную окружения PATH).
2. Напишите запрос, выводящий информацию обо всех студентах из таблицы students.
3. Напишите запрос, выводящий информацию только о студентах с именем ‘Petr’ из таблицы students.
4. Напишите запрос, выводящий список имен студентов из таблицы students.
5. Напишите запрос, изменяющий имя студента ‘Ivan’ на ‘Petr’ в таблице students.
6. Напишите запрос, удаляющий всех студентов с именем ‘Egor’ в таблице students.
7. Напишите запрос, очищающий таблицу students.

ЛАБОРАТОРНАЯ РАБОТА 2
ОБЪЕДИНЕНИЕ ТАБЛИЦ С ПОМОЩЬЮ ОПЕРАТОРА JOIN В MYSQL

Цель работы – получение практических навыков создания запросов на внутренние и внешние объединения таблиц на языке SQL.

Задание

1. Создать базу данных «university» в программе-дизайнере MySQL Workbench.

2. В базе данных «university» создать таблицу «departments» с полями:

- ☞ id тип int – ключ (PK);
- ☞ name тип varchar, ненулевое (NN).

3. Заполнить таблицу «departments» пятью произвольными записями (поле id следует заполнять уникальными целыми числами, например порядковыми номерами).

4. В базе данных «university» создать таблицу «users» с полями:

- ☞ id тип int – ключ (PK), счетчик (AI);
- ☞ name тип varchar, ненулевое (NN);
- ☞ d_id тип int.

5. Заполнить таблицу «users» произвольными семью записями (поле id следует оставить незаполненным), при этом:

- ☞ пять строк таблицы «users» должны содержать в поле d_id какое-либо уникальное число из поля id таблицы «departments»;
- ☞ две строки из таблицы «users» должны содержать в поле d_id числа, не используемые в строках таблицы «departments» в поле id.

6. Сохранить созданную в программе-дизайнере схему базы данных на локальный компьютер.

7. Запустить генерацию базы данных на сервере MySQL (Пункт меню: Database->Forward Engineer. В опциях необходимо поставить галки против пунктов DROP Objects Before Each CREATE Object и Generate INSERT Statements for Tables).

8. Подключиться к базе данных MySQL (команда mysql –u root –p).

9. Активизировать базу данных «university» (команда *use university*).

10. Последовательно выполнить следующие команды и записать в отчет команду и полученную на экране таблицу:

SELECT u.id, u.name, d.name FROM users u INNER JOIN departments d ON u.d_id = d.id;

SELECT u.id, u.name, d.name FROM users u LEFT JOIN departments d ON u.d_id = d.id;

SELECT u.id, u.name, d.name FROM users u RIGHT JOIN departments d ON u.d_id = d.id;

SELECT u.id, u.name, d.name FROM users u CROSS JOIN departments d;

11. Проанализировать полученные результаты.

Контрольные вопросы

1. Что такое декартово произведение таблиц?
2. Что такое левое объединение таблиц?
3. Что такое правое объединение таблиц?
4. Что такое внутреннее объединение таблиц?

ЛАБОРАТОРНАЯ РАБОТА 3
ХРАНИМЫЕ ПРОЦЕДУРЫ В MYSQL

Цель работы – получение практических навыков создания хранимых процедур и работы с переменными на языке SQL.

Задание

1. Создать базу данных «university» в программе-дизайнере MySQL Workbench.

2. В базе данных «university» создать таблицу «departments» с полями:

- ☞ id тип int – ключ (PK);
- ☞ name тип varchar, ненулевое (NN);

3. Заполнить таблицу «departments» произвольными записями - две строки (поле id следует заполнять уникальными целыми числами, например порядковыми номерами, а для поля «name» использовать название групп своего потока).

4. В базе данных «university» создать таблицу «users» с полями:

- ☞ id тип int – ключ (PK), счетчик (AI);
- ☞ name тип varchar, ненулевое (NN);
- ☞ d_id тип int;

5. Заполнить таблицу «users» произвольными записями - пять строк (insert), при этом: две записи из таблицы «users» связать с первой группой из таблицы «departments», а остальные записи связать со второй группой через поле id таблицы «departments» и d_id таблицы «users». Обязательно создать одну запись со значением в поле name – «Ivan».

6. Создать хранимую процедуру с именем *getUserInfo* (раздел Routines в MySQL Workbech).

7. Заполнить тело процедуры согласно образцу:

```
DELIMITER //
CREATE PROCEDURE `university`.`getUserInfo` (IN param1 VARCHAR(45), OUT param2 INT)
BEGIN
   DECLARE UserName VARCHAR(45);
   SET UserName = 'Ivan';

   IF param1 IS NOT NULL THEN
      SET UserName = param1;
   END IF;

    SELECT d.name AS departments INTO @department FROM users u LEFT JOIN departments d ON u.d_id = d.id WHERE u.name = UserName;
   SELECT COUNT(*) INTO param2 FROM users;
END//
```

8. Сохранить созданную в программе-дизайнере схему базы данных на локальный компьютер.

9. Запустить генерацию базы данных на сервере MySQL (Пункт меню: Database->Forward Engineer. В опциях необходимо поставить галки против пунктов DROP Objects Before Each CREATE Object и Generate INSERT Statements for Tables).

10. Подключиться к базе данных MySQL (команда mysql –u root –p).

11. Активизировать базу данных «university» (*use*).

12. Выполнить команды *SELECT * FROM users;* и *SELECT * FROM departments;* Записать результат вывода в черновик.

13. Запустить созданную хранимую процедуру командой *CALL getUserInfo(NULL, @total);*

14. Выполнить команды *SELECT @department;* и *SELECT @total;* Записать результаты вывода в черновик.

15. Запустить созданную хранимую процедуру, изменив первый параметр с *NULL* на имя пользователя из таблицы «users». Записать результаты вывода в черновик.

16. Выполнить команду *SELECT @total;* Записать результаты вывода в черновик.

Контрольные вопросы

1. Где сохраняется хранимая процедура?
2. Может ли хранимая процедура вызывать другую хранимую процедуру?
3. Какая SQL команда вызывает хранимую процедуру?
4. Какие существуют типы параметров хранимой процедуры?
5. Какой синтаксис переменной в MySQL?

ЛАБОРАТОРНАЯ РАБОТА 4
ТРАНЗАКЦИИ В MYSQL

Цель работы – получение практических навыков в работе с транзакциями и их свойствами.

Задание

1. Создать или открыть ранее сохраненную базу данных «university» в программе-дизайнере MySQL Workbench.

2. В базе данных «university» создать таблицу «users» с полями (если она не существует):

- ☞ id тип int – ключ (PK);
- ☞ name тип varchar, ненулевое (NN);
- ☞ d_id тип int.

3. Заполнить таблицу «users» произвольными записями - пять строк.

4. Сохранить созданную в программе-дизайнере схему базы данных на локальный компьютер.

5. Запустить генерацию базы данных на сервере MySQL (Пункт меню: Database->Forward Engineer. В опциях необходимо поставить галки против пунктов DROP Objects Before Each CREATE Object и Generate INSERT Statements for Tables).

6. Подключиться к базе данных MySQL (команда mysql –u root –p).

7. Активизировать базу данных «university» (*use*).

8. В командной строке выполнить следующую последовательность команд (в черновике результаты вывода записывать под заголовком: «Откат транзакции, обрыв связи с базой данных»):

SET AUTOCOMMIT=0;
*SELECT * FROM users;* результаты вывода записать в черновик;
START TRANSACTION;
SAVEPOINT sve_point;
DELETE FROM users;
*SELECT * FROM users;* результаты вывода записать в черновик;
ROLLBACK TO SAVEPOINT sve_point;
*SELECT * FROM users;* результаты вывода записать в черновик;
DELETE FROM users;
*SELECT * FROM users;* результаты вывода записать в черновик;
exit; (окно клиента mysql должно закрыться).

9. Подключиться к базе данных MySQL через терминал (mysql –u root –p).

10. Активизировать базу данных «university» (*use*).

11. В командной строке выполнить следующую последовательность команд:

SET AUTOCOMMIT=0;

*SELECT * FROM users;* результаты вывода записать в черновик.

12. В командной строке выполнить следующую последовательность команд (в черновике результаты вывода записывать под заголовком: «Подтверждение транзакции и обрыв связи с базой данных»):

START TRANSACTION;

INSERT INTO users (id, name, d_id) VALUES (100, 'Antonio', 1);

COMMIT;

*SELECT * FROM users;* результаты вывода записать в черновик;

exit;

13. Подключиться к базе данных MySQL через терминал (mysql –u root –p).

14. Активизировать базу данных «university» (*use*).

15. В командной строке выполнить *SELECT * FROM users;* результаты вывода записать в черновик.

Контрольные вопросы

1. Что такое транзакция?
2. Приведите пример синтаксиса транзакции.
3. Какими SQL командами должна заканчиваться каждая транзакция?
4. Что такое ACID? Расшифруйте и объясните термины ACID.
5. Приведите уровни изоляции в порядке от сильного уровня к слабому.
6. Какие существуют режимы AUTOCOMMIT?

ЛАБОРАТОРНАЯ РАБОТА 5
CASE, ВСТРОЕННЫЕ ФУНКЦИИ MYSQL

Цель работы – получение практических навыков в работе со встроенными функциями на языке SQL.

Задание

1. Создать или открыть ранее сохраненную базу данных «university» в программе-дизайнере MySQL Workbench.
2. В базе данных «university» создать таблицу «tasks» с полями:
   - ☞ id тип int, ключ (PK), счетчик (AI);
   - ☞ taskname тип varchar(45), ненулевое (NN);
   - ☞ taskmonth тип varchar(45);
   - ☞ taskday тип varchar(45);
   - ☞ u_id тип int.
3. Создать хранимую процедуру с именем *createTask* (раздел Routines в MySQL Workbech).
4. Заполнить тело процедуры согласно образцу:

```
DELIMITER //
CREATE PROCEDURE `university`.`createTask` (IN tname VARCHAR(45), IN tdate DATETIME, OUT muchdays VARCHAR(45))
BEGIN
    DECLARE tmonth VARCHAR(45);
    SELECT CONCAT('Task month is: ',
            (CASE MONTH(tdate)
                WHEN 1 THEN 'Jan'
                WHEN 2 THEN 'Feb'
                WHEN 3 THEN 'Mar'
                WHEN 4 THEN 'Apr'
                WHEN 5 THEN 'May'
                WHEN 6 THEN 'Jun'
                WHEN 7 THEN 'Jul'
                WHEN 8 THEN 'Aug'
                WHEN 9 THEN 'Sep'
                WHEN 10 THEN 'Oct'
                WHEN 11 THEN 'Nov'
                WHEN 12 THEN 'Dec'
                ELSE 'None'
                END
            )) INTO tmonth;
    INSERT INTO tasks (taskname, taskday, taskmonth) VALUES (tname, DAY(tdate), tmonth);
    SELECT CONCAT('Remains days: ', DATEDIFF(tdate, CURDATE())) INTO muchdays;
END//
```

5. Сохранить созданную в программе-дизайнере схему базы данных на локальный компьютер.

6. Запустить генерацию базы данных на сервере MySQL (Пункт меню: Database->Forward Engineer. В опциях необходимо поставить галки против пунктов DROP Objects Before Each CREATE Object и Generate INSERT Statements for Tables).

7. Подключиться к базе данных MySQL (команда mysql –u root –p).

8. Активизировать базу данных «university» (*use*).

9. В командной строке выполнить следующую последовательность команд:

CALL createTask('Database optimization', '2009-11-01', @days);

SELECT CONCAT('Optimization', @days); результаты вывода записать в черновик;

CALL createTask('Database replication', '2015-09-14', @days);

SELECT CONCAT('Replication ', @days); результаты вывода записать в черновик;

CALL createTask('<Ввести свою задачу>', '<Ввести свою дату>', @days);

*SELECT * FROM tasks;* результаты вывода записать в черновик.

Контрольные вопросы

1. Как функционирует инструкция CASE языка SQL?
2. Какое значение возвращает встроенная функция MONTH?
3. Какие действия выполняет функция CONCAT? Что она возвращает?

ЛАБОРАТОРНАЯ РАБОТА 6
КУРСОРЫ В MYSQL

Цель работы – получение практических навыков в работе с курсорами и обработчиками событий на языке SQL.

Задание

1. Создать или открыть ранее сохраненную базу данных «university» в программе-дизайнере MySQL Workbench.
2. В базе данных «university» создать таблицу «users» с полями:
   - ☞ id тип int – ключ (PK);
   - ☞ name тип varchar(45), ненулевое (NN);
   - ☞ d_id тип int;
3. Заполнить таблицу «users» произвольными записями - пять строк.
4. В базе данных «university» создать таблицу «hobbies» с полями:
   - ☞ id тип int – ключ (PK), счетчик (AI);
   - ☞ hobby тип varchar(45), ненулевое (NN);
   - ☞ u_id тип int.
5. Заполнить таблицу «hobbies» произвольными записями – семь строк. Поле hobby должно содержать текстовое название хобби студента, а поле u_id число – идентификатор студента из таблицы «users». Каждый студент может иметь несколько различных хобби.
6. Создать хранимую процедуру с именем *showHobbies* (раздел Routines в MySQL Workbech).
7. Заполнить тело процедуры согласно образцу:

```
DELIMITER //
CREATE PROCEDURE `university`.`showHobbies` (OUT printstr VARCHAR(500))
BEGIN
  DECLARE done BOOLEAN DEFAULT FALSE;
  DECLARE cur_id, cur_u_id INT;
  DECLARE cur_name, cur_hobby CHAR(45);
  DECLARE outstr VARCHAR(500) DEFAULT '\n';
  DECLARE workstr VARCHAR(500) DEFAULT '';
  DECLARE curusers CURSOR FOR SELECT id, name FROM users;
  DECLARE curhobbies CURSOR FOR SELECT u_id, hobby FROM hobbies;
  DECLARE CONTINUE HANDLER FOR NOT FOUND SET done := TRUE;
  OPEN curusers;
  USERSLOOP: LOOP
    FETCH curusers INTO cur_id, cur_name;
    IF done THEN
      LEAVE USERSLOOP;
    END IF;
    SELECT CONCAT(CONCAT('The hobby list ', cur_name), ' is ') INTO workstr;
    OPEN curhobbies;
```

```
    FETCH curhobbies INTO cur_u_id, cur_hobby;
    WHILE NOT done DO
        IF cur_id = cur_u_id THEN
          SELECT CONCAT(workstr, cur_hobby) INTO workstr;
          SELECT CONCAT(workstr, ', ') INTO workstr;
       END IF;
      FETCH curhobbies INTO cur_u_id, cur_hobby;
    END WHILE;
    SET done := FALSE;
    CLOSE curhobbies;
    SET outstr := CONCAT(outstr, CONCAT(workstr, '\n'));
  END LOOP USERSLOOP;
  CLOSE curusers;
  SET printstr := outstr;
END//
```

8. Сохранить созданную в программе-дизайнере схему базы данных на локальный компьютер. Запустить генерацию базы данных на сервере MySQL. Подключиться к базе данных MySQL (команда mysql –u root –p).

9. Активизировать базу данных «university» (*use*). В командной строке выполнить следующую последовательность команд:

CALL showHobbies(@list);

SELECT @list; результаты вывода записать в черновик.

Контрольные вопросы

1. Что такое курсор? Приведите пример синтаксиса курсора?
2. Какая команда SQL передвигает указатель курсора вперед?
3. Что такое обработчик/драйвер? Приведите пример синтаксиса.

ЛАБОРАТОРНАЯ РАБОТА 7
ЦЕЛОСТНОСТЬ ДАННЫХ, ТРИГГЕРЫ В MYSQL

Цель работы - получение практических навыков в обеспечении целостности данных базы MySQL с использованием триггеров.

Задание

1. Создать или открыть ранее сохраненную базу данных «university» в программе-дизайнере MySQL Workbench.

2. В базе данных «university» создать таблицу «users» с полями:

- id тип int – первичный ключ (PK);
- name тип varchar(45), ненулевое (NN);
- isupdate тип boolean, ненулевое (NN), по умолчанию False.

3. Для таблицы «users», создать триггер (вкладка Triggers):

```
USE `university`;
DELIMITER //
CREATE TRIGGER insertResult AFTER INSERT ON users
   FOR EACH ROW
  BEGIN
     INSERT INTO results SET laboratory=false, examination = 0, u_id = NEW.id;
    END;//
```

4. В базе данных «university» создать таблицу «results» с полями:

- id тип int – счетчик (AI), первичный ключ (PK), ненулевое (NN);
- laboratory тип boolean, ненулевое (NN);
- examination тип int, ненулевое (NN);
- u_id тип int, ненулевое (NN).

5. Для таблицы «results» создать Foreign Key (вкладка Foreign Keys) с именем u_id на таблицу «university.users», колонку id. В качестве события указать «On Delete » со значением «CASCADE».

6. Для таблицы «results», создать триггер (вкладка Triggers):

```
USE `university`;
DELIMITER //
CREATE TRIGGER updateUser AFTER UPDATE ON results
   FOR EACH ROW
   BEGIN
     UPDATE users Set isupdate=true WHERE id = NEW.u_id;
   END;//
```

7. Сохранить созданную в программе-дизайнере схему базы данных на локальный компьютер. Запустить генерацию базы данных на сервере MySQL. Подключиться к базе данных MySQL (команда mysql –u root –p).

8. Активизировать базу данных «university» (*use*). В командной строке выполнить следующую последовательность команд:

```
INSERT INTO users VALUES (1, 'Ivan', 0);
```

INSERT INTO users VALUES (2, 'Petr', 0);
INSERT INTO users VALUES (3, 'Egor', 0);
INSERT INTO users VALUES (4, 'Vladimir', 0);
*SELECT * FROM users;* результаты вывода записать в черновик;
*SELECT * FROM results;* результаты вывода записать в черновик;
DELETE FROM users WHERE name LIKE 'Ivan';
DELETE FROM users WHERE name LIKE 'Vladimir';
*SELECT * FROM users;* результаты вывода записать в черновик;
*SELECT * FROM results;* результаты вывода записать в черновик;
UPDATE results SET laboratory = true, examination = 5 WHERE u_id = 2;
*SELECT * FROM results WHERE u_id = 2;* результаты вывода записать в черновик;
*SELECT * FROM users;* результаты вывода записать в черновик.
9. Проанализировать полученные результаты.

Контрольные вопросы

1. Что такое внешний ключ, его предназначение?
2. Можно ли обеспечить целостность данных без внешних ключей?
3. Что такое триггер? Принцип его работы.
4. Какие существуют режимы связи, созданные посредством внешнего ключа?
5. В чем разница между порожденной и порождающей таблицами?

ЛАБОРАТОРНАЯ РАБОТА 8
ДИНАМИЧЕСКИЙ ВЫВОД ДАННЫХ ИЗ MYSQL С ПОМОЩЬЮ PHP СКРИПТА

Цель работы – получение практических навыков в создании PHP скриптов, позволяющих осуществлять динамический вывод данных из таблиц MySQL.

Задание

1. Создать базу данных «library» в программе-дизайнере MySQL Workbench.

2. В базе данных «library» создать таблицу «books» (charset – UTF8) с полями:

- ☞ id тип int, счетчик (AI), первичный ключ (PK), ненулевое (NN);
- ☞ name тип varchar(255), ненулевое (NN), Charset – UTF8;
- ☞ author тип varchar(100), ненулевое (NN), Charset – UTF8;
- ☞ desc тип varchar(255), Charset – UTF8;
- ☞ image тип varchar(255), Charset – UTF8;
- ☞ s_id тип int.

3. В базе данных «library» создать таблицу «sections» (charset – UTF8) с полями:

- ☞ s_id тип int, счетчик (AI), первичный ключ (PK), ненулевое (NN);
- ☞ name тип varchar(255), ненулевое (NN), Charset – UTF8.

4. Для таблицы «books» создать внешний ключ (вкладка Foreign Keys) с именем s_id на таблицу «library.sections», столбец s_id. В качестве событий указать:

- ☞ «ON DELETE » со значение «CASCADE»;
- ☞ «ON UPDATE» со значение «CASCADE».

5. В таблицу «books» ввести восемь строк с информацией о книгах (табл. 8.1). В таблице «sections» создать три раздела. Названия разделов определяются студентом самостоятельно, после анализа названия книг в приложении. Каждая из восьми книг должна принадлежать какому-либо одному разделу.

6. Сохранить созданную в программе-дизайнере схему базы данных на локальный компьютер. Запустить генерацию базы данных на сервере MySQL. (Пункт меню: Database->Forward Engineer. В опциях необходимо поставить галки против пунктов DROP Objects Before Each CREATE Object и Generate INSERT Statements for Tables).

7. Подключиться к базе данных MySQL (команда mysql –u root –p).

8. Активизировать базу данных «library» (*use*).

9. С помощью команды *UPDATE* внести следующие изменения в строки таблицы «books»:

☞ для книги «Стив Джобс. Биография» в поле «image» соответствующей строки внести значение: «jobs.jpg»;

☞ для книги «Java 2. Библиотека профессионала. Том 1. Основы» в поле «image» соответствующей строки внести значение: «java.jpg».

Тексты запросов *UPDATE* записать в черновик.

10. С помощью команд *SELECT* создать и выполнить два запроса на выборку всех записей из таблиц: «books» и «sections». Убедиться в наличии данных в таблицах. В черновик записать полученные таблицы (для таблицы «books» в черновик записать все значения, кроме значений столбца desc).

11. В домашней директории http сервера создать PHP скрипт с именем «library.php» (соблюдение регистра обязательно) и содержанием:

```
<?php
$hostname = "localhost";
$username = "root";
$password = "";
$dbName = "library";
?>
<html>
<head>
   <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
   <title>Library</title>
</head>
<body>
<?
mysql_connect($hostname,$username,$password) OR DIE("Can't create a
connection!");
@mysql_select_db("$dbName") or die("Can't select a database!");

$query = "SELECT * FROM sections ORDER BY s_id";
mysql_set_charset("utf8");
$result = mysql_query($query);
$number = mysql_numrows($result);
echo "<form action='library.php' method='post' name='sections_form'>";
echo "Sections: ";
echo "<select name='section'>";
$select_sec = $_POST["section"]==""?1:$_POST["section"];
$i=0;
while ($i < $number){
   echo "<option value='".mysql_result($result, $i,
"s_id")."'".($i+1==$select_sec?" selected>":">");
   echo mysql_result($result, $i, "name")."</option>";
   $i++;
}
```

```
echo "<input type='submit' value='Search...'/>";
echo "</select>";
echo "</form>";

echo "<center>Search result</center>";
echo "<table border='1' cols='4' cellpadding='0' cellspacing='0'>";
echo "<tr>";
echo "<td>Image</td>";
echo "<td>Name</td>";
echo "<td>Author</td>";
echo "<td>Description</td>";
echo "</tr>";

if ($_POST["section"]!=""){ $section_id = $_POST["section"];}
else {$section_id = 1;}

$query = "SELECT * FROM books WHERE s_id = $section_id ORDER BY id";
$result = mysql_query($query);
while ($row = mysql_fetch_assoc($result)) {
   echo "<tr>";
   echo "<td>".($row["image"]==""?"No image":"<img width='80'
src='".$row["image"]."'></img>")."</td>";
   echo "<td>".$row["name"]."</td>";
   echo "<td>".$row["author"]."</td>";
   echo "<td>".$row["desc"]."</td>";
   echo "</tr>";
}
echo "</table>";
mysql_close();
?>
</body>
</html>
```

12. Скопировать в директорию, где расположен файл «library.php», файлы jpg с обложками книг (расположение файлов уточнить у преподавателя).

13. Запустить браузер и открыть в нем созданную страницу. Строка (URL) разработанной PHP страницы должна иметь следующий вид: http://localhost/library.php (после localhost возможно наличие дополнительной директории, в зависимости от настроек HTTP сервера). Вывести на странице для каждого раздела библиотеки свой набор книг, путем выбора раздела из выпадающего меню и нажатия кнопки «Search». В черновик записать названия разделов и соответствующие им наименования книг.

14. Проанализировать полученные результаты.

Таблица 8.1

Список книг библиотечной базы данных MySQL

Наименование/ Name	**Автор/Author**	**Аннтотация/Desc**
Стив Джобс. Биография	Уолтер Айзексон	В основу этой биографии легли беседы с самим Стивом Джобсом, а также с его родственниками, друзьями, врагами, соперниками и коллегами
Сократ	Игорь Суриков	Один из величайших философов в мировой истории, при этом за всю жизнь не написавший ни одного философского трактата
Леонардо да Винчи	В. П. Зубов	Книга В.П. Зубова впервые была издана в 1961 г. большим тиражом и получила международное признание. Со временем она стала библиографической редкостью
Ницше	Жиль Делез	Классическая работа одного из крупнейших французских философов XX века, включающая в себя компактное и емкое изложение биографии и философского творчества Фридриха Ницше
Метро 2033. Британия	Грант Макмастер	Герой романа «Метро 2033. Британия» путешествует между Глазго и Лондоном через изменившиеся до неузнаваемости Шотландию и Англию
Дюна. Мессия Дюны. Дети Дюны	Фрэнк Герберт	Самый авторитетный журнал научной фантастики «Локус» попросту признал «Дюну» первым романом эпопеи о «песчаной планете», лучшим научно-фантастическим романом всех времен и народов
Java 2. Библиотека профессионала. Том 1. Основы	Кей Хорстманн, Гари Корнелл	Книга ведущих специалистов по программированию на языке Java представляет собой обновленное издание фундаментального труда, учитывающее всю специфику новой версии платформы Java SE 6
Linux. Необходимый код и команды. Карманный справочник	Скотт Граннеман	Данная книга представляет собой краткое пособие по основным командам операционной системы Linux

Контрольные вопросы

1. Где выполняется PHP скрипт: на стороне клиента или сервера?
2. Какие основные компоненты необходимы для выполнения PHP скрипта?
3. Приведите синтаксис объявления переменной в PHP?
4. Как работает конкатенация строк в PHP?
5. Необходимо ли задавать тип переменной в PHP скрипте?
6. Синтаксис комментариев в PHP.

ЛАБОРАТОРНАЯ РАБОТА 9
ИЗМЕНЕНИЕ ДАННЫХ В MYSQL С ПОМОЩЬЮ PHP СКРИПТА

Цель работы – получение практических навыков в создании PHP скриптов, позволяющих вносить изменения и осуществлять динамический вывод данных из таблиц MySQL.

Задание

1. Создать базу данных «phonebook» в программе-дизайнере MySQL Workbench.

2. В базе данных «phonebook» создать таблицу «numbers» (charset – UTF8) с полями:

- ☞ id тип int, счетчик (AI), первичный ключ (PK), ненулевое (NN);
- ☞ areacode тип int (3), ненулевое (NN);
- ☞ phonenum тип varchar(7), ненулевое (NN), Charset – UTF8;
- ☞ name тип varchar(255), ненулевое (NN), Charset – UTF8;
- ☞ g_id тип int.

3. В базе данных «phonebook» создать таблицу «groups» (charset – UTF8) с полями:

- ☞ id тип int, счетчик (AI), первичный ключ (PK), ненулевое (NN);
- ☞ group тип varchar(255), ненулевое (NN), Charset – UTF8.

4. В таблицу «groups» ввести пять строк с названием разделов телефонной книги, например: друзья, коллеги, школьные друзья и т. п.

5. Сохранить созданную в программе-дизайнере схему базы данных на локальный компьютер. Запустить генерацию базы данных на сервере MySQL. (Пункт меню: Database->Forward Engineer. В опциях необходимо поставить галки против пунктов DROP Objects Before Each CREATE Object и Generate INSERT Statements for Tables).

6. В домашней директории веб сервера (/var/www) создать PHP скрипт с именем «phonebook.php» (соблюдение регистра обязательно) и содержанием:

```
<?php
$hostname = "localhost";
$username = "root";
$password = "";
$dbName = "phonebook";
?>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Phonebook</title>
</head>
<body>
```

```
<?
  mysql_connect($hostname,$username,$password) OR DIE("Can't create a
connection!");
  @mysql_select_db("$dbName") or die("Can't select a database!");
  mysql_set_charset("utf8");

  $name = $_POST["name"];
  $areacode = $_POST["areacode"];
  $phonenum = $_POST["phonenum"];
  $group = $_POST["group"];


  if ($name!="" && $areacode!="" && $phonenum!="")
  {
 $query = " A. … … … … ";
     if (!mysql_query($query))
     {
   die('Error: ' . mysql_error());
     }
      echo "1 record added";
  }

  $query = " B. … … … … ";
  $result = mysql_query($query);

  if (mysql_numrows($result) == 0)
  {
 echo "Sorry, empty book <br><br>";
  }
  else
  {
     echo "<table border='1' cols='4' cellpadding='0' cellspacing='0'>";
     echo
"<tr><td>Name</td><td>Areacode</td><td>Phonenum</td><td>Group</td></
tr>";
       while ($row = mysql_fetch_assoc($result)) {
    echo "<tr>";
    echo "<td>".$row["name"]."</td>";
         echo "<td>".$row["areacode"]."</td>";
         echo "<td>".$row["phonenum"]."</td>";
         echo "<td>".$row["group"]."</td>";
    echo "</tr>";
     }
 echo "</table><br>";
  }
```

```
        echo "<form action='phonebook.php' method='post'
name='phonebook_form'>";
        echo "<table border='0' cols='4' cellpadding='0' cellspacing='0'
width='400'>";
        echo "<tr>";
        echo "<td>Name: <input type='text' name='name'></input></td>";
        echo "<td>Areacode: <input type='text' name='areacode'></input></td>";
        echo "<td>Phone: <input type='text' name='phonenum'></input></td>";
        echo "<td>Group: <select name='group'>";
        $query = " C. … … … … ";
        $result = mysql_query($query);
        while ($row = mysql_fetch_assoc($result)) {
       echo"<option value='".$row["id"]."'>".$row["group"]."</option>";
        }
        echo "</select></td>";
        echo "</tr>";
        echo "</table>";
        echo "<table border='0' cols='3' cellpadding='0' cellspacing='0'>";
        echo "<tr>";
        echo "<td><input type='submit' name='insert'
value='Insert'></input></td>";
        echo "</tr>";
        echo "</table>";
        echo "</form>";

        mysql_close();

    ?>
    </body>
    </html>
```

7. В PHP скрипте самостоятельно написать запросы для пунктов:

A. запрос на добавление строк в таблицу «numbers». В таблицу необходимо внести значения из переменных *$name*, *$areacode*, *$phonenum* и *$group*. Обратите внимание на тип переменных!

B. запрос на левое объединение таблиц «numbers» и «groups» по полям g_id и id и сортировкой по имени контакта в телефонной книжке.

C. запрос на выборку всех записей из таблицы «groups» и сортировкой по имени группы. Не следует забывать, что слово group в MySQL является командой языка!

8. Полученные три запроса переписать в черновик.

9. Запустить браузер и открыть в нем созданную страницу. Строка (URL) разработанной PHP страницы должна иметь следующий вид: http://localhost/phonebook.php (после localhost возможно наличие дополнительной директории, в зависимости от настроек HTTP сервера).

10. Добавить через созданную страницу семь записей в телефонную книжку.

11. Записать в черновик итоговую таблицу.

12. Проанализировать полученные результаты.

Контрольные вопросы

1. Какие команды языка PHP устанавливают соединение с MySQL и позволяют выбрать базу данных?

2. Какая команда используется для закрытия соединения с MySQL?

3. Какой командой PHP осуществляется выполнения SQL запроса?

4. Каким образом из результирующего набора извлекаются строки, полученные при выполнении SQL запроса? Приведите два варианта.

СПРАВОЧНЫЕ МАТЕРИАЛЫ ПО ЯЗЫКУ PHP (HYPERTEXT PREPROCESSOR)

PHP – это язык программирования, предназначенный для интерактивного создания веб-страниц на компьютере, который называется веб-сервером. В отличие от HTML, когда веб-браузер генерирует страницу на основе тегов и разметки, PHP-код исполняется между запрошенной страницей и веб-сервером, добавляя и изменяя основной код HTML.

Язык PHP упрощает разработку веб-страниц, поскольку платформа PHP содержит весь необходимый программный код. Хотя PHP прекрасно подходит для разработки веб-приложений, хранением информации сам он не занимается. Разработчики сценариев на PHP обычно берут базу данных MySQL, которая и служит делопроизводителем для пользовательской информации, обрабатываемой PHP. СУБД MySQL автоматизирует большую часть задач, связанных с хранением и извлечением пользовательской информации на основе заданных критериев.

Разработка динамических веб-страниц включает три основных компонента: веб-сервер, язык программирования сценариев, исполняемых на стороне сервера, и базу данных.

Чтобы выделить PHP-код и тем самым проинформировать веб-сервер о необходимости его обработки, PHP-код размещают между формальными или неформальными тегами, смешивая с HTML. В примере 1 демонстрируется это с помощью конструкций *echo* и *print*. Конструкции *echo* и *print* почти совпадают, за исключением того, что конструкция *echo* может принимать несколько аргументов и не возвращает никакого значения, тогда как конструкция *print* способна принимать только один аргумент. Файл этого примера назван hello.php; но для имени можно взять любое другое имя, главное чтобы оно имело расширение .php. Это расширение сообщает веб-серверу, что файл нужно обрабатывать как PHP-код.

Пример 1. Вызов echo и print в hello.php:

```
<html>
<head><title>Hello World</title></head>
<body>
<?php
echo("Hello World!<br />");
print("Goodbye.<br />");
print 'Over and out.';
?>
</body>
</html>
```

Когда браузер запросит этот файл, PHP проинтерпретирует его и воспроизведет текст в формате HTML. В примере 2 приводится текст HTML, который будет получен в результате обработки кода из примера 1.

Пример 2. Текст HTML, созданный в результате интерпретации PHP-кода из примера 1:

```
<html>
<head><title>Hello World</title></head>
<body>
Hello World!<br /> Goodbye.<br />Over and out.
</body>
</html>
```

При создании PHP-кода принято добавлять в него комментарии, чтобы упростить чтение и сопровождение. Язык PHP поддерживает два типа комментариев; обе формы записи комментариев приводятся в примере 3. Комментарии сохраняются в PHP-файле, но интерпретатор не выводит их. Он выводит лишь комментарии HTML.

Пример 3. Применение комментариев упрощает читаемость кода:

```
<html>
<head><title>Hello World</title></head>
<body>
<?php
// Однострочным комментарием можно сообщить, что
// сценарий собирается напечатать Hello World!
/* Это многострочный комментарий.
Он больше подходит для комментирования
целых блоков программного кода */
echo ("Hello world!<br />");
print ("Goodbye.<br />");
?>
</body>
</html>
```

В примере 3 использованы два типа оформления комментариев: *//* – для однострочных и */* ... */* – для многострочных. Если требуется вставить комментарий в HTML-разметку, в этом случае следует использовать открывающий *<!--* и закрывающий *-->* теги комментария.

Все инструкции PHP-кода завершаются символом точки с запятой ‘;’. Поэтому символ точки с запятой нельзя использовать в программных именах. Признак хорошего стиля, равно как и полезная привычка, – сразу после точки с запятой начинать новую строку.

В языке PHP переменную определяют так:

```
$variable_name = value;
```

Обратите внимание на некоторые ключевые моменты синтаксиса переменной. Имя переменной всегда должно начинаться с символа доллара ‘$’. Первый символ после знака доллара должен быть алфавитным символом или символом подчеркивания. Это ни в коем случае не может быть цифра; в противном случае код не будет работать.

Все переменные хранят данные определенных типов. PHP автоматически выбирает тип переменной, соответствующий присвоенному значению. К этим типам данных относятся строки, числа и более сложные типы, например массивы.

Конкатенация – это объединение нескольких текстовых строк и переменных в одну строку, как показано в примере 4. Такое объединение позволяет избавиться от лишних инструкций *echo;*.

Пример 4. Конкатенация строк:

```
<?php
$my_string = "Hello Max. My name is: ";
$newline = "<br />";
echo $my_string . "Paula" . $newline;
echo "Hi, I'm Max. Who are you? " . $my_string . $newline;
echo "Hi, I'm Max. Who are you? " . $my_string . "Paula";
//Последняя строка равнозначна строке
// echo "Hi, I'm Max. Who are you? $my_string Paula";
?>
```

Переменные и текстовые строки объединяются с помощью символа точки ‘.’. Можно делать это многократно. Конкатенация строк и переменных экономит время, помогая быстрее создавать динамические веб-сайты.

Пример 5. Вызов функции mysql_connect:

```
// Подключиться к базе данных
$connection = mysql_connect($db_host, $db_username, $db_password);
if (!$connection) {
die("Невозможно подключиться к базе данных: <br />". mysql_error());
}
```

В примере 5 функция *mysql_connect* принимает в качестве аргументов имя хоста, имя пользователя и пароль. Если соединение было благополучно установлено, функция возвращает дескриптор соединения. Если соединение не может быть установлено, возвращается значение *FALSE*. Чтобы убедиться в том, что соединение установлено, нужно проверить возвращаемое значение. Если при подключении была обнаружена какая-либо ошибка, например неверный пароль, следует вывести предупреждение с помощью функции *mysql_error*, указав причину ошибки.

Следующий этап после установления соединения – выбор используемой базы данных с помощью функции *mysql_select_db*. Она принимает два аргумента: имя базы данных и необязательный дескриптор соединения. Если дескриптор не указан, по умолчанию будет использовано соединение, установленное последним вызовом функции *mysql_connect*.

```
// Выбрать базу данных
$db_select = mysql_select_db($db_database);
if (!$db_select) {
die("Невозможно выбрать базу данных: <br />". mysql_error());
```

```
}
```

Исполнение запросов к базе данных производится с помощью функции *mysql_query*. Она принимает два аргумента: запрос и необязательный дескриптор соединения с базой данных и возвращает результат запроса. Ссылка на результат сохранится в переменной именем *$result*. Эту переменную также следует проверять на равенство значению *FALSE*, чтобы убедиться в отсутствии ошибок в строке запроса или в соединении с базой данных.

```
// Исполнить запрос
$result = mysql_query( $query );
if (!$result) {
die("Невозможно исполнить запрос к базе данных: <br />". mysql_error());
}
```

Исполнив запрос, база данных возвращает результирующий набор данных. Это такие же строки, как при исполнении запросов с помощью клиента командной строки MySQL.

Извлечь строки из результирующего набора данных можно с помощью функции *mysql_fetch_row*, ее синтаксис:

```
array mysql_fetch_row ( resource $result);
```

Данная функция принимает в качестве аргумента результат исполнения запроса, который был сохранен в переменной *$result*. При каждом вызове она возвращает одну строку до тех пор, пока не достигнет последней записи в результирующем наборе данных, после чего будет возвращать значение *FALSE*. Таким образом, нужно организовать выборку данных в цикле с помощью функции *mysql_fetch_row* и определить некоторый программный код, выводящий каждую строку:

```
// Получить и отобразить результаты
while ($result_row = mysql_fetch_row(($result))){
echo 'Название: '.$result_row[1] . '<br />';
echo 'Автор: '.$result_row[4] . '<br /> ';
echo 'Страниц: '.$result_row[2] . '<br /><br />';
}
```

Записи в результирующем наборе данных хранятся в виде массивов, и за одну операцию можно извлечь только одну строку. Конструкция *$result_row[2]* означает обращение ко второму полю (в соответствии с порядком следования столбцов в запросе или в определении таблицы при использовании запроса вида *SELECT **) строки из результирующего набора данных. Это не единственный способ извлечения информации из результирующего набора данных. Функция *mysql_fetch_array* позволяет разместить результаты в массиве за один шаг. Она принимает в первом аргументе результирующий набор данных, а во втором (необязательном) аргументе – значение, описывающее способ связывания данных. Если во втором аргументе передать значение *MYSQL_ASSOC*, будет выполнена

индексация результатов в массиве на основе имен столбцов в запросе. Если передать значение *MYSQL_NUM*, доступ к результатам можно будет осуществлять с помощью числовых индексов, отсчет которых начинается с нуля. Значение по умолчанию – *MYSQL_BOTH*. В этом случае обращаться к массиву можно любым из вышеперечисленных способов.

Функция *mysql_fetch_assoc* – это альтернативный вариант использованию функции *mysql_fetch_array* со значением второго аргумента *MYSQL_ASSOC*.

Как правило, по окончании работы с базой данных следует закрыть соединение. Это можно осуществить с помощью функции *mysql_close*, которая сообщит PHP и MySQL, что соединение уже использоваться не будет, и освободит все занимаемые им ресурсы и память. Синтаксис функции: *mysql_close($connection).*

Павел Сергеевич Зернов

БАЗЫ ДАННЫХ

Методические указания к лабораторным работам

Редактор ***Л.А. Медведева***

Верстка ***М.Ю. Кусовой***

План 2012 г., п. 3
Подписано к печати
Объем 2,25 усл. печ. л. Тираж 60 экз. Заказ
Издательство СПбГУТ. 191186 СПб., наб. р. Мойки, 61
Отпечатано в СПбГУТ

П.С. Зернов

БАЗЫ ДАННЫХ

Методические указания к лабораторным работам

САНКТ-ПЕТЕРБУРГ
2012