**Введение**

**СУБД MySQL** является превосходным сервером БД. Основная причина её популярности – это быстрота (производительность) и стабильность. Система MySQL хорошо проверена и надёжна. MySQL является прекрасным инструментом изучения удаленных баз данных, благодаря простоте использования, а также исключительно скромными требованиями к дисковому пространству и памяти. В MySQL используется двойная схема лицензирования: бесплатная и коммерческая.

Пользовательский интерфейс представляет собой «интерфейс командной строки», который называется «монитор MySQL». При установке MySQL по умолчанию устанавливается и Монитор MySQL.

**Операторы SQL**

Язык **SQL** (structured query language – язык структурированных запросов) разработан фирмой IBM в 1974 году и был принят Американским Национальным институтом стандартов (ANSI) в качестве национального стандарта США в 1992 году. Операторы SQL поддерживаются большинством СУБД. Язык SQL ориентирован на текст. Т.к. он был разработан задолго до появления графических интерфейсов пользователя, то для работы с ним требуется лишь текстовый редактор.

Этот язык представляет собой настолько удачную программную разработку для манипулирования данными реляционной модели, что в настоящее время он является наиболее распространенным программным продуктом такого рода.

**Правила**:

1. Ключевые слова SQL не зависят от регистра
2. каждая команда и каждый оператор MySql заканчивается символом «точка с запятой»
3. **[ ]** не обязательный элемент

**Хранимые процедуры и функции**

Начиная с версии MySQL 5.0 появилась возможность использования хранимых процедур и функций. Хранимая процедура - это процедура хранящаяся в базе данных на сервере.

Просмотреть список всех хранимых процедур в БД

**SHOW PROCEDURE STATUS Where DB = ‘имя БД‘;**

Создание простой процедуры

**CREATE PROCEDURE Имя ([параметр тип, параметр тип, ...])оператор\_SQL;**

Пример:

1) CREATE PROCEDURE Primer1 ( N float) SELECT \* FROM SPORT  
WHERE ROST N ;

2) CREATE PROCEDURE Primer2( N float, M float) SELECT \* FROM SPORT  
WHERE ROST N and VES

3) CREATE PROCEDURE Primer3( N int, M float ) UPDATE SPORT  
SET ROST = M WHERE Nom = N ;

Вызов процедуры

**CALL Имя\_процедуры ( значения );**

Пример:

1) CALL PRimer1 ( 180.5 ) ;

2) CALL Primer2( 180 , 80.5 ) ;

3) CALL Primer3( 2, 190 ) ;

Хранимые процедуры, конечно, не очень полезны, если они содержат одну инструкцию. Составная инструкция может содержать много инструкций, взятых в ключевые слова BEGIN и END.

Каждая инструкция внутри BEGIN … END должна быть завершена операторным разделителем точкой с запятой (;).

Значит необходимо создать другой разделитель инструкции для конца описания процедуры, т.е. оператора CREATE PROCEDURE.

Создание нового разделителя инструкции (т.е. нового признака конца  
оператора)

**DELIMITER символ**

Обратите внимание на то, что в конце нет точки с запятой. Разделителем инструкции может быть больше чем один символ

Пример: **DELIMITER #**

В хранимых процедурах можно определить локальные переменные оператором

**DECLARE имя\_переменной тип\_переменной ;**

Общий вид создания процедуры

**CREATE PROCEDURE Имя ([параметр тип, параметр тип, ... ])**

**BEGIN**

**[ DECLARE имя\_переменной тип\_переменной ] ;**

**[ DECLARE имя\_переменной тип\_переменной ] ;**

**. . .**

**оператор SQL ;**

**оператор SQL ;**

. . .

**END символ**

Пример:

Mysql DELIMITER #

Mysql CREATE PROCEDURE primer4(P int)

BEGIN

DECLARE x INT ;

DECLARE y INT ;

SET y = 10 ;

SET x = P + y;

select x;

END #

Mysql DELIMITER ;

Mysql CALL primer4(5);

На экран будет выведен результат 15.

Создание простой функции

**CREATE FUNCTiON Имя([ параметр тип, параметр тип , ... ]) RETURNS тип**

**RETURN значение ;**

Пример:

CREATE FUNCTiON Primer5 ( N Date ) RETURNS int

RETURN EXTRACT(DAY FROM N );

Вызов функции

**SELECT Имя\_функции ( значения );**

Пример: SELECT Primer5 (‘2018-09-21’);

На экран будет выведен результат 21.

Общий вид создания функции

**CREATE FUNCTiON Имя ([параметр тип, параметр тип , ...]) RETURNS тип**

**BEGIN**

**[ DECLARE имя\_переменной тип\_переменной ] ;**

**[ DECLARE имя\_переменной тип\_переменной ] ;**

**. . .**

**Оператор\_SQL ;**

**Оператор\_SQL ;**

**. . .**

**RETURN значение ;**

**END символ**

Пример: изменим формат даты

Mysql DELIMITER #

Mysql CREATE FUNCTiON primer6 (P Date) RETURNS VarChar(10)

BEGIN

DECLARE Y INT ;

DECLARE M INT ;

DECLARE D INT ;

DECLARE S VarChar(10) ;

SET y = EXTRACT( YEAR FROM P );

SET M = EXTRACT( MONTH FROM P );

SET D = EXTRACT( DAY FROM P );

SET S = CONCAT( D, ‘.’, M, ‘.’, Y );

RETURN S ;

END #

Mysql DELIMITER ;

Mysql SELECT primer6(‘2018-11-21’);

На экран будет выведен результат 21.11.2018

*Замечание*: Функции не должны содержать ссылки к таблицам. Они могут включать инструкции SET, которые содержат ссылки на таблицы, например,

SET @Имя\_глобальной\_переменной =

( SELECT группирующая функция ( поле ) FROM Имя\_таблицы )

Пример:

Mysql DELIMITER #

Mysql CREATE FUNCTiON primer7 (P Float, N int) RETURNS FLOAT

BEGIN

SET @A = (SELECT AVG( ROST ) FROM SPORT WHERE VES P and  
EXTRACT(YEAR FROM DMY) = N );

RETURN @A ;

END #

Mysql DELIMITER ;

Mysql SELECT primer7(70 ,1992);

На экран будет выведен средний рост спортсменов 1992 года рождения, вес которых превышает 70 кг

Лабораторная работа

***Тема*: Создание и использование хранимых процедур и функций в MySQL**

***Цель*:**Обрести практические навыки создания и вызова хранимых процедур и функций

**Задания лабораторной работы:**

1. Установите связь с сервером
2. Создайте на сервере свою базу данных с именем, соответствующим Вашей фамилии (например, BDKomarova)
3. Создайте таблицу Student\_ВАША ФАМИЛИЯ (например, Student\_Komarova)содержащую поля: N\_stud (автоинкрементное), FIO(символьное), D\_R (дата), Gruppa(символьное), Stipendiya (вещественное) , Adres(символьное)
4. Просмотрите структуру таблицы
5. Заполните таблицу Student\_ВАША ФАМИЛИЯ следующими записями (10 студентов)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| N\_Stud | FIO | D\_R | Gruppa | Stipendiya | Kod\_spec |
| 1 | Сидоров | 1992-07-23 | 4п1 | 880 | 3322 |
| 2 | Петров | 1993-11-07 | 4п3 | NULL | 4466 |
| 3 | Григорьев | 1992-04-19 | 4п2 | 920 | 8811 |
| 4 | Данилова | 1992-06-11 | 4п2 | 1200 | 8811 |
| 5 | Андреев | 1993-11-06 | 4п1 | NULL | 3322 |
| 6 | Серова | 1991-02-25 | 4п1 | NULL | 3322 |
| 7 | Коробкова | 1992-11-20 | 4п2 | 880 | 8811 |
| 8 | Смирнов | 1991-07-16 | 4п2 | 920 | 8811 |
| 9 | Подгорнова | 1991-09-25 | 4п3 | 1200 | 4466 |
| 10 | Лебедев | 1991-07-26 | 4п3 | 920 | 4466 |

1. Проверьте заполнение таблицы. Для этого составьте запрос для вывода на экран всей информации о студентах
2. Создайте процедуру EX1 для вывода на экран информации о студентах группы N.
3. Вызовите процедуру EX1 на выполнение для группы 4п1
4. Создайте процедуру EX2 для изменения размера стипендии студента с номером N на М рублей
5. Вызовите процедуру EX2 на выполнение для студента ЛЕБЕДЕВ, изменив его размер стипендии на 1500
6. Cоставьте запрос для вывода на экран всей информации о студентах
7. Создайте процедуру EX3 для вывода на экран информации о студентах (фамилия студента, его специальность, размер стипендии), указанного кода специальности
8. Вызовите процедуру EX3 на выполнение несколько раз для разных специальностей
9. Создайте процедуру EX4 для вывода на экран информации об общей сумме стипендии студентов указанной группы
10. Вызовите процедуру EX4 на выполнение несколько раз для разных групп
11. Создайте процедуру EX5 для вывода на экран информации о студентах указанного года рождения
12. Вызовите процедуру EX5 на выполнение несколько раз с разными данными
13. Создайте процедуру EX6 для вывода на экран информации о студентах указанной группы, которые не получают стипендию
14. Вызовите процедуру EX6 на выполнение несколько раз для разных групп
15. Создайте процедуру EX7 для вывода на экран информации о студентах дата рождения которых попадает в указанный интервал дат. Интервал дат должен передаваться в процедуру.
16. Вызовите процедуру EX7 на выполнение
17. Создайте процедуру EX8, которая:  
    - увеличивает размер стипендии в два раза у студентов указанной  
    группы;  
    - выводит информации о студентах этой группы;  
    - выводит информацию о студентах всех остальных групп, кроме указанной в качестве параметра процедуры.
18. Просмотрите все имеющиеся в Вашей базе данных процедуры
19. Создайте функцию EX9, которая возвращает дату в формате день.месяц.год
20. Вызовите функцию EX9 на выполнение
21. Создайте функцию EX10, которая возвращает количество студентов, фамилия которых начинается с буквы «П»
22. Вызовите функцию EX10 на выполнение
23. Создайте функцию EX11, которая возвращает сумму денег, необходимую для выплаты стипендии студентам указанной группы.
24. Вызовите функцию EX11 на выполнение несколько раз для разных групп
25. Создайте функцию EX12, которая возвращает количество студентов в указанной группе.
26. Вызовите функцию EX12 на выполнение несколько раз для разных групп
27. Оформите отчёт по лабораторной работе

Отчёт по лабораторной работе должен представлять собой файл **Отчёт\_Фамилия.doc**, дополненный рисунками экранных форм с результатами выполнения каждого задания.

Например,

