Министерство науки и высшего образования Российской Федерации  
Пензенский государственный университет  
Кафедра вычислительная техника

**ОТЧЕТ**

по лабораторной работе №4

по дисциплине «Программно-аппаратные средства хранения и обработки данных»

на тему «Создание хранимых процедур»

Выполнили:

студенты группы 22ВВП1

Демин М. С.

Беляев Д. И.

Амиров И. Р.

Приняли:

Дубинин В.Н.

Карамышева Н.С.

Пенза 2025

**Название**

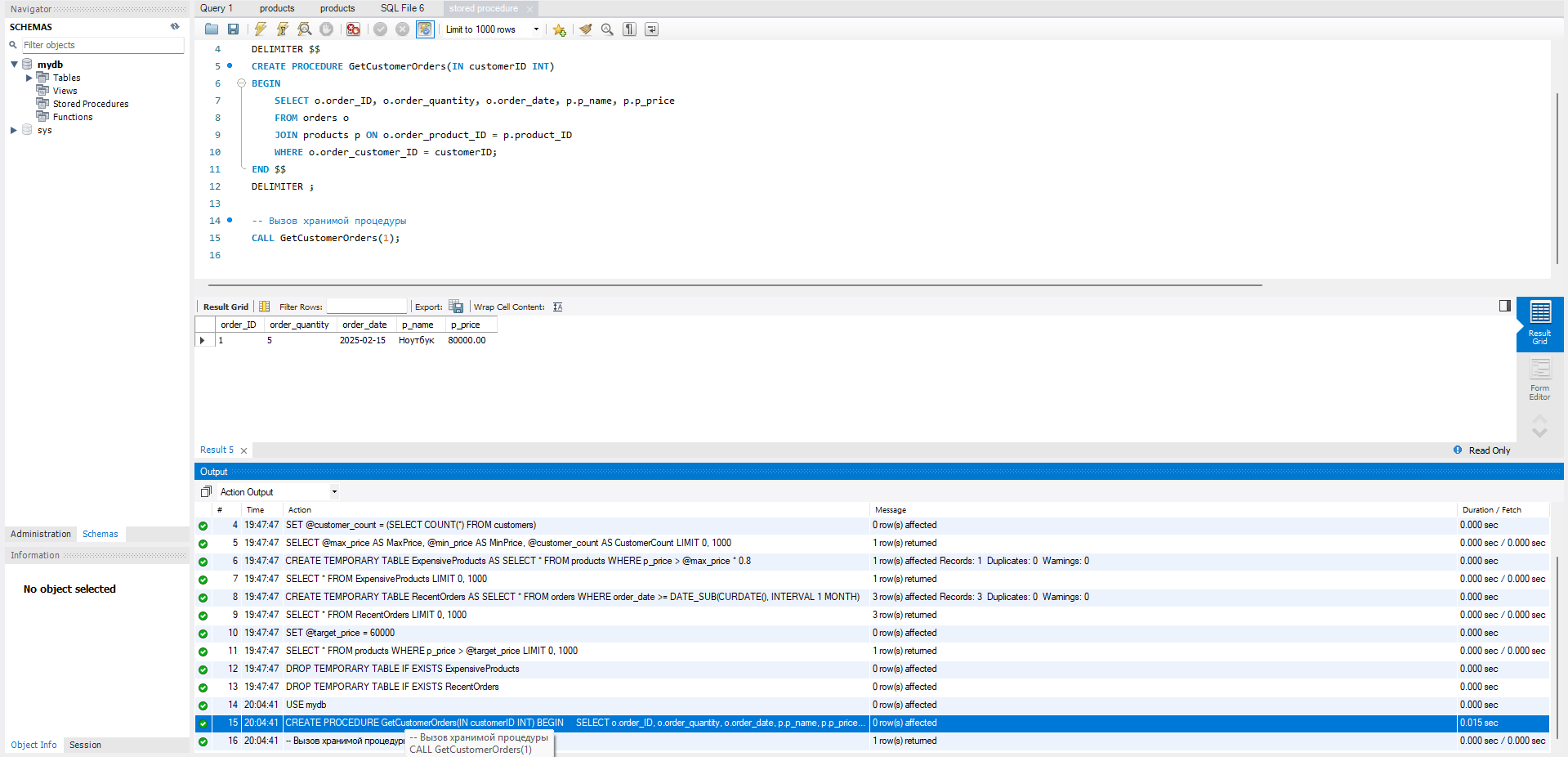
Создание хранимых процедур

**Цель работы**

Научиться создавать хранимые процедуры. Изучить операторы управления потоком данных. Научиться удалять хранимые процедуры. Изучить обработчик ошибок. Изучить курсоры

**Ход работы:**

1. Создание хранимой процедуры



Листинг:

USE mydb;

-- Создание хранимой процедуры для получения заказов клиента

DELIMITER $$

CREATE PROCEDURE GetCustomerOrders(IN customerID INT)

BEGIN

SELECT o.order\_ID, o.order\_quantity, o.order\_date, p.p\_name, p.p\_price

FROM orders o

JOIN products p ON o.order\_product\_ID = p.product\_ID

WHERE o.order\_customer\_ID = customerID;

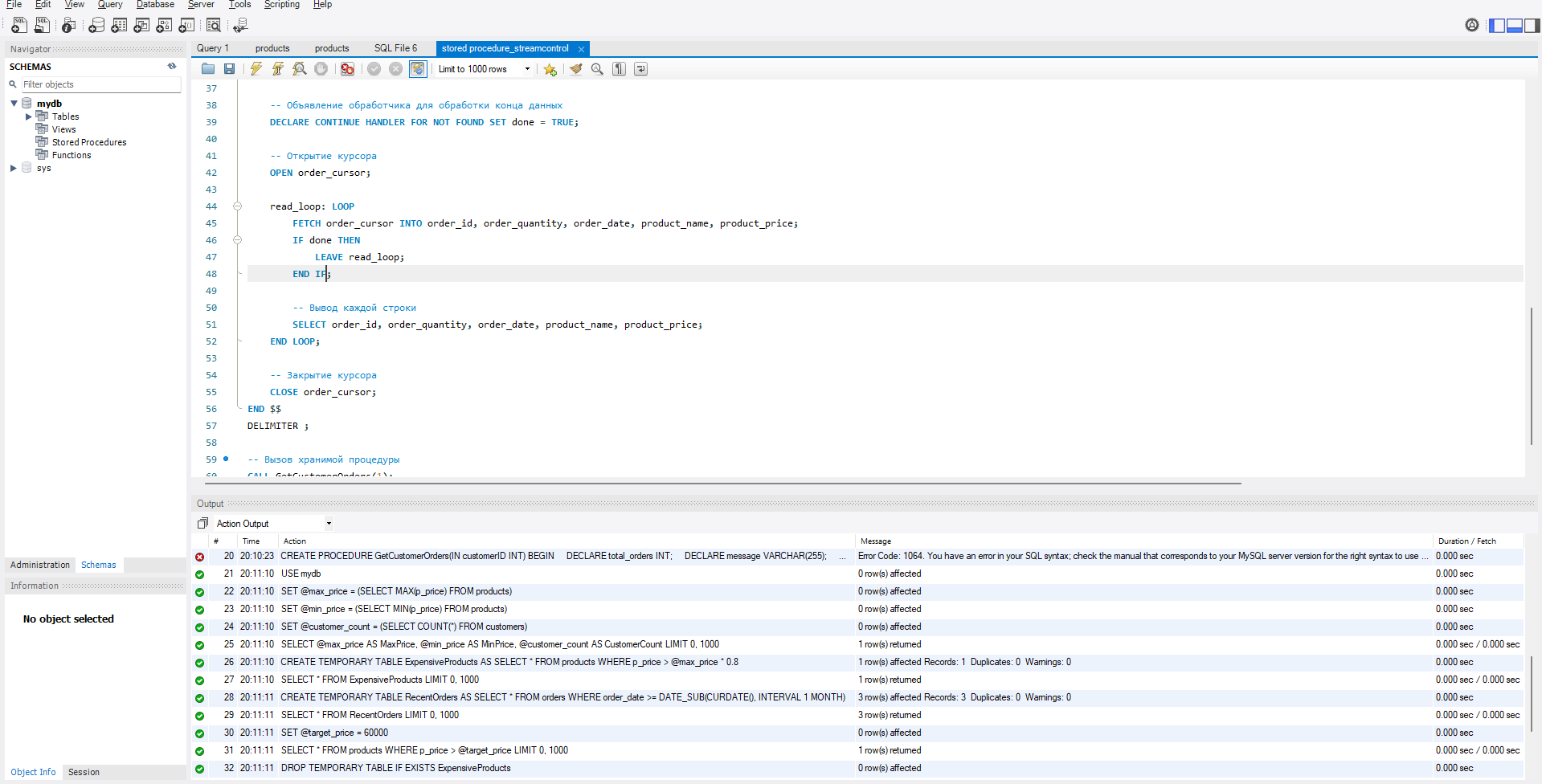
END $$

DELIMITER ;

-- Вызов хранимой процедуры

CALL GetCustomerOrders(1);

1. Добавили операторы управления потоком данных



Листинг:

USE mydb;

-- Создание хранимой процедуры для получения заказов клиента с управлением потоком данных

DELIMITER $$

CREATE PROCEDURE GetCustomerOrders(IN customerID INT)

BEGIN

DECLARE total\_orders INT;

DECLARE message VARCHAR(255);

DECLARE done INT DEFAULT FALSE;

DECLARE order\_id INT;

DECLARE order\_quantity INT;

DECLARE order\_date DATE;

DECLARE product\_name VARCHAR(255);

DECLARE product\_price DECIMAL(10,2);

-- Подсчет количества заказов клиента

SELECT COUNT(\*) INTO total\_orders FROM orders WHERE order\_customer\_ID = customerID;

-- Управление потоком данных с IF

IF total\_orders = 0 THEN

SET message = 'У клиента нет заказов';

ELSEIF total\_orders < 5 THEN

SET message = 'У клиента мало заказов';

ELSE

SET message = 'У клиента много заказов';

END IF;

-- Вывод сообщения

SELECT message AS OrderStatus;

-- Объявление курсора

DECLARE order\_cursor CURSOR FOR

SELECT o.order\_ID, o.order\_quantity, o.order\_date, p.p\_name, p.p\_price

FROM orders o

JOIN products p ON o.order\_product\_ID = p.product\_ID

WHERE o.order\_customer\_ID = customerID;

-- Объявление обработчика для обработки конца данных

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

-- Открытие курсора

OPEN order\_cursor;

read\_loop: LOOP

FETCH order\_cursor INTO order\_id, order\_quantity, order\_date, product\_name, product\_price;

IF done THEN

LEAVE read\_loop;

END IF;

-- Вывод каждой строки

SELECT order\_id, order\_quantity, order\_date, product\_name, product\_price;

END LOOP;

-- Закрытие курсора

CLOSE order\_cursor;

END $$

DELIMITER ;

-- Вызов хранимой процедуры

CALL GetCustomerOrders(1);

1. Добавили в этот скрипт удаление хранимых процедур, обработчики ошибок, курсоры

КАРТИНКА

Листинг:

USE mydb;

-- Создание хранимой процедуры для получения заказов клиента с управлением потоком данных

DELIMITER $$

CREATE PROCEDURE GetCustomerOrders(IN customerID INT)

BEGIN

DECLARE total\_orders INT;

DECLARE message VARCHAR(255);

DECLARE done INT DEFAULT FALSE;

DECLARE order\_id INT;

DECLARE order\_quantity INT;

DECLARE order\_date DATE;

DECLARE product\_name VARCHAR(255);

DECLARE product\_price DECIMAL(10,2);

-- Объявление курсора (ДОЛЖНО быть перед HANDLER)

DECLARE order\_cursor CURSOR FOR

SELECT o.order\_ID, o.order\_quantity, o.order\_date, p.p\_name, p.p\_price

FROM orders o

JOIN products p ON o.order\_product\_ID = p.product\_ID

WHERE o.order\_customer\_ID = customerID;

-- Обработчик ошибок

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

SELECT 'Ошибка при выполнении процедуры' AS ErrorMessage;

END;

-- Обработчик конца данных курсора

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

-- Подсчет количества заказов клиента

SELECT COUNT(\*) INTO total\_orders FROM orders WHERE order\_customer\_ID = customerID;

-- Управление потоком данных с IF

IF total\_orders = 0 THEN

SET message = 'У клиента нет заказов';

ELSEIF total\_orders < 5 THEN

SET message = 'У клиента мало заказов';

ELSE

SET message = 'У клиента много заказов';

END IF;

-- Вывод сообщения

SELECT message AS OrderStatus;

-- Открытие курсора

OPEN order\_cursor;

read\_loop: LOOP

FETCH order\_cursor INTO order\_id, order\_quantity, order\_date, product\_name, product\_price;

IF done THEN

LEAVE read\_loop;

END IF;

-- Вывод каждой строки

SELECT order\_id, order\_quantity, order\_date, product\_name, product\_price;

END LOOP;

-- Закрытие курсора

CLOSE order\_cursor;

END $$

DELIMITER ;

1. Для заданной предметной области написали две хранимые процедуры и включить их в БД

**КАРТИНКА**

Листинг:

USE mydb;

-- Использование переменных и временных таблиц

-- Объявление переменных

SET @max\_price = (SELECT MAX(p\_price) FROM products);

SET @min\_price = (SELECT MIN(p\_price) FROM products);

SET @customer\_count = (SELECT COUNT(\*) FROM customers);

-- Вывод значений переменных

SELECT @max\_price AS MaxPrice, @min\_price AS MinPrice, @customer\_count AS CustomerCount;

-- Создание временной таблицы с дорогими продуктами

CREATE TEMPORARY TABLE ExpensiveProducts AS

SELECT \* FROM products WHERE p\_price > @max\_price \* 0.8;

-- Выборка из временной таблицы

SELECT \* FROM ExpensiveProducts;

-- Создание временной таблицы с заказами за последний месяц

CREATE TEMPORARY TABLE RecentOrders AS

SELECT \* FROM orders WHERE order\_date >= DATE\_SUB(CURDATE(), INTERVAL 1 MONTH);

-- Выборка из временной таблицы

SELECT \* FROM RecentOrders;

-- Использование переменной в условии

SET @target\_price = 60000;

SELECT \* FROM products WHERE p\_price > @target\_price;

-- Удаление временных таблиц

DROP TEMPORARY TABLE IF EXISTS ExpensiveProducts;

DROP TEMPORARY TABLE IF EXISTS RecentOrders;

-- Удаление хранимых процедур, если они уже существуют

DROP PROCEDURE IF EXISTS GetCustomerOrders;

DROP PROCEDURE IF EXISTS CalculateTotalSpent;

-- Создание хранимой процедуры для получения заказов клиента с управлением потоком данных

DELIMITER $$

CREATE PROCEDURE GetCustomerOrders(IN customerID INT)

BEGIN

DECLARE total\_orders INT;

DECLARE message VARCHAR(255);

DECLARE done INT DEFAULT FALSE;

DECLARE order\_id INT;

DECLARE order\_quantity INT;

DECLARE order\_date DATE;

DECLARE product\_name VARCHAR(255);

DECLARE product\_price DECIMAL(10,2);

-- Обработчик ошибок

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

SELECT 'Ошибка при выполнении процедуры' AS ErrorMessage;

END;

-- Подсчет количества заказов клиента

SELECT COUNT(\*) INTO total\_orders FROM orders WHERE order\_customer\_ID = customerID;

-- Управление потоком данных с IF

IF total\_orders = 0 THEN

SET message = 'У клиента нет заказов';

ELSEIF total\_orders < 5 THEN

SET message = 'У клиента мало заказов';

ELSE

SET message = 'У клиента много заказов';

END IF;

-- Вывод сообщения

SELECT message AS OrderStatus;

-- Объявление курсора

DECLARE order\_cursor CURSOR FOR

SELECT o.order\_ID, o.order\_quantity, o.order\_date, p.p\_name, p.p\_price

FROM orders o

JOIN products p ON o.order\_product\_ID = p.product\_ID

WHERE o.order\_customer\_ID = customerID;

-- Объявление обработчика для обработки конца данных

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

-- Открытие курсора

OPEN order\_cursor;

read\_loop: LOOP

FETCH order\_cursor INTO order\_id, order\_quantity, order\_date, product\_name, product\_price;

IF done THEN

LEAVE read\_loop;

END IF;

-- Вывод каждой строки

SELECT order\_id, order\_quantity, order\_date, product\_name, product\_price;

END LOOP;

-- Закрытие курсора

CLOSE order\_cursor;

END $$

-- Создание второй хранимой процедуры для расчета общей суммы заказов клиента

CREATE PROCEDURE CalculateTotalSpent(IN customerID INT, OUT totalSpent DECIMAL(10,2))

BEGIN

DECLARE total DECIMAL(10,2);

-- Подсчет общей суммы заказов клиента

SELECT SUM(o.order\_quantity \* p.p\_price) INTO total

FROM orders o

JOIN products p ON o.order\_product\_ID = p.product\_ID

WHERE o.order\_customer\_ID = customerID;

-- Если заказов нет, установить 0

IF total IS NULL THEN

SET total = 0;

END IF;

-- Присвоение значения выходной переменной

SET totalSpent = total;

END $$

DELIMITER ;

-- Вызов хранимых процедур

CALL GetCustomerOrders(1);

SET @totalSpent = 0;

CALL CalculateTotalSpent(1, @totalSpent);

SELECT @totalSpent AS TotalSpentByCustomer;

**Вывод:**

Научились создавать хранимые процедуры, удалять хранимые процедуры. Изучили операторы управления потоком данных, обработчик ошибок, курсоры.