

**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ СВЯЗИ И МАССОВЫХ
КОММУНИКАЦИЙ**
Ордена Трудового Красного Знамени
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский технический университет связи и информатики»

Кафедра «Математическая кибернетика и информационные технологии»

Лабораторная работа №6

по дисциплине «Математические основы баз данных»

по теме:

«Создание хранимых процедур»

Выполнил: Студент группы
БПИ2403
Сон Владимир Сергеевич
Проверил:
Старший преподаватель
Фатхулин Тимур Джалилевич

Москва
2025

Цель работы

Освоить построение сложных SQL-запросов с объединением таблиц и вложенными подзапросами.

Основные теоретические сведения

Хранимые процедуры - именованный набор SQL-инструкций, сохраняемый на сервере БД.

Создание процедуры

CREATE PROCEDURE имя_процедуры

Типы параметров:

- ***IN*** - входной параметр (значение передается в процедуру)
- ***OUT*** - выходной параметр (значение возвращается из процедуры)
- ***INOUT*** - входно-выходной параметр

Вызов процедуры:

CALL имя_процедуры(параметры)

Основные конструкции:

Условные операторы:

IF условие THEN;

ELSEIF условие THEN

Циклы:

WHILE условие DO

Курсыры - для обработки наборов данных построчно:

DECLARE курсор CURSOR FOR запрос;

FETCH курсор INTO переменные

Обработчики ошибок:

DECLARE mun HANDLER FOR условие действие;

Задачи

Для базы данных из прошлых работ (Вариант 30):

- для заданной предметной области написать две хранимые процедуры и включить их в БД;
- составить отчет по лабораторной работе.

Ход работы

Создаём процедуру и объединяем 3 таблицы через JOIN для связи платежей с клиентами. Рассчитывает общую сумму всех платежей указанного клиента:

```
MariaDB [TradingArea]> DELIMITER //
MariaDB [TradingArea]> CREATE PROCEDURE GetClientTotalPayments(
    → IN client_id INT,
    → OUT total_payments DECIMAL(10,2))
    → BEGIN
    →     SELECT SUM(p.Amount) INTO total_payments
    →     FROM Payment p
    →     JOIN Lease l ON p.LeaseID = l.LeaseID
    →     JOIN Contract ct ON l.ContractID = ct.ContractID
    →     WHERE ct.ClientID = client_id;
    → END //
Query OK, 0 rows affected (0,007 sec)

MariaDB [TradingArea]>
MariaDB [TradingArea]> DELIMITER ;
MariaDB [TradingArea]> CALL GetClientTotalPayments(1, @total);
Query OK, 1 row affected (0,005 sec)

MariaDB [TradingArea]> SELECT @total AS 'Общая сумма платежей';
+-----+
| Общая сумма платежей |
+-----+
| 75000.00 |
+-----+
1 row in set (0,000 sec)
```

Создаём процедуру с курсором для обработки данных построчно и объявляем обработчик ошибок для конца данных *NOT FOUND*. Создаём временную таблицу для хранения результатов и в цикле обрабатываем все платежи за указанный месяц для вывода отсортированного результата. Получаем отчет по арендным платежам за месяц:

```

MariaDB [TradingArea]> DELIMITER //
MariaDB [TradingArea]>
MariaDB [TradingArea]> CREATE PROCEDURE GenerateRentReport(IN report_month DATE)
→ BEGIN
→
→     DECLARE done INT DEFAULT FALSE;
→     DECLARE v_client_name VARCHAR(255);
→     DECLARE v_shopping_center VARCHAR(255);
→     DECLARE v_payment_amount DECIMAL(10,2);
→     DECLARE v_payment_date DATE;
→     DECLARE payment_cursor CURSOR FOR
→         SELECT c.CompanyName, sc.Name, p.Amount, p.PaymentDate
→             FROM Payment p
→             JOIN Lease l ON p.LeaseID = l.LeaseID
→             JOIN Contract ct ON l.ContractID = ct.ContractID
→
→             JOIN Client c ON ct.ClientID = c.ClientID
→             JOIN Store s ON l.StoreID = s.StoreID
→             JOIN ShoppingCenters sc ON s.ShoppingCenterID = sc.ShoppingCenterID
→             WHERE YEAR(p.PaymentDate) = YEAR(report_month)
→                 AND MONTH(p.PaymentDate) = MONTH(report_month);
→     DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
→     CREATE TEMPORARY TABLE IF NOT EXISTS RentReport (
→         ClientName VARCHAR(255), ShoppingCenter VARCHAR(255), PaymentAmount DECIMAL(10,2), PaymentDate DATE);
→     DELETE FROM RentReport;
→     OPEN payment_cursor;
→     payment_loop: LOOP
→         FETCH payment_cursor INTO v_client_name, v_shopping_center, v_payment_amount, v_payment_date;
→         IF done THEN
→             LEAVE payment_loop;
→         END IF;
→         INSERT INTO RentReport VALUES (v_client_name, v_shopping_center, v_payment_amount, v_payment_date);
→     END LOOP;
→     CLOSE payment_cursor;
→
→     SELECT * FROM RentReport ORDER BY PaymentAmount DESC;
→ END //
Query OK, 0 rows affected (0,007 sec)

MariaDB [TradingArea]> DELIMITER :
MariaDB [TradingArea]> CALL GenerateRentReport('2024-03-01');
+-----+-----+-----+-----+
| ClientName          | ShoppingCenter | PaymentAmount | PaymentDate |
+-----+-----+-----+-----+
| ЗАО "KDECALL"       | Южный        | 120000.00    | 2024-03-10  |
| ООО "Силсонг"       | Северный      | 75000.00     | 2024-03-05  |
+-----+-----+-----+-----+
2 rows in set (0,005 sec)

Query OK, 2 rows affected (0,006 sec)

```

Вывод

В ходе работы успешно освоены методы создания и использования хранимых процедур в MySQL. Изучены ключевые аспекты работы с хранимыми процедурами: использование входных и выходных параметров, организация условной логики, обработка наборов данных через курсоры.