/\*1. Добавление нового курса\*/

DELIMITER //

CREATE PROCEDURE add\_course(

IN course\_name VARCHAR(255),

IN course\_description TEXT,

IN course\_price DECIMAL(10, 2)

)

BEGIN

INSERT INTO Course (Name, Description, Price)

VALUES (course\_name, course\_description, course\_price);

END //

DELIMITER ;

CALL add\_course('JavaScript для начинающих', 'Основы программирования на JavaScript', 6000);

/\*2. Добавление нового урока\*/

DELIMITER //

CREATE PROCEDURE add\_lesson(

IN lesson\_name VARCHAR(255),

IN lesson\_description TEXT,

IN lesson\_materials TEXT,

IN course\_id INT

)

BEGIN

INSERT INTO Lesson (Name, Description, Materials, Course\_ID)

VALUES (lesson\_name, lesson\_description, lesson\_materials, course\_id);

END //

DELIMITER ;

CALL add\_lesson('Введение в JavaScript', 'Основные понятия и синтаксис', 'PDF, видео', 11);

/\*3. Добавление нового учащегося\*/

DELIMITER //

CREATE PROCEDURE add\_student(

IN full\_name VARCHAR(255),

IN email VARCHAR(255),

IN phone VARCHAR(15)

)

BEGIN

INSERT INTO Student (Full\_Name, Email, Phone)

VALUES (full\_name, email, phone);

END //

DELIMITER ;

CALL add\_student('Анна Смирнова', 'anna@example.com', '89123456789');

/\*4. Регистрация оплаты за курс\*/

DELIMITER //

CREATE PROCEDURE register\_payment(

IN student\_id INT,

IN course\_id INT,

IN amount DECIMAL(10, 2),

IN payment\_date DATE

)

BEGIN

-- Вставка данных об оплате в таблицу Payment

INSERT INTO Payment (Student\_ID, Course\_ID, Amount, Payment\_Date)

VALUES (student\_id, course\_id, amount, payment\_date);

-- Получение названия курса для уведомления

DECLARE course\_name VARCHAR(255);

SELECT Name INTO course\_name FROM Course WHERE ID = course\_id;

-- Формирование текста уведомления

DECLARE notification\_text TEXT;

SET notification\_text = CONCAT('Ваша оплата за курс "', course\_name, '" на сумму ', amount, ' подтверждена.');

-- Вставка уведомления в таблицу Notification

INSERT INTO Notification (Student\_ID, Text, Notification\_Date)

VALUES (student\_id, notification\_text, payment\_date);

END //

DELIMITER ;

CALL register\_payment(1, 1, 5000, '2023-10-01');

/\*5. Добавление оценки за урок\*/

DELIMITER //

CREATE PROCEDURE add\_grade(

IN lesson\_id INT,

IN student\_id INT,

IN grade INT

)

BEGIN

INSERT INTO Grade (Lesson\_ID, Student\_ID, Grade)

VALUES (lesson\_id, student\_id, grade);

END //

DELIMITER ;

CALL add\_grade(1, 1, 5);

/\*6. Получение списка уроков курса\*/

DELIMITER //

CREATE PROCEDURE get\_course\_lessons(

IN course\_id INT

)

BEGIN

SELECT Name FROM Lesson WHERE Course\_ID = course\_id;

END //

DELIMITER ;

CALL get\_course\_lessons(1);

/\*7. Получение списка курсов учащегося\*/

DELIMITER //

CREATE PROCEDURE get\_student\_courses(

IN student\_id INT

)

BEGIN

SELECT Course.Name

FROM Course

JOIN Payment ON Course.ID = Payment.Course\_ID

WHERE Payment.Student\_ID = student\_id;

END //

DELIMITER ;

CALL get\_student\_courses(1);

/\*8. Получение статистики по оплатам\*/

DELIMITER //

CREATE PROCEDURE get\_payment\_statistics(

IN start\_date DATE,

IN end\_date DATE,

OUT total\_payments INT,

OUT total\_amount DECIMAL(15, 2)

)

BEGIN

-- Общее количество оплат

SELECT COUNT(\*) INTO total\_payments

FROM payments

WHERE payment\_date BETWEEN start\_date AND end\_date;

-- Общая сумма оплат

SELECT SUM(amount) INTO total\_amount

FROM payments

WHERE payment\_date BETWEEN start\_date AND end\_date;

END //

DELIMITER ;

CALL get\_payment\_statistics('2023-09-01', '2023-09-30', @total\_payments, @total\_amount);

-- Вывод результатов

SELECT @total\_payments AS total\_payments, @total\_amount AS total\_amount;

/\*9. Получение списка уведомлений учащегося\*/

DELIMITER //

CREATE PROCEDURE get\_student\_notifications(

IN student\_id INT

)

BEGIN

-- Выборка уведомлений для указанного учащегося

SELECT message

FROM notifications

WHERE student\_id = student\_id;

END //

DELIMITER ;

CALL get\_student\_notifications(1);

/\*10. Получение средней оценки учащегося\*/

DELIMITER //

CREATE PROCEDURE get\_student\_average\_grade(

IN student\_id INT,

OUT average\_grade DECIMAL(5, 2)

)

BEGIN

-- Вычисление средней оценки учащегося

SELECT AVG(grade) INTO average\_grade

FROM grades

WHERE student\_id = student\_id;

END //

DELIMITER ;

CALL get\_student\_average\_grade(1, @average\_grade);

-- Вывод результатов

SELECT @average\_grade AS average\_grade;