SQL-скрипты

Код создания бд

```
SET @OLD_UNIQUE_CHECKS = @@ UNIQUE_CHECKS, UNIQUE_CHECKS = 0;

SET @OLD_FOREIGN_KEY_CHECKS = @@ FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS = 0;

SET @OLD_SQL_MODE = @@ SQL_MODE, SQL_MODE =
'ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBS
TITUTION';

CREATE SCHEMA IF NOT EXISTS `fitness_center` DEFAULT CHARACTER
SET utf8;

USE `fitness_center`;

CREATE TABLE IF NOT EXISTS `client`(
    `id` int NOT NULL AUTO_INCREMENT,
    `firstName` varchar(15) NOT NULL,
    `lastName` varchar(15) NOT NULL,
    `phone` varchar(15) NOT NULL,
    `email` varchar(20) NOT NULL,
    `password` varchar(255) NOT NULL,
    PRIMARY KEY (`id`)) ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `trainer`(
    `id` int NOT NULL AUTO_INCREMENT,
    `firstName` varchar(15) NOT NULL,
    `lastName` varchar(15) NOT NULL,
    `phone` varchar(15) NOT NULL,
    `speciality` varchar(25) NOT NULL,
    PRIMARY KEY (`id`)) ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `hall`(
    `id` int NOT NULL AUTO_INCREMENT,
    `name` varchar(25) NOT NULL,
    `capacity` int NOT NULL,
    PRIMARY KEY (`id`)) ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `class`(
    `id` int NOT NULL AUTO_INCREMENT,
    `name` varchar(45) NOT NULL,
    `duration` DECIMAL(1, 2) NOT NULL,
    `beginAt` DATETIME NOT NULL,
    `price` int NOT NULL,
    `hall_id` int NOT NULL,
    `trainer_id` int NOT NULL,
    PRIMARY KEY (`id`),
    FOREIGN KEY (`hall_id`) REFERENCES `hall`(`id`) ON DELETE NO ACTION ON UPDATE NO ACTION,
    FOREIGN KEY (`trainer_id`) REFERENCES `trainer`(`id`) ON DELETE NO ACTION ON UPDATE NO ACTION) ENGINE =
InnoDB;

CREATE TABLE IF NOT EXISTS `sign_for_class`(
    `id` int NOT NULL AUTO_INCREMENT,
    `client_id` int NOT NULL,
    `class_id` int NOT NULL,
    `date` DATETIME NOT NULL,
    PRIMARY KEY (`id`),
    FOREIGN KEY (`client_id`) REFERENCES `client`(`id`) ON DELETE NO ACTION ON UPDATE NO ACTION,
    FOREIGN KEY (`class_id`) REFERENCES `class`(`id`) ON DELETE NO ACTION ON UPDATE NO ACTION) ENGINE =
InnoDB;

SET SQL_MODE = @OLD_SQL_MODE;

SET FOREIGN_KEY_CHECKS = @OLD_FOREIGN_KEY_CHECKS;

SET UNIQUE_CHECKS = @OLD_UNIQUE_CHECKS;
```

Код создания объектов

```
-- Представление для удобного просмотра расписания и тренеров
CREATE VIEW schedule_view AS
SELECT
    c.id AS class_id,
    c.name AS class_name,
    c.duration,
    c.beginAt,
    c.price,
    t.firstName AS trainer_firstName,
    t.lastName AS trainer_lastName,
    h.name AS hall_name,
    h.capacity
FROM
    fitness_center.class c
    JOIN fitness_center.trainer t ON c.trainer_id = t.id
    JOIN fitness_center.hall h ON c.hall_id = h.id
ORDER BY
    c.beginAt;

-- Процедура для добавления нового тренера
CREATE PROCEDURE add_trainer(IN firstName varchar(15), IN lastName varchar(15), IN phone varchar(15), IN
speciality varchar(25))
BEGIN
    INSERT INTO fitness_center.trainer(firstName, lastName, phone, speciality)
        VALUES(firstName, lastName, phone, speciality);

END;

-- Процедура для удаления тренера
CREATE PROCEDURE delete_trainer(IN trainer_id int)
BEGIN
    DELETE FROM fitness_center.trainer
    WHERE id = trainer_id;

END;

-- Процедура для регистрации клиента на занятие
CREATE PROCEDURE register_for_class(IN client_id int, IN class_id int, IN registration_date DATETIME)
BEGIN
    INSERT INTO fitness_center.sign_for_class(client_id, class_id, date)
        VALUES(client_id, class_id, registration_date);

END;
```

```
-- Процедура для добавления нового клиента
CREATE PROCEDURE add_client(IN firstName varchar(15), IN lastName varchar(15), IN phone varchar(15), IN email
varchar(20), IN PASSWORD VARCHAR(255))
BEGIN
    INSERT INTO fitness_center.client(firstName, lastName, phone, email, PASSWORD)
        VALUES(firstName, lastName, phone, email, PASSWORD);

END;

CREATE PROCEDURE add_class_with_checks(IN p_name varchar(45), IN p_duration DECIMAL(1, 2), IN p_beginAt
DATETIME, IN p_price int, IN p_hall_id int, IN p_trainer_id int)
BEGIN
DECLARE
    trainer_exists int;
    DECLARE hall_exists int;
    DECLARE time_conflict int;
    -- Проверка существования тренера
    SELECT
        COUNT(*) INTO trainer_exists
    FROM
        fitness_center.trainer
    WHERE
        id = p_trainer_id;
    IF trainer_exists = 0 THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Тренер не существует.';
    END IF;
    -- Проверка существования зала
    SELECT
        COUNT(*) INTO hall_exists
    FROM
        fitness_center.hall
    WHERE
        id = p_hall_id;
    IF hall_exists = 0 THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Зал не существует.';
    END IF;
    -- Проверка на пересечение времени
    SELECT
        COUNT(*) INTO time_conflict
    FROM
        fitness_center.class
    WHERE
        hall_id = p_hall_id
        AND ((beginAt < DATE_ADD(p_beginAt, interval p_duration HOUR)
                AND DATE_ADD(beginAt, interval duration HOUR) > p_beginAt));
    IF time_conflict > 0 THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Время занятия пересекается с другим занятием в этом
зале.';
    END IF;
    -- Если все проверки пройдены, добавляем занятие
    INSERT INTO fitness_center.class(name, duration, beginAt, price, hall_id, trainer_id)
        VALUES (p_name, p_duration, p_beginAt, p_price, p_hall_id, p_trainer_id);
```

```
END;
```