<!--Для выполнения заданий по созданию и заполнению базы данных деканата для студентов, а также по выборке данных, выполните следующие шаги.

Л.р. №1. Создание и заполнение отношений

1. Таблица "Дисциплины" (Disciplines)

Создадим таблицу для хранения дисциплин с шифром и названием дисциплины.

sql

CREATE TABLE Disciplines (

discipline\_code CHAR(8) PRIMARY KEY, -- Шифр дисциплины

name VARCHAR(100) -- Название дисциплины

);

2. Таблица "Студенты" (Students)

Таблица для хранения информации о студентах. Включает номер зачётной книжки, ФИО, год поступления, группу, курс, средний балл, форму обучения и академический отпуск.

sql

CREATE TABLE Students (

record\_book\_number CHAR(10) PRIMARY KEY, -- Номер зачётной книжки

full\_name VARCHAR(50) NOT NULL, -- ФИО студента

admission\_year INT NOT NULL, -- Год поступления

group\_code CHAR(7) NOT NULL, -- Группа

course INT(1) NOT NULL, -- Курс

average\_score DECIMAL(3,1), -- Средний балл

study\_form CHAR(8) DEFAULT 'дневная', -- Форма обучения (по умолчанию дневная)

academic\_leave INT(1) DEFAULT 0 -- Академический отпуск (по умолчанию 0)

);

3. Таблица "Сессия" (Exams)

Таблица для хранения информации о сессии, которая включает группу, шифр дисциплины (внешний ключ), дату экзамена, аудиторию и преподавателя.

sql

CREATE TABLE Exams (

group\_code CHAR(7), -- Группа

discipline\_code CHAR(8), -- Шифр дисциплины (внешний ключ)

exam\_date DATE, -- Дата экзамена

classroom INT(3), -- Аудитория

examiner VARCHAR(100), -- Преподаватель

PRIMARY KEY (group\_code, discipline\_code, exam\_date), -- Композитный первичный ключ

FOREIGN KEY (discipline\_code) REFERENCES Disciplines(discipline\_code) -- Внешний ключ

);

Л.р. №2. Выборка данных

1. Проверка уникальности комбинации "Дата" и "Аудитория"

Для проверки уникальности комбинации "Дата" и "Аудитория" можно использовать запрос с группировкой и подсчетом количества записей. Если количество записей больше 1, это значит, что для данной даты и аудитории назначено несколько экзаменов.

sql

SELECT exam\_date, classroom, COUNT(\*)

FROM Exams

GROUP BY exam\_date, classroom

HAVING COUNT(\*) > 1;

Этот запрос возвращает все комбинации "Дата" и "Аудитория", которые встречаются несколько раз.

2. Создание расписания экзаменов на сессию для произвольной группы

Для создания расписания экзаменов для произвольной группы (например, "Группа1"), можно выполнить запрос, который выводит все экзамены, связанные с этой группой:

sql

SELECT e.exam\_date, e.classroom, d.name AS discipline, e.examiner

FROM Exams e

JOIN Disciplines d ON e.discipline\_code = d.discipline\_code

WHERE e.group\_code = 'Группа1'

ORDER BY e.exam\_date;

Этот запрос создает расписание экзаменов для группы "Группа1", сортируя по дате экзаменов.

3. Создание упорядоченных списков

Студенты групп ‘С’ второго курса (без находящихся в академическом отпуске)

Для того чтобы создать список студентов группы "С" второго курса, которые не находятся в академическом отпуске, можно использовать следующий запрос:

sql

SELECT full\_name, record\_book\_number, average\_score

FROM Students

WHERE group\_code = 'С' AND course = 2 AND academic\_leave = 0

ORDER BY full\_name;

Этот запрос возвращает список студентов второго курса группы "С", отсортированный по ФИО, и исключает студентов, находящихся в академическом отпуске.

Студенты с наибольшим средним баллом в своей группе

Для того чтобы найти студентов с наибольшим средним баллом в каждой группе, можно воспользоваться запросом с использованием подзапроса и группировки:

sql

SELECT s.full\_name, s.record\_book\_number, s.average\_score, s.group\_code

FROM Students s

WHERE s.average\_score = (

SELECT MAX(average\_score)

FROM Students

WHERE group\_code = s.group\_code

)

ORDER BY s.group\_code;

Этот запрос выбирает студентов с наибольшим средним баллом в каждой группе и сортирует их по группе.-->