**Документация по структуре базы данных.**

**Таблица Students**

* **StudentID (INT)**: Уникальный идентификатор студента (первичный ключ).
* **FirstName (VARCHAR(50))**: Имя студента.
* **LastName (VARCHAR(50))**: Фамилия студента.
* **BirthDate (DATE)**: Дата рождения студента.
* **ContactNumber (VARCHAR(20))**: Контактный номер студента.
* **Уникальный ключ unique\_student (FirstName, LastName, BirthDate)**: Гарантирует уникальность комбинации Имя, Фамилия и Дата рождения.

**Таблица Teachers**

* **TeacherID (INT)**: Уникальный идентификатор преподавателя (первичный ключ).
* **FirstName (VARCHAR(50))**: Имя преподавателя.
* **LastName (VARCHAR(50))**: Фамилия преподавателя.
* **BirthDate (DATE)**: Дата рождения преподавателя.
* **ContactNumber (VARCHAR(20))**: Контактный номер преподавателя.
* **Уникальный ключ unique\_teacher (FirstName, LastName, BirthDate)**: Гарантирует уникальность комбинации Имя, Фамилия и Дата рождения.

**Таблица Subjects**

* **SubjectID (INT)**: Уникальный идентификатор предмета (первичный ключ).
* **SubjectName (VARCHAR(50))**: Название предмета.
* **Уникальный ключ unique\_subject (SubjectName)**: Гарантирует уникальность названия предмета.

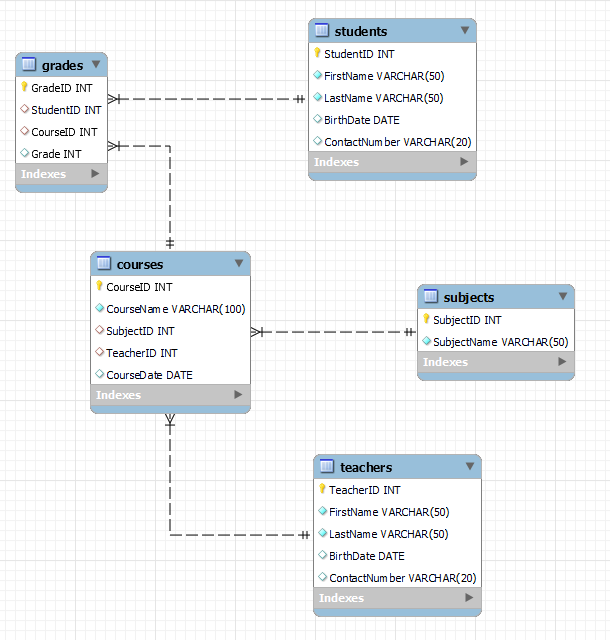
**Таблица Courses**

* **CourseID (INT)**: Уникальный идентификатор курса (первичный ключ).
* **CourseName (VARCHAR(100))**: Название курса.
* **SubjectID (INT)**: Внешний ключ, связывается с SubjectID в таблице Subjects.
* **TeacherID (INT)**: Внешний ключ, связывается с TeacherID в таблице Teachers.
* **CourseDate (DATE)**: Дата проведения курса.
* **Уникальный ключ unique\_course (CourseName, SubjectID)**: Гарантирует уникальность комбинации Название курса и SubjectID.

**Таблица Grades**

* **GradeID (INT):** Уникальный идентификатор оценки (первичный ключ).
* **StudentID (INT)**: Внешний ключ, связывается с StudentID в таблице Students.
* **CourseID (INT)**: Внешний ключ, связывается с CourseID в таблице Courses.
* **Grade (INT CHECK (Grade BETWEEN 1 AND 5))**: Оценка студента, ограниченная от 1 до 5.
* **Уникальный ключ unique\_grade (StudentID, CourseID)**: Гарантирует уникальность комбинации StudentID и CourseID.

**Модель базы данных:**

****

1. **Задайте возможность:**
   * **выводить список студентов по определённому предмету;**

SELECT Students.StudentID,

Students.FirstName,

Students.LastName

FROM Students

JOIN Grades ON Students.StudentID = Grades.StudentID

JOIN Courses ON Grades.CourseID = Courses.CourseID

WHERE CourseName = 'SQL';

* + **выводить список предметов, которые преподаёт конкретный преподаватель;**

SELECT Subjects.SubjectID,

Subjects.SubjectName

FROM Subjects

JOIN Courses ON Subjects.SubjectID = Courses.SubjectID

WHERE Courses.TeacherID = 7;

* + **выводить средний балл студента по всем предметам;**

SELECT Students.StudentID,

Students.FirstName,

Students.LastName,

AVG(Grades.Grade) AS AverageGrade

FROM Students

JOIN Grades ON Students.StudentID = Grades.StudentID

GROUP BY Students.StudentID;

* + **выводить рейтинг преподавателей по средней оценке студентов;**

SELECT Teachers.TeacherID,

Teachers.FirstName,

Teachers.LastName,

AVG(Grades.Grade) AS AverageGrade

FROM Teachers

JOIN Courses ON Teachers.TeacherID = Courses.TeacherID

JOIN Grades ON Courses.CourseID = Grades.CourseID

WHERE Courses.CourseDate <= CURDATE()

GROUP BY Teachers.TeacherID

ORDER BY AverageGrade DESC;

* + **выводить список преподавателей, которые преподавали более 3 предметов за последний год;**

SELECT Teachers.TeacherID,

Teachers.FirstName,

Teachers.LastName

FROM Teachers

JOIN Courses ON Teachers.TeacherID = Courses.TeacherID

WHERE Courses.CourseDate BETWEEN DATE\_SUB(CURDATE(), INTERVAL 1 YEAR) AND CURDATE()

GROUP BY Teachers.TeacherID

HAVING COUNT(DISTINCT Courses.SubjectID) > 3;

* + **выводить список студентов, которые имеют средний балл выше 4 по математическим предметам, но ниже 3 по гуманитарным;**

SELECT Students.StudentID,

Students.FirstName,

Students.LastName

FROM Students

JOIN Grades ON Students.StudentID = Grades.StudentID

JOIN Courses ON Grades.CourseID = Courses.CourseID

JOIN Subjects ON Courses.SubjectID = Subjects.SubjectID

WHERE Subjects.SubjectName = 'Математика'

GROUP BY Students.StudentID

HAVING AVG (Grades.Grade) > 4

INTERSECT

SELECT Students.StudentID,

Students.FirstName,

Students.LastName

FROM Students

JOIN Grades ON Students.StudentID = Grades.StudentID

JOIN Courses ON Grades.CourseID = Courses.CourseID

JOIN Subjects ON Courses.SubjectID = Subjects.SubjectID

WHERE Subjects.SubjectName = 'Гуманитарные предметы'

GROUP BY Students.StudentID

HAVING AVG(Grades.Grade) < 3;

* + **определить предметы, по которым больше всего двоек в текущем семестре;**

SELECT Subjects.SubjectID,

Subjects.SubjectName,

COUNT(\*) AS Cnt

FROM Subjects

JOIN Courses ON Subjects.SubjectID = Courses.SubjectID

JOIN Grades ON Courses.CourseID = Grades.CourseID

WHERE Courses.CourseDate BETWEEN '2023-02-01' AND '2023-06-01'

AND Grades.Grade = 2

GROUP BY Subjects.SubjectID

ORDER BY Cnt DESC

LIMIT 1;

* + **выводить студентов, которые получили высший балл по всем своим экзаменам, и преподавателей, которые вели эти предметы;**

SELECT DISTINCT

Students.StudentID,

Students.FirstName AS StudentFirstName,

Students.LastName AS StudentLastName,

Teachers.TeacherID,

Teachers.FirstName AS TeacherFirstName,

Teachers.LastName AS TeacherLastName

FROM Students

JOIN Grades ON Students.StudentID = Grades.StudentID

JOIN Courses ON Grades.CourseID = Courses.CourseID

JOIN Teachers ON Courses.TeacherID = Teachers.TeacherID

WHERE Grades.Grade = 5;

* + **просматривать изменение среднего балла студента по годам обучения;**

SELECT Students.StudentID,

Students.FirstName,

Students.LastName,

AVG (CASE WHEN YEAR(Courses.CourseDate) = 2020 THEN Grades.Grade END) AS year\_2020,

AVG (CASE WHEN YEAR(Courses.CourseDate) = 2021 THEN Grades.Grade END) AS year\_2021,

AVG (CASE WHEN YEAR(Courses.CourseDate) = 2022 THEN Grades.Grade END) AS year\_2022

FROM Students

JOIN Grades ON Students.StudentID = Grades.StudentID

JOIN Courses ON Grades.CourseID = Courses.CourseID

GROUP BY Students.StudentID;

* + **определить группы, в которых средний балл выше, чем в других, по аналогичным предметам, чтобы выявить лучшие методики преподавания или особенности состава группы.**

SELECT Courses.CourseID,

Subjects.SubjectID,

AVG(Grades.Grade) AS avg\_grade

FROM Courses

JOIN Grades ON Courses.CourseID = Grades.CourseID

JOIN Subjects ON Subjects.SubjectID = Courses.SubjectID

GROUP BY Courses.CourseID, Subjects.SubjectID

ORDER BY avg\_grade DESC

LIMIT 1;

1. **Вставьте записи о новом студенте с его личной информацией: ФИО, дата рождения, контактные данные и др.**

INSERT INTO Students(FirstName, LastName, BirthDate, ContactNumber)

VALUES ('Иван', 'Петров', '1999-01-01', '1234567890');

1. **Добавьте возможность:**
   1. **обновления контактной информации преподавателя, например, электронной почты или номера телефона, на основе его идентификационного номера или ФИО;**

UPDATE Teachers

SET ContactNumber = '000-000-000'

WHERE TeacherID = 1;

* 1. **удаления записи о предмете, который больше не преподают в учебном заведении. Учтите возможные зависимости, такие как оценки студентов по этому предмету;**

SET @ToDelete = 1;

DELETE FROM Grades

WHERE CourseID IN (SELECT CourseID FROM Courses WHERE SubjectID = @ToDelete);

DELETE FROM Courses

WHERE SubjectID = @ToDelete;

DELETE FROM Subjects

WHERE SubjectID = @ToDelete;