ТЕХНИЧЕСКИ УНИВЕРСИТЕТ - СОФИЯ

Катедра „Компютърни системи“

КУРСОВ ПРОЕКТ

ПО БАЗИ ДАННИ

Студент: Радослав Тихомиров Здравков

ФАК. № 121221004 Група: 47Б

Graphical user interface, text

Description automatically generated

1.Да се проектира база от данни и да се представи ER диаграма със съответни CREATE TABLE заявки за средата MySQL.

2.Напишете заявка, в която демонстрирате SELECT с ограничаващо условие по избор.

3.Напишете заявка, в която използвате агрегатна функция и GROUP BY по ваш избор.

4.Напишете заявка, в която демонстрирате INNER JOIN по ваш избор.

5.Напишете заявка, в която демонстрирате OUTER JOIN по ваш избор.

6.Напишете заявка, в която демонстрирате вложен SELECT по ваш избор.

7.Напишете заявка, в която демонстрирате едновременно JOIN и агрегатна функция.

8.Създайте тригер по ваш избор.

9.Създайте процедура, в която демонстрирате използване на курсор.

1. **Да се проектира база от данни и да се представи ER диаграма със съответни CREATE TABLE заявки за средата MySQL.**

От условието става ясно, че основните обекти в нашата база данни ще са студентите, треньорите, спортовете и стипендиите. За тях трябва да създадем таблици: students, coaches, sports, scholarships.Връзките ще са М:1 между студентите и стипендиите, 1:1 между треньорите и спортовете и М:М между студентите и спортовете. За да осъществим М:М връзката ще ни е необходима междинна таблица student\_sport. Таблицата students ще съхранява информация за id, име, ЕГН, години, оценка, телефонен номер, адрес и id на стипендия; таблицата coaches - id, име, ЕГН, почасово заплащане, заплата(изчислена на база почасовото плащане) и id на спорт; таблицата sports – id и име; scholarships – id и тип(никаква; за успех между 3.50 и 5.49; за успех над 5.50). Базата данни има следната ER диаграма:

Diagram

Description automatically generated

Заявки за създаване на базата данни и таблиците:

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

INSERT заявки за попълване на таблиците:

Text

Description automatically generated

Graphical user interface, text

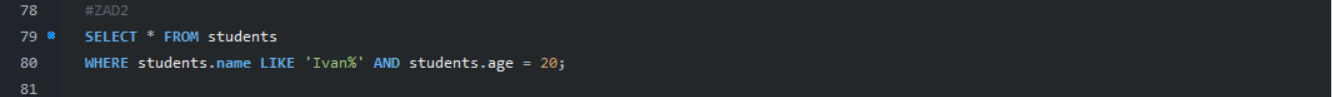
Description automatically generated with medium confidence

Диаграма от workbench след създаване на таблиците:

Diagram

Description automatically generated

**2.Напишете заявка, в която демонстрирате SELECT с ограничаващо условие по избор.**

****

Заявката извлича ученици с име “Ivan” и са на 20 години.

****

**3.Напишете заявка, в която използвате агрегатна функция и GROUP BY по ваш избор.**

**Graphical user interface, application

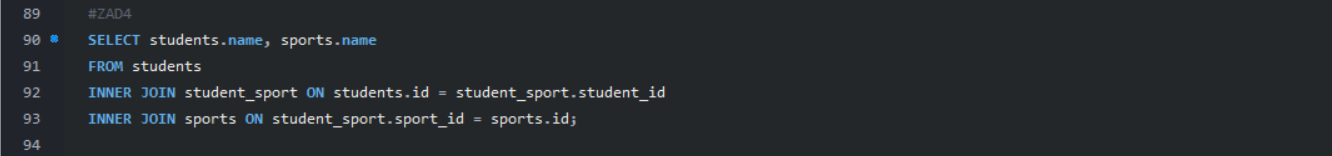
Description automatically generated**

Използваме агрегатната функция COUNT, за да покажем по колко спорта посещава всеки студент, като групираме по име на студента.

**Graphical user interface, text, application

Description automatically generated**

**4.Напишете заявка, в която демонстрирате INNER JOIN поваш избор.**

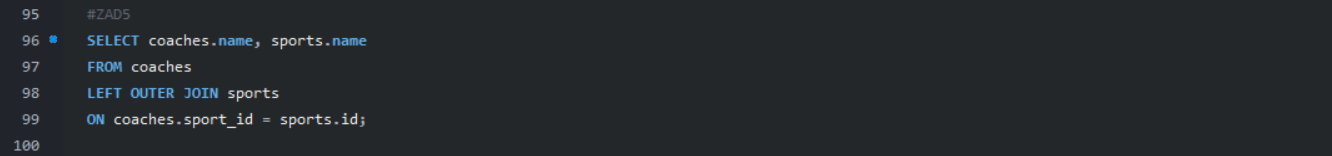
****

Използваме вътрешен JOIN, за да извлечем името на студент и спорта, който тренира.

**Graphical user interface, text, application

Description automatically generated**

**5.Напишете заявка, в която демонстрирате OUTER JOIN по ваш избор.**

****

Използваме LEFT JOIN и виждаме всичко от лявата(първата) таблица, въпреки че в дясната имаме стойност NULL.

**Graphical user interface, text, application

Description automatically generated**

**6.Напишете заявка, в която демонстрирате вложен SELECT по ваш избор.**

**Shape

Description automatically generated with medium confidence**

Използваме вложен SELECT, за да вземем id-то на студента от таблицата student\_sport и с него да свържем таблиците students и sports.

**Graphical user interface, text, application

Description automatically generated**

**7.Напишете заявка, в която демонстрирате едновременно JOIN и агрегатна функция.**

**Graphical user interface

Description automatically generated with medium confidence**

Заявката изкарва имената на спортовете и колко студенти ги практикуват, като за целта сме свързали таблиците students и sports и сме използвали агрегатната функция COUNT. Групирали сме отново по име, но този път на спорта.

**Graphical user interface, text, application

Description automatically generated**

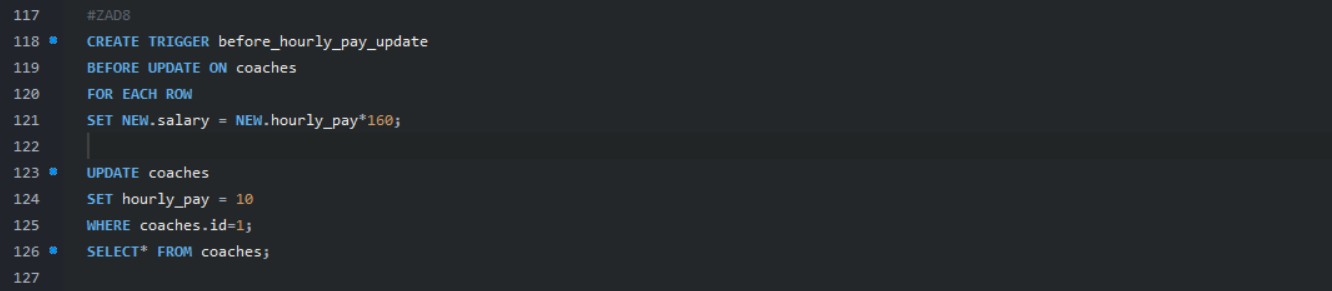
**8.Създайте тригер по ваш избор.**

Почасовото заплащане и заплатата на треньор с id = 1 преди извършването на update заявка:

**Graphical user interface, text, application

Description automatically generated**

Тригерът се активира преди ъпдейт на таблицата coaches и преизчислява заплатата, ако имаме промяна в сумата, която треньорите получават на час. Заплатата се получава като умножим сумата, която всеки треньор взима на час по 160 часа(4 седмици в месеца по 40 работни часа на седмица).

****

Почасовото заплащане и заплатата на треньор с id = 1 след извършването на update заявка и активиране на trigger-a:

Graphical user interface, table

Description automatically generated

**9.Създайте процедура, в която демонстрирате използване на курсор.**

Създаваме курсор, който да запазва в 2 променливи данните за име и ЕГН на студент, което се случва в цикъл докато не се достигне NOT FOUND и не се напусне цикъла. На всяка итерация на цикъла запазваме извлечените от курсора данни в създадената от нас временна таблица, за да може преди приключване на процедурата да видим дали курсорът работи както сме очаквали.

**Text

Description automatically generated**

Съдържанието на временната таблица, което получаваме след като извикаме процедурата:

Graphical user interface, text, application

Description automatically generated