**Предметная область**

Университет состоит из совокупности факультетов (школ). Поступление абитуриентов осуществляется на образовательные программы по результатам Единого государственного экзамена (ЕГЭ). Каждая образовательная программа относится к определенному факультету, для нее определены необходимые для поступления предметы ЕГЭ, минимальный балл по этим предметам, а также план набора (количество мест) на образовательную программу.

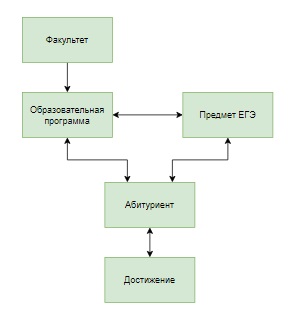
В приемную комиссию абитуриенты подают заявления на образовательную программу, каждый абитуриент может выбрать несколько образовательных программ (но не более трех). В заявлении указывается фамилия, имя, отчество абитуриента, а также его достижения: получил ли он медаль за обучение в школе, имеет ли значок ГТО и пр. При этом за каждое достижение определен дополнительный балл. Абитуриент предоставляет сертификат с результатами сдачи  ЕГЭ. Если абитуриент выбирает образовательную программу, то у него обязательно должны быть сданы предметы, определенные на эту программу, причем балл должен быть не меньше минимального по данному предмету.

Зачисление абитуриентов осуществляется так: сначала вычисляется сумма баллов по предметам на каждую образовательную программу, добавляются баллы достижения, затем абитуриенты сортируются в порядке убывания суммы баллов и отбираются первые по количеству мест, определенному планом набора.

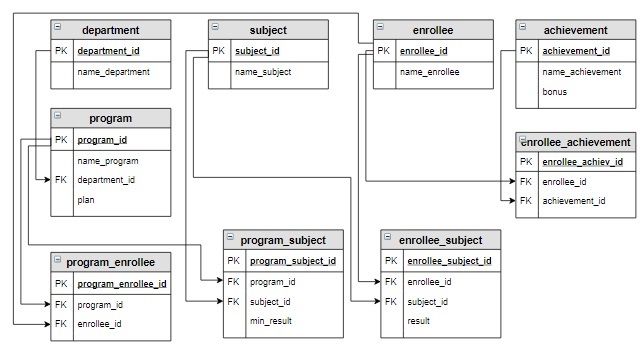
**Запросы для предметной области:**

1. [Вывести абитуриентов, которые хотят поступать на определенную образовательную программу](https://stepik.org/lesson/310418/step/2?unit=292724).
2. [Вывести образовательные программы, на которые для поступления необходим определенный предмет ЕГЭ](https://stepik.org/lesson/310418/step/3?unit=292724).
3. [Вывести статистическую информацию по каждому предмету  ЕГЭ](https://stepik.org/lesson/310418/step/4?unit=292724).
4. [Вывести образовательные программы, минимальные баллы по каждому предмету которых, превышают заданное значение](https://stepik.org/lesson/310418/step/5?unit=292724).
5. [Вывести образовательные программы. которые имеют самый большой план набора](https://stepik.org/lesson/310418/step/6?unit=292724).
6. [Посчитать, сколько дополнительных баллов получит каждый абитуриент](https://stepik.org/lesson/310418/step/7?unit=292724).
7. [Посчитать конкурс на каждую образовательную программу](https://stepik.org/lesson/310418/step/8?unit=292724).
8. [Вывести образовательные программы, на которые для поступления необходимы два определенных предмета ЕГЭ](https://stepik.org/lesson/310418/step/9?unit=292724).
9. [Посчитать количество баллов каждого абитуриента на каждую образовательную программу по результатам ЕГЭ](https://stepik.org/lesson/310418/step/10?unit=292724).
10. [Вывести абитуриентов, которые не могут быть зачислены на образовательную программу](https://stepik.org/lesson/310418/step/11?unit=292724).

**Концептуальная схема базы данных**



**Логическая схема базы данных**



**Структура и наполнение таблиц**

Таблица**department**:

|  |  |
| --- | --- |
| **department\_id** | **name\_department** |
| **INT PRIMARY KEY AUTO\_INCREMENT** | **VARCHAR(30)** |
| 1 | Инженерная школа |
| 2 | Школа естественных наук |

Таблица**subject**:

|  |  |
| --- | --- |
| **subject\_id** | **name\_subject** |
| **INT PRIMARY KEY AUTO\_INCREMENT** | **VARCHAR(30)** |
| 1 | Русский язык |
| 2 | Математика |
| 3 | Физика |
| 4 | Информатика |

Таблица **program** (в последнем столбце указан план набора абитуриентов на образовательную программу):

|  |  |  |  |
| --- | --- | --- | --- |
| **program\_id** | **name\_program** | **department\_id** | **plan** |
| **INT PRIMARY KEY AUTO\_INCREMENT** | **VARCHAR(50)** | **INT** | **INT** |
| 1 | Прикладная математика и информатика | 2 | 2 |
| 2 | Математика и компьютерные науки | 2 | 1 |
| 3 | Прикладная механика | 1 | 2 |
| 4 | Мехатроника и робототехника | 1 | 3 |

Таблица **enrollee**:

|  |  |
| --- | --- |
| **enrollee\_id** | **name\_enrollee** |
| **INT PRIMARY KEY AUTO\_INCREMENT** | **VARCHAR(50)** |
| 1 | Баранов Павел |
| 2 | Абрамова Катя |
| 3 | Семенов Иван |
| 4 | Яковлева Галина |
| 5 | Попов Илья |
| 6 | Степанова Дарья |

Таблица **achievement**(таблица включает все достижения, которые учитываются при поступлении в университет, в последнем столбце указывается количество баллов, которое добавляется к сумме баллов по предметам ЕГЭ при расчете общего балла абитуриента):

|  |  |  |
| --- | --- | --- |
| **achievement\_id** | **name\_achievement** | **bonus** |
| **INT PRIMARY KEY AUTO\_INCREMENT** | **VARCHAR(30)** | **INT** |
| 1 | Золотая медаль | 5 |
| 2 | Серебряная медаль | 3 |
| 3 | Золотой значок ГТО | 3 |
| 4 | Серебряный значок ГТО | 1 |

Таблица **enrollee\_achievement**(в таблице содержится информация о том, какие достижения имеют абитуриенты):

|  |  |  |
| --- | --- | --- |
| **enrollee\_achiev\_id** | **enrollee\_id** | **achievement\_id** |
| **INT PRIMARY KEY AUTO\_INCREMENT** | **INT** | **INT** |
| 1 | 1 | 2 |
| 2 | 1 | 3 |
| 3 | 3 | 1 |
| 4 | 4 | 4 |
| 5 | 5 | 1 |
| 6 | 5 | 3 |

Таблица **program\_subject**(в таблице указано, какие предметы ЕГЭ необходимы для поступления на каждую программу, в последнем столбце – минимальный балл по каждому предмету для образовательной программы):

|  |  |  |  |
| --- | --- | --- | --- |
| **program\_subject\_id** | **program\_id** | **subject\_id** | **min\_result** |
| **INT PRIMARY KEY AUTO\_INCREMENT** | **INT** | **INT** | **INT** |
| 1 | 1 | 1 | 40 |
| 2 | 1 | 2 | 50 |
| 3 | 1 | 4 | 60 |
| 4 | 2 | 1 | 30 |
| 5 | 2 | 2 | 50 |
| 6 | 2 | 4 | 60 |
| 7 | 3 | 1 | 30 |
| 8 | 3 | 2 | 45 |
| 9 | 3 | 3 | 45 |
| 10 | 4 | 1 | 40 |
| 11 | 4 | 2 | 45 |
| 12 | 4 | 3 | 45 |

Таблица **program\_enrollee**(таблица включает информацию, на какую образовательную программу хочет поступить абитуриент):

|  |  |  |
| --- | --- | --- |
| **program\_enrollee\_id** | **program\_id** | **enrollee\_id** |
| **INT PRIMARY KEY AUTO\_INCREMENT** | **INT** | **INT** |
| 1 | 3 | 1 |
| 2 | 4 | 1 |
| 3 | 1 | 1 |
| 4 | 2 | 2 |
| 5 | 1 | 2 |
| 6 | 1 | 3 |
| 7 | 2 | 3 |
| 8 | 4 | 3 |
| 9 | 3 | 4 |
| 10 | 3 | 5 |
| 11 | 4 | 5 |
| 12 | 2 | 6 |
| 13 | 3 | 6 |
| 14 | 4 | 6 |

Таблица **enrollee\_subject**(баллы ЕГЭ каждого абитуриента):

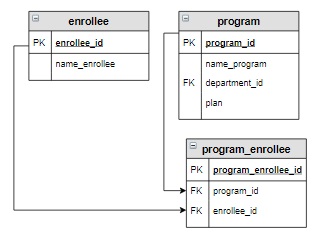
|  |  |  |  |
| --- | --- | --- | --- |
| **enrollee\_subject\_id** | **enrollee\_id** | **subject\_id** | **result** |
| **INT PRIMARY KEY AUTO\_INCREMENT** | **INT** | **INT** | **INT** |
| 1 | 1 | 1 | 68 |
| 2 | 1 | 2 | 70 |
| 3 | 1 | 3 | 41 |
| 4 | 1 | 4 | 75 |
| 5 | 2 | 1 | 75 |
| 6 | 2 | 2 | 70 |
| 7 | 2 | 4 | 81 |
| 8 | 3 | 1 | 85 |
| 9 | 3 | 2 | 67 |
| 10 | 3 | 3 | 90 |
| 11 | 3 | 4 | 78 |
| 12 | 4 | 1 | 82 |
| 13 | 4 | 2 | 86 |
| 14 | 4 | 3 | 70 |
| 15 | 5 | 1 | 65 |
| 16 | 5 | 2 | 67 |
| 17 | 5 | 3 | 60 |
| 18 | 6 | 1 | 90 |
| 19 | 6 | 2 | 92 |
| 20 | 6 | 3 | 88 |
| 21 | 6 | 4 | 94 |

**Пояснение.**Для всех внешних ключей в базе данных установлено **ON DELETE CASCADE**.

**Задание**

Вывести абитуриентов, которые хотят поступать на образовательную программу «Мехатроника и робототехника» в отсортированном по фамилиям виде.

**Фрагмент логической схемы базы данных:**



**Связанные шаги**

**Текст задания**(чтобы не прокручивать страницу) :

*Вывести абитуриентов, которые хотят поступать на образовательную программу «Мехатроника и робототехника» в отсортированном по фамилиям виде.*

**Результат**

**Наполнение таблиц program, enrollee, program\_enrollee**

таблица program

+------------+-------------------------------------+---------------+------+

| program\_id | name\_program | department\_id | plan |

+------------+-------------------------------------+---------------+------+

| 1 | Прикладная математика и информатика | 2 | 1 |

| 2 | Математика и компьютерные науки | 2 | 2 |

| 3 | Прикладная механика | 1 | 2 |

| 4 | Мехатроника и робототехника | 1 | 3 |

+------------+-------------------------------------+---------------+------+

таблица enrollee

+-------------+-----------------+

| enrollee\_id | name\_enrollee |

+-------------+-----------------+

| 1 | Баранов Павел |

| 2 | Абрамова Катя |

| 3 | Семенов Иван |

| 4 | Яковлева Галина |

| 5 | Попов Илья |

| 6 | Степанова Дарья |

+-------------+-----------------+

таблица program\_enrollee

+---------------------+------------+-------------+

| program\_enrollee\_id | program\_id | enrollee\_id |

+---------------------+------------+-------------+

| 1 | 3 | 1 |

| 2 | 4 | 1 |

| 3 | 1 | 1 |

| 4 | 2 | 2 |

| 5 | 1 | 2 |

| 6 | 1 | 3 |

| 7 | 2 | 3 |

| 8 | 4 | 3 |

| 9 | 3 | 4 |

| 10 | 3 | 5 |

| 11 | 4 | 5 |

| 12 | 2 | 6 |

| 13 | 3 | 6 |

| 14 | 4 | 6 |

+---------------------+------------+-------------+

1

SELECT name\_enrollee

2

FROM program

3

    JOIN program\_enrollee ON program.program\_id = program\_enrollee.program\_id

4

    JOIN enrollee ON enrollee.enrollee\_id = program\_enrollee.enrollee\_id

5

WHERE name\_program LIKE 'Мехатроника и робототехника'

6

ORDER BY 1;

7

​

8

SELECT name\_enrollee

9

FROM enrollee

10

WHERE enrollee\_id IN (

11

   SELECT enrollee\_id FROM program\_enrollee WHERE program\_id = (

12

    SELECT program\_id FROM program WHERE name\_program LIKE 'Мехатроника и робототехника'))

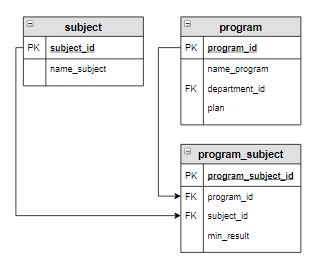
13

ORDER BY 1;

**Задание**

Вывести образовательные программы, на которые для поступления необходим предмет «Информатика». Программы отсортировать в обратном алфавитном порядке.

**Фрагмент логической схемы базы данных:**



**Связанные шаги**

**Текст задания**(чтобы не прокручивать страницу) :

*Вывести образовательные программы, на которые для поступления необходим предмет «Информатика». Программы отсортировать в обратном алфавитном порядке.*

1

SELECT name\_program

2

FROM

3

  subject

4

   JOIN program\_subject ON subject.subject\_id = program\_subject.subject\_id

5

   JOIN program ON program.program\_id = program\_subject.program\_id

6

WHERE name\_subject LIKE 'Информатика'

7

ORDER BY 1 DESC;

8

​

9

SELECT name\_program

10

FROM program

11

WHERE program\_id IN(

12

   SELECT program\_id FROM program\_subject WHERE subject\_id = (

13

    SELECT subject\_id FROM subject WHERE name\_subject = 'Информатика'))

14

ORDER BY 1 DESC;

15

​

16

SELECT\*FROM subject;

17

SELECT\*FROM program\_subject;

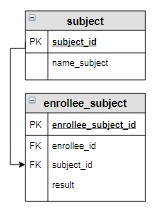
18

SELECT\*FROM program;

**Задание**

Выведите количество абитуриентов, сдавших ЕГЭ по каждому предмету, максимальное, минимальное и среднее значение баллов по предмету ЕГЭ. Вычисляемые столбцы назвать **Количество**, **Максимум**, **Минимум**, **Среднее**. Информацию отсортировать по названию предмета в алфавитном порядке, среднее значение округлить до одного знака после запятой.

**Фрагмент логической схемы базы данных**



**Связанные шаги**

**Текст задания**(чтобы не прокручивать страницу) :

*Выведите количество абитуриентов, сдавших ЕГЭ по каждому предмету, максимальное, минимальное и среднее значение баллов по предмету ЕГЭ. Вычисляемые столбцы назвать****Количество****,****Максимум****,****Минимум****,****Среднее****. Информацию отсортировать по названию предмета в алфавитном порядке, среднее значение округлить до одного знака после запятой.*

1

SELECT name\_subject,

2

       COUNT(enrollee\_id) AS 'Количество',

3

        MAX(result) AS 'Максимум',

4

         MIN(result) AS 'Минимум',

5

          ROUND(AVG(result), 1) AS 'Среднее'

6

FROM

7

  subject

8

   JOIN enrollee\_subject

9

   ON subject.subject\_id = enrollee\_subject.subject\_id

10

GROUP BY 1

11

ORDER BY 1;

12

​

13

SELECT\*FROM subject;

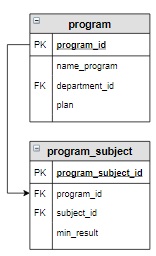
14

SELECT\*FROM enrollee\_subject;

**Задание**

Вывести образовательные программы, для которых минимальный балл ЕГЭ по каждому предмету больше или равен 40 баллам. Программы вывести в отсортированном по алфавиту виде.

**Фрагмент логической схемы базы данных**



**Связанные шаги**

**Текст задания**(чтобы не прокручивать страницу) :

*Вывести образовательные программы, для которых минимальный балл ЕГЭ по каждому предмету больше или равен 40 баллам. Программы вывести в отсортированном по алфавиту виде.*

**Результат**

**Наполнение таблиц program, program\_subject**

таблица program

+------------+-------------------------------------+---------------+------+

| program\_id | name\_program | department\_id | plan |

+------------+-------------------------------------+---------------+------+

| 1 | Прикладная математика и информатика | 2 | 1 |

| 2 | Математика и компьютерные науки | 2 | 2 |

| 3 | Прикладная механика | 1 | 2 |

| 4 | Мехатроника и робототехника | 1 | 3 |

+------------+-------------------------------------+---------------+------+

таблица program\_subject

+--------------------+------------+------------+------------+

| program\_subject\_id | program\_id | subject\_id | min\_result |

+--------------------+------------+------------+------------+

| 1 | 1 | 1 | 40 |

| 2 | 1 | 2 | 50 |

| 3 | 1 | 4 | 60 |

| 4 | 2 | 1 | 30 |

| 5 | 2 | 2 | 50 |

| 6 | 2 | 4 | 60 |

| 7 | 3 | 1 | 30 |

| 8 | 3 | 2 | 45 |

| 9 | 3 | 3 | 45 |

| 10 | 4 | 1 | 40 |

| 11 | 4 | 2 | 45 |

| 12 | 4 | 3 | 45 |

+--------------------+------------+------------+------------+

1

SELECT name\_program

2

FROM

3

  program

4

    JOIN program\_subject

5

     ON program.program\_id = program\_subject.program\_id

6

GROUP BY 1

7

HAVING MIN(min\_result) >= 40

8

ORDER BY 1;

9

​

10

SELECT name\_program

11

 FROM program

12

WHERE program\_id NOT IN (SELECT program\_id

13

                           FROM program\_subject

14

                          WHERE min\_result < 40)

15

ORDER BY 1;

16

​

17

select distinct name\_program from program inner join program\_subject using (program\_id)

18

where program\_id not in (select program\_id from program\_subject where min\_result <40)

19

order by name\_program;

20

​

21

SELECT DISTINCT name\_program

22

FROM

23

  program

24

    JOIN program\_subject

25

     ON program.program\_id = program\_subject.program\_id

26

WHERE program.program\_id NOT IN (SELECT program\_id FROM program\_subject WHERE min\_result < 40)

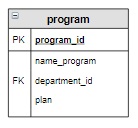
27

ORDER BY name\_program;

**Задание**

Вывести образовательные программы, которые имеют самый большой план набора, вместе с этой величиной.

**Фрагмент логической схемы базы данных:**



1

SELECT name\_program, plan

2

FROM program

3

WHERE plan = (SELECT MAX(plan) FROM program);

4

​

5

SELECT name\_program, plan

6

FROM program

7

ORDER BY 2 DESC

8

LIMIT 1;

9

​

10

SELECT name\_program, plan

11

FROM program

12

WHERE plan >= ALL(SELECT plan FROM program);

13

​

14

WITH p AS (

15

         SELECT name\_program, plan,

16

        RANK() OVER (ORDER BY plan DESC) AS g

17

         FROM program)

18

SELECT name\_program, plan

19

FROM p

20

WHERE g = 1;

21

​

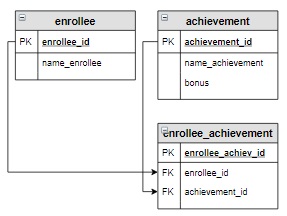
22

SELECT \* FROM program;

**Задание**

Посчитать, сколько дополнительных баллов получит каждый абитуриент. Столбец с дополнительными баллами назвать **Бонус**. Информацию вывести в отсортированном по фамилиям виде.

**Фрагмент логической схемы базы данных:**



**Пояснение**

**Связанные шаги**

**Текст задания**(чтобы не прокручивать страницу) :

*Посчитать, сколько дополнительных баллов получит каждый абитуриент. Столбец с дополнительными баллами назвать****Бонус****. Информацию вывести в отсортированном по фамилиям виде.*

1

SELECT name\_enrollee, SUM(IF(bonus IS NULL, 0, bonus)) AS 'Бонус'

2

FROM

3

  enrollee

4

   LEFT JOIN enrollee\_achievement

5

   ON enrollee.enrollee\_id = enrollee\_achievement.enrollee\_id

6

   LEFT JOIN achievement

7

   ON achievement.achievement\_id = enrollee\_achievement.achievement\_id

8

GROUP BY 1

9

ORDER BY 1;

10

​

11

SELECT \* FROM achievement;

12

SELECT \* FROM enrollee;

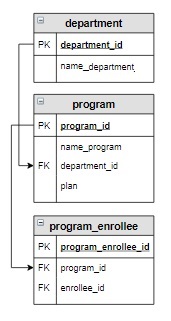
13

SELECT \* FROM enrollee\_achievement;

**Задание**

Выведите сколько человек подало заявление на каждую образовательную программу и конкурс на нее (число поданных заявлений деленное на количество мест по плану), округленный до 2-х знаков после запятой. В запросе вывести название факультета, к которому относится образовательная программа, название образовательной программы, план набора абитуриентов на образовательную программу (**plan**), количество поданных заявлений (**Количество**) и **Конкурс**. Информацию отсортировать в порядке убывания конкурса.

**Фрагмент логической схемы базы данных:**



**Пояснение**

**Связанные шаги**

**Текст задания**(чтобы не прокручивать страницу) :

*Выведите сколько человек подало заявление на каждую образовательную программу и конкурс на нее (число поданных заявлений деленное на количество мест по плану), округленный до 2-х знаков после запятой. В запросе вывести название факультета, к которому относится образовательная программа, название образовательной программы, план набора абитуриентов на образовательную программу (****plan****), количество поданных заявлений (****Количество****) и****Конкурс****. Информацию отсортировать в порядке убывания конкурса.*

1

SELECT name\_department,

2

      name\_program,

3

        plan,

4

         COUNT(enrollee\_id) AS 'Количество',

5

          ROUND(COUNT(enrollee\_id)/plan, 2) AS 'Конкурс'

6

FROM

7

  department

8

    JOIN program

9

     ON department.department\_id = program.department\_id

10

      JOIN program\_enrollee

11

       ON program.program\_id = program\_enrollee.program\_id

12

GROUP BY 1,2,3

13

ORDER BY 5 DESC;

**Задание**

Вывести образовательные программы, на которые для поступления необходимы предмет «Информатика» и «Математика» в отсортированном по названию программ виде.

**Логическая схемы базы данных (чтобы потренироваться выбирать таблицы для запроса):**

**Пояснение**

**Связанные шаги**

**Текст задания**(чтобы не прокручивать страницу) :

*Вывести образовательные программы, на которые для поступления необходимы предмет «Информатика» и «Математика» в отсортированном по названию программ виде.*

1

SELECT name\_program /\*COUNT(name\_subject) ТУТ ВИЩИТУЄ СКОЛЬКО ПРЕДМЕТОВ НА ОДНУ ПРОГРАМУ\*/

2

FROM

3

  program

4

    JOIN program\_subject

5

     ON program.program\_id = program\_subject.program\_id

6

      JOIN subject

7

       ON subject.subject\_id = program\_subject.subject\_id

8

        AND name\_subject IN ('Математика', 'Информатика')

9

GROUP BY 1

10

HAVING COUNT(name\_subject) = 2

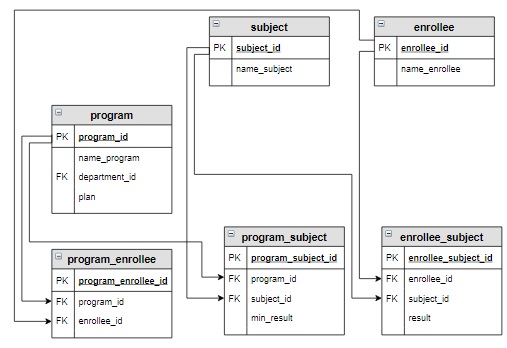
11

ORDER BY 1;

**Задание**

Посчитать количество баллов каждого абитуриента на каждую образовательную программу, на которую он подал заявление, по результатам ЕГЭ. В результат включить название образовательной программы, фамилию и имя абитуриента, а также столбец с суммой баллов, который назвать **itog**. Информацию вывести в отсортированном сначала по образовательной программе, а потом по убыванию суммы баллов виде.

**Фрагмент логической схемы базы данных:**



**Текст задания**(чтобы не прокручивать страницу) :

*Посчитать количество баллов каждого абитуриента на каждую образовательную программу, на которую он подал заявление, по результатам ЕГЭ. В результат включить название образовательной программы, фамилию и имя абитуриента, а также столбец с суммой баллов, который назвать****itog****. Информацию вывести в отсортированном сначала по образовательной программе, а потом по убыванию суммы баллов виде.*

1

SELECT name\_program,

2

      name\_enrollee,

3

        SUM(result) AS 'itog'

4

FROM

5

  program

6

    JOIN program\_enrollee

7

     ON program.program\_id = program\_enrollee.program\_id

8

      JOIN enrollee

9

       ON enrollee.enrollee\_id = program\_enrollee.enrollee\_id

10

        JOIN enrollee\_subject

11

         ON enrollee.enrollee\_id = enrollee\_subject.enrollee\_id

12

          JOIN subject

13

           ON subject.subject\_id = enrollee\_subject.subject\_id

14

            JOIN program\_subject

15

             ON subject.subject\_id = program\_subject.subject\_id

16

              AND program.program\_id = program\_subject.program\_id

17

GROUP BY 1, 2

18

ORDER BY 1, 3 DESC;

19

​

20

/\*select p.name\_program, e.name\_enrollee, sum(es.result) as itog

21

from program as p

22

join program\_enrollee as pe using(program\_id)

23

join enrollee as e using(enrollee\_id)

24

join enrollee\_subject as es using(enrollee\_id)

25

join subject as s using(subject\_id)

26

join program\_subject as ps using(subject\_id, program\_id)

27

group by p.name\_program, e.name\_enrollee

28

order by p.name\_program asc, itog desc;\*/

**не нашел такого решения в опубликованных, пусть будет и такой вариант**

**natural join делает присоединение по всем столбцам с одинаковым именем, поэтому последний join присоединяется по enrollee\_id и subject\_id автоматом**

**Natural Join: Guidelines:**

* The associated tables have one or more pairs of identically named columns.
* The columns must be the same data type.
* Don’t use ON clause in a NATURAL JOIN.

1

select name\_program, name\_enrollee, sum(result) as itog

2

from program\_enrollee

3

natural join program

4

natural join program\_subject

5

natural join enrollee

6

natural join enrollee\_subject

7

group by name\_enrollee, name\_program

8

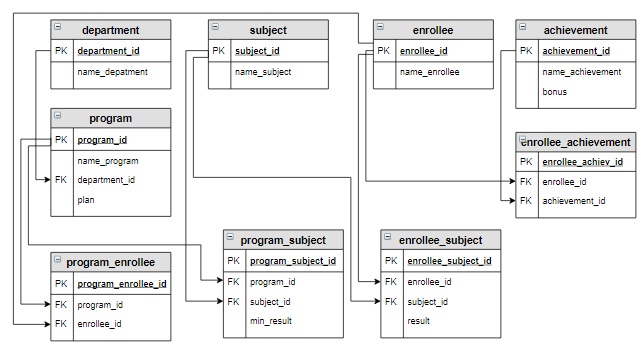
order by name\_program, itog desc;

**Задание**

Вывести название образовательной программы и фамилию тех абитуриентов, которые подавали документы на эту образовательную программу, но не могут быть зачислены на нее. Эти абитуриенты имеют результат по одному или нескольким предметам ЕГЭ, необходимым для поступления на эту образовательную программу, меньше минимального балла. Информацию вывести в отсортированном сначала по программам, а потом по фамилиям абитуриентов виде.

Например, Баранов Павел по «Физике» набрал 41 балл, а  для образовательной программы «Прикладная механика» минимальный балл по этому предмету определен в 45 баллов. Следовательно, абитуриент на данную программу не может поступить.

**Логическая схемы базы данных (чтобы потренироваться выбирать таблицы для запроса):**



Для этого задания в базу данных добавлена строка:

INSERT INTO enrollee\_subject (enrollee\_id, subject\_id, result) VALUES (2, 3, 41);

Добавлен человек, который сдавал **Физику**, но не подал документы ни на одну образовательную программу, где этот предмет нужен.

**Текст задания**(чтобы не прокручивать страницу) :

*Вывести название образовательной программы и фамилию тех абитуриентов, которые подавали документы на эту образовательную программу, но не могут быть зачислены на нее. Эти абитуриенты имеют результат по одному или нескольким предметам ЕГЭ, необходимым для поступления на эту образовательную программу, меньше минимального балла. Информацию вывести в отсортированном сначала по программам, а потом по фамилиям абитуриентов виде.*

*Например, Баранов Павел по «Физике» набрал 41 балл, а  для образовательной программы «Прикладная механика» минимальный балл по этому предмету определен в 45 баллов. Следовательно, абитуриент на данную программу не может поступить.*

1

SELECT name\_program,

2

      name\_enrollee

3

FROM

4

  program

5

    JOIN program\_enrollee

6

     ON program.program\_id = program\_enrollee.program\_id

7

      JOIN enrollee

8

       ON enrollee.enrollee\_id = program\_enrollee.enrollee\_id

9

        JOIN enrollee\_subject

10

         ON enrollee.enrollee\_id = enrollee\_subject.enrollee\_id

11

          JOIN subject

12

           ON subject.subject\_id = enrollee\_subject.subject\_id

13

            JOIN program\_subject

14

             ON subject.subject\_id = program\_subject.subject\_id

15

              AND program.program\_id = program\_subject.program\_id

16

WHERE result < min\_result

17

ORDER BY 1, 2;

**Задание**

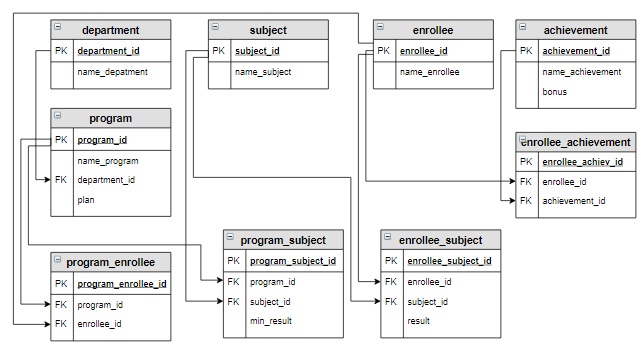
Придумайте один или несколько запросов на выборку для предметной области «Абитуриент» (в таблицы занесены данные, как на [первом шаге](https://stepik.org/lesson/310418/step/1?unit=292724) урока). Проверьте, правильно ли они работают.

При желании можно формулировку запросов  разместить в комментариях.

Размещенные задания можно реализовать для закрепления материала урока.

Оценивайте понравившиеся Вам запросы.

**Логическая схема базы данных:**



**Наполнение таблиц**

Enter an SQL query

Correct answer from **13,231** learners

Total **94%** of tries are correct



1

SELECT name\_enrollee, name\_program, MIN(result), name\_subject

2

FROM

3

  enrollee

4

    JOIN enrollee\_achievement

5

     ON enrollee.enrollee\_id = enrollee\_achievement.enrollee\_id

6

       JOIN achievement ON achievement.achievement\_id = enrollee\_achievement.achievement\_id

7

        AND bonus = 5

8

         JOIN program\_enrollee

9

          ON enrollee.enrollee\_id = program\_enrollee.enrollee\_id

10

           JOIN program

11

            ON program.program\_id = program\_enrollee.program\_id

12

             JOIN enrollee\_subject

13

              ON enrollee.enrollee\_id = enrollee\_subject.enrollee\_id

14

               JOIN subject

15

                ON subject.subject\_id = enrollee\_subject.subject\_id

16

GROUP BY 1, 2, 4

17

ORDER BY 2;