Платформа Stepik представляет довольно скудную навигацию по шагам модуля. Очень трудно вспомнить, на каком шаге изучался тот или иной материал. База данных этого урока позволяет с помощью запросов реализовать удобный поиск нужных шагов.

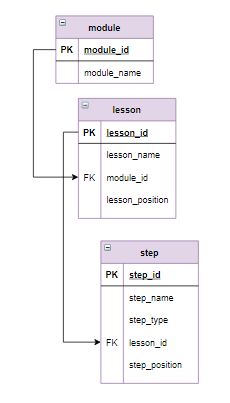
**Задание**

Отобрать все шаги, в которых рассматриваются вложенные запросы (то есть в названии шага упоминаются вложенные запросы). Указать к какому уроку и модулю они относятся. Для этого вывести 3 поля:

* в поле **Модуль** указать номер модуля и его название через пробел;
* в поле **Урок** указать номер модуля, порядковый номер урока (**lesson\_position**) через точку и название урока через пробел;
* в поле **Шаг** указать номер модуля, порядковый номер урока (**lesson\_position**) через точку, порядковый номер шага (**step\_position**) через точку и название шага через пробел.

Длину полей **Модуль** и **Урок**ограничить 19 символами, при этом слишком длинные надписи обозначить многоточием в конце (16 символов - это номер модуля или урока, пробел и  название **Урока** или **Модуля** к ним присоединить **"..."**). Информацию отсортировать по возрастанию номеров модулей, порядковых номеров уроков и порядковых номеров шагов.

**Фрагмент логической схемы базы данных:**



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

*Отобрать все шаги, в которых рассматриваются вложенные запросы (то есть в названии шага упоминаются вложенные запросы). Указать к какому уроку и модулю они относятся. Для этого вывести 3 поля:*

* *в поле****Модуль****указать номер модуля и его название через пробел;*
* *в поле****Урок****указать номер модуля, порядковый номер урока (****lesson\_position****) через точку и название урока через пробел;*
* *в поле****Шаг****указать номер модуля, порядковый номер урока (****lesson\_position****) через точку, порядковый номер шага (****step\_position****) через точку и название шага через пробел.*

*Длину полей****Модуль****и****Урок****ограничить 19 символами, при этом слишком длинные надписи обозначить многоточием в конце (16 символов - это номер модуля или урока, пробел и  название****Урока****или****Модуля,****к ним присоединить****"..."****). Информацию отсортировать по возрастанию номеров модулей, порядковых номеров уроков и порядковых номеров шагов.*

1

SELECT

2

  CONCAT(module.module\_id, ' ', LEFT(module\_name, 14), '...') AS 'Модуль',

3

    CONCAT(module.module\_id, '.', lesson\_position, ' ', LEFT(lesson\_name, 12), '...') AS 'Урок',

4

    CONCAT(module.module\_id, '.', lesson\_position, '.', step\_position, " ", step\_name) AS 'Шаг'

5

FROM

6

    module

7

     JOIN lesson

8

      ON module.module\_id = lesson.module\_id

9

       JOIN step

10

        ON step.lesson\_id = lesson.lesson\_id

11

WHERE step\_name LIKE '%вложен%запрос%'

12

ORDER BY 1,2,3;

13

​

14

​

15

WITH CTE\_step AS  -- Ищем шаги с вложенными запросами

16

(SELECT lesson\_id, step\_position, step\_name

17

FROM step

18

WHERE step\_name LIKE '%вложенн% запрос%'

19

),

20

​

21

CTE\_res AS  -- К шагам добавляем модуль и урок с требуемой нумерацией

22

(

23

SELECT CONCAT(module\_id, ' ', module\_name) AS Модуль,

24

CONCAT(module\_id, '.', lesson\_position, ' ', lesson\_name) AS Урок,

25

CONCAT(module\_id, '.', lesson\_position, '.', step\_position, ' ', step\_name) AS Шаг

26

FROM CTE\_step

27

INNER JOIN lesson USING (lesson\_id)

28

INNER JOIN module USING (module\_id)

29

ORDER BY module\_id, lesson\_position, step\_position

30

)

31

32

-- Результат: обрезаем тексты до 19 символов если нужно

33

SELECT IF(LENGTH(Модуль) > 19, CONCAT(LEFT(Модуль, 16),'...'), Модуль) AS Модуль,

34

IF(LENGTH(Урок) > 19, CONCAT(LEFT(Урок, 16),'...'), Урок) AS Урок,

35

Шаг

36

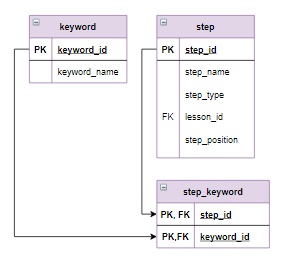
FROM CTE\_res

Еще одна возможность улучшить навигацию по курсу - это реализация поиска шагов по ключевым словам. Для этого необходимо создать таблицу с терминами **keyword**, а затем связать ее с таблицей **step** через вспомогательную таблицу **step\_keyword**. Каждая запись этой таблицы - это **id** шага и**id** встречающегося на этом шаге ключевого слова.

**Задание**

Заполнить таблицу **step\_keyword**следующим образом: если ключевое слово есть в названии шага, то включить в **step\_keyword** строку с **id** шага и **id** ключевого слова.

**Фрагмент логической схемы базы данных:**



**Пояснение**

**Связанные шаги**

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

*Заполнить таблицу****step\_keyword****следующим образом: если ключевое слово есть в названии шага, то включить в****step\_keyword****строку с****id****шага и****id****ключевого слова.*

1

INSERT INTO step\_keyword(step\_id, keyword\_id)

2

SELECT step\_id, keyword\_id

3

FROM step, keyword

4

WHERE INSTR(CONCAT(" ", step\_name, " "), CONCAT(" ", keyword\_name, " ")) > 0

5

       OR INSTR(CONCAT(" ", step\_name, " "), CONCAT(" ", keyword\_name, ",")) > 0

6

         OR INSTR(CONCAT(" ", step\_name, " "), CONCAT(" ", keyword\_name, "()")) > 0

7

ORDER BY 2;

8

SELECT \* FROM step\_keyword;

9

​

10

INSERT INTO step\_keyword

11

SELECT step\_id, keyword\_id

12

FROM   step JOIN keyword ON step\_name REGEXP CONCAT('\\b', keyword\_name, '\\b')

13

ORDER BY keyword\_id;

14

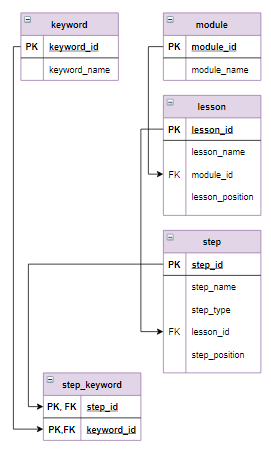
SELECT \* FROM step\_keyword;

**Задание**

Реализовать поиск по ключевым словам. Вывести шаги, с которыми связаны ключевые слова **MAX** и **AVG** одновременно. Для шагов указать **id** модуля, позицию урока в модуле, позицию шага в уроке через точку, после позиции шага перед заголовком - пробел. Позицию шага в уроке вывести в виде двух цифр (если позиция шага меньше 10, то перед цифрой поставить 0). Столбец назвать **Шаг**. Информацию отсортировать по первому столбцу в алфавитном порядке.

**Пояснение**

**Фрагмент логической схемы базы данных:**



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

*Реализовать поиск по ключевым словам. Вывести шаги, с которыми связаны ключевые слова****MAX****и****AVG****одновременно. Для шагов указать****id****модуля, позицию урока в модуле, позицию шага в уроке через точку, после позиции шага перед заголовком - пробел. Позицию шага в уроке вывести в виде двух цифр (если позиция шага меньше 10, то перед цифрой поставить 0). Столбец назвать****Шаг****. Информацию отсортировать по первому столбцу в алфавитном порядке.*

1

/\*SELECT CONCAT(module.module\_id, '.', lesson.lesson\_position, '.',

2

              IF(step.step\_position < 10, CONCAT('0', step.step\_position), step.step\_position),

3

              ' ', step\_name) AS 'Шаг'

4

FROM

5

  module

6

    JOIN lesson

7

    ON module.module\_id = lesson.module\_id

8

      JOIN step

9

      ON lesson.lesson\_id = step.lesson\_id

10

WHERE step\_name LIKE ("%MAX и AVG%") OR step\_name LIKE ('%SELECT')

11

ORDER BY module.module\_id,

12

        lesson.lesson\_position,

13

          step.step\_position;\*/

14

​

15

SELECT CONCAT(module.module\_id, '.', lesson.lesson\_position, '.',

16

              IF(step.step\_position < 10, CONCAT('0', step.step\_position), step.step\_position),

17

               ' ', step\_name) AS 'Шаг'

18

FROM step\_keyword

19

 JOIN step USING(step\_id)

20

 JOIN lesson USING(lesson\_id)

21

 JOIN module USING(module\_id)

22

 JOIN keyword USING(keyword\_id)

23

WHERE keyword\_name like '%AVG%' OR keyword\_name like '%MAX%'

24

GROUP BY step\_id

25

HAVING COUNT(keyword\_name) = 2

26

ORDER BY step\_position;

**Выборка данных по нескольким условиям, оператор CASE**

С помощью оператора **CASE** можно в зависимости от нескольких условий получить один из нескольких результатов.

Оператор **CASE** записывается в виде:

CASE

WHEN логическое\_выражение\_1 THEN выражение\_1

WHEN логическое\_выражение\_2 THEN выражение\_2

...

ELSE выражение\_else

END

Раздел**ELSE** является необязательным.

Выполняется оператор **CASE** так:

* вычисляется **логическое\_выражение\_1**, если оно истинно, то результатом оператора является **выражение\_1**, если ложно - выполнение оператора продолжается;
* вычисляется **логическое\_выражение\_2,** если оно истинно, то результатом оператора является **выражение\_2**, если ложно - выполнение оператора продолжается;
* если все логические выражения оказались ложными, то результат оператора - **выражение\_else**

**CASE**можно использовать в  **SELECT, UPDATE, DELETE, SET, WHERE, ORDER BY, HAVING** - всюду, где можно использовать выражения.

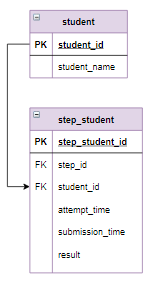
**Пример**

Отнести каждого студента к группе,  в зависимости от пройденных заданий:

|  |  |
| --- | --- |
| **Интервал** | **Группа** |
| от 0 до 10 | I |
| от 11 до 15 | II |
| от 16 до 27 | III |
| больше 27 | IV |

Пройденными считаются задания с хотя бы одним верным ответом. В таблице **step\_student** сохраняются все попытки пользователей, следовательно, могут быть пользователи, у которых на одно задание есть несколько верных попыток.

***Фрагмент логической схемы базы данных:***



***Шаг 1.***Выведем всех студентов и все шаги, которые они прошли с результатом "correct". Этот шаг обязателен, чтобы не учитывать  правильные решения несколько раз.

*Запрос:*

SELECT student\_name, step\_id

FROM

student

INNER JOIN step\_student USING(student\_id)

WHERE result = "correct"

GROUP BY student\_name, step\_id;

Результат:

Query result:

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

| student\_name | step\_id |

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

| student\_52 | 10 |

| student\_11 | 10 |

| student\_19 | 10 |

| student\_4 | 10 |

| student\_5 | 10 |

| student\_53 | 10 |

| student\_39 | 10 |

| student\_32 | 10 |

| student\_61 | 10 |

| student\_43 | 10 |

| student\_13 | 10 |

| student\_57 | 10 |

...

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

Affected rows: 1126.

***Шаг 2.***Посчитаем, сколько шагов прошел каждый студент.

*Запрос:*

SELECT student\_name, count(\*) as rate

FROM

    (

SELECT student\_name, step\_id

    FROM

student

INNER JOIN step\_student USING(student\_id)

    WHERE result = "correct"

    GROUP BY student\_name, step\_id

) query\_in

GROUP BY student\_name

ORDER BY 2;

Результат:

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

| student\_name | rate |

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

| student\_29 | 8 |

| student\_47 | 8 |

| student\_16 | 9 |

| student\_5 | 9 |

| student\_63 | 9 |

| student\_33 | 10 |

| student\_17 | 10 |

| student\_64 | 10 |

...

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

Affected rows: 64

***Шаг 3***. Отнести каждого студента к группе в зависимости от пройденных шагов.

*Запрос:*

SELECT student\_name, rate,

CASE

WHEN rate <= 10 THEN "I"

WHEN rate <= 15 THEN "II"

WHEN rate <= 27 THEN "III"

ELSE "IV"

END AS Группа

FROM

(

SELECT student\_name, count(\*) as rate

FROM

(

SELECT student\_name, step\_id

FROM

student

INNER JOIN step\_student USING(student\_id)

WHERE result = "correct"

GROUP BY student\_name, step\_id

) query\_in

GROUP BY student\_name

ORDER BY 2

) query\_in\_1;

*Результат:*

Query result:

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

| student\_name | rate | Группа |

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

| student\_29 | 8 | I |

| student\_47 | 8 | I |

| student\_16 | 9 | I |

| student\_5 | 9 | I |

| student\_63 | 9 | I |

| student\_33 | 10 | I |

| student\_17 | 10 | I |

| student\_64 | 10 | I |

| student\_58 | 10 | I |

| student\_38 | 10 | I |

| student\_12 | 11 | II |

| student\_10 | 11 | II |

...

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

Affected rows: 64

**Задание**

Посчитать, сколько студентов относится к каждой группе. Столбцы назвать **Группа**, **Интервал**, **Количество**. Указать границы интервала.

1

SELECT

2

   CASE

3

       WHEN rate <= 10 THEN "I"

4

       WHEN rate <= 15 THEN "II"

5

       WHEN rate <= 27 THEN "III"

6

       ELSE "IV"

7

   END AS Группа,

8

   CASE

9

       WHEN rate <= 10 THEN "от 0 до 10"

10

       WHEN rate <= 15 THEN "от 11 до 15"

11

       WHEN rate <= 27 THEN "от 16 до 27"

12

       ELSE "больше 27"

13

   END AS Интервал, COUNT(student\_name) AS Количество

14

FROM

15

   (

16

    SELECT student\_name, count(\*) as rate

17

    FROM

18

        (

19

         SELECT student\_name, step\_id

20

         FROM

21

            student

22

             INNER JOIN step\_student USING(student\_id)

23

         WHERE result = "correct"

24

         GROUP BY student\_name, step\_id

25

        ) query\_in

26

    GROUP BY student\_name

27

    ORDER BY 2

28

   ) query\_in\_1

29

GROUP BY Группа, Интервал;

/\*Шаг 1. Выведем всех студентов и все шаги, которые они прошли с результатом "correct". Этот шаг обязателен, чтобы не учитывать правильные решения несколько раз.\*/

/\*SELECT student\_name, step\_id

FROM

student

JOIN step\_student

ON student.student\_id = step\_student.student\_id

WHERE result LIKE "correct"

GROUP BY 1, 2;\*/

/\*Шаг 2. Посчитаем, сколько шагов прошел каждый студент.\*/

/\*SELECT student\_name, COUNT(step\_id) AS 'COUNT'

FROM (SELECT student\_name, step\_id

FROM student

JOIN step\_student

ON student.student\_id = step\_student.student\_id

WHERE result LIKE "correct"

GROUP BY 1, 2) table\_1

GROUP BY 1

ORDER BY 2;\*/

/\*Шаг 3. Отнести каждого студента к группе в зависимости от пройденных шагов.\*/

/\*SELECT student\_name, COUNT\_,

CASE

WHEN COUNT\_ <= 10 THEN 'I'

WHEN COUNT\_ <= 15 THEN 'II'

WHEN COUNT\_ <= 27 THEN 'III'

ELSE 'IV'

END AS 'Група'

FROM

(SELECT student\_name, COUNT(step\_id) AS 'COUNT\_'

FROM (SELECT student\_name, step\_id

FROM student

JOIN step\_student

ON student.student\_id = step\_student.student\_id

WHERE result LIKE "correct"

GROUP BY 1, 2) table\_1

GROUP BY 1

ORDER BY 2) table\_2;\*/

/\*Посчитать, сколько студентов относится к каждой группе. Столбцы назвать Группа, Интервал, Количество. Указать границы интервала.\*/

SELECT

CASE

WHEN COUNT\_ <= 10 THEN 'I'

WHEN COUNT\_ <= 15 THEN 'II'

WHEN COUNT\_ <= 27 THEN 'III'

ELSE 'IV'

END AS 'Группа',

CASE

WHEN COUNT\_ <= 10 THEN 'от 0 до 10'

WHEN COUNT\_ <= 15 THEN 'от 11 до 15'

WHEN COUNT\_ <= 27 THEN 'от 16 до 27'

ELSE 'больше 27'

END AS 'Интервал',

COUNT(student\_name) AS 'Количество'

FROM

(SELECT student\_name, COUNT(\*) AS 'COUNT\_'

FROM (SELECT student\_name, step\_id

FROM student

JOIN step\_student

ON student.student\_id = step\_student.student\_id

WHERE result = "correct"

GROUP BY 1, 2) table\_1

GROUP BY 1

ORDER BY 2) table\_2

GROUP BY 1, 2;