Урок 5

```
-- Разбор д/з

use fe37_38;

-- 25. Создать таблицу Jobs

-- Job_id целое число первичный ключ автоинкремент

create table jobs(

job_id integer primary key auto_increment);


-- 26. Добавить поля title строка не null, description строка не null, salary целое число

alter table jobs

add title varchar(128) not null,

add description varchar(128) not null,

add salary integer;


-- 27. Переименовать поле title на job_title, description на job_description

alter table jobs

change title job_title varchar(128) not null,

change description job_description varchar(128) not null;


-- 28. Удалить поле salary

alter table jobs

drop salary;


-- 29. Добавить поля min_salary тип numeric(2, 1), max_salary целое число

alter table jobs

add min_salary numeric(2, 1),

add max_salary integer;


-- 30. Изменить тип min_salary на целое число

alter table jobs
```

```
modify column min_salary integer;

-- 31. Удалить таблицу
drop table jobs;

-- Пример:
-- Создать таблицу phones
create table phones(
id integer primary key auto_increment,
product_name varchar(128) not null,
product_count integer,
starting_price integer not null);

-- Заполнить таблицу
insert into Phones(Product_Name, Product_Count, Starting_Price)
                        values('iPhone X', 3, 680),
        ('iPhone XR', 2, 700),
        ('iPhone XS', 10, 720),
        ('iPhone 11', 1, 790),
        ('iPhone 11 Pro', 12, 850),
        ('iPhone 11 Pro Max', 2, 890),
        ('iPhone SE', 1, 850),
        ('iPhone 12 Mini', 15, 890),
        ('iPhone 12', 20, 910),
        ('iPhone 12 Pro', 5, 950),
        ('iPhone 12 Pro Max', 13, 1000);
```

```
-- Оператор AS

--  С помощью оператора AS можно изменить название выходного столбца или определить его псевдоним:


select product_name as title

from phones;


-- Вывести поле product_name как title, посчитать и вывести общую сумму заказов (product_count * starting_price) как total_sum

select product_name as title, product_count * starting_price as total_sum

from phones;


-- Вывести поля product_name как name, product_count как count и starting_price как price.

select product_name as name, product_count as count, starting_price as price

from phones;



-- CASE оператор

-- В MySQL оператор CASE имеет функциональность оператора IF-THEN-ELSE


-- CASE

--    WHEN условие_1 THEN результат_1

--    WHEN условие_2 THEN результат_2

--    .....

--    WHEN условие_N THEN результат_N

--    [ELSE альтернативный_результат]

-- END


select product_name, product_count,

case

        when product_count <= 2 then 'Товар заканчивается'
```

```
    when product_count <= 5 then 'Мало товара'
    when product_count <= 10 then 'Есть в наличии'
    else 'Много товара'
end as category
from phones;


-- Создать новое поле category, заполнить ее:
-- Если Product_Count меньше/равно 2, category = 'Товар заканчивается',
-- Если Product_Count меньше/равно 5, category = 'Мало товара',
-- Если Product_Count меньше/равно 10, category = 'Есть в наличии',
-- В остальных случаях category ='Много товара';

alter table phones
add category varchar(128);

set sql_safe_updates = 0;

update phones
set category =
case
        when product_count <= 2 then 'Товар заканчивается'
        when product_count <= 5 then 'Мало товара'
    when product_count <= 10 then 'Есть в наличии'
    else 'Много товара'
end;

-- Создать новое поле tax целое число
-- Заполнить поле tax
-- Если начальная цена меньше 700, tax = null,
```

```
-- если больше либо равно 700 и меньше 850, tax = 15,
-- если больше или равно 850, tax = 25

alter table phones
add tax integer;

update phones
set tax =
case
        when starting_price < 700 then null
   when starting_price < 850 then 15
   else 25
end;


 -- Посчитать конечную цену и представить его как Final_price
 select product_name, starting_price, tax, starting_price + starting_price * tax / 100 as final_price
 from phones;

 -- Создать новое поле для final_price numeric(7, 2)
 -- Заполнить поле final_price (начанльная цена + процент)

alter table phones
add final_price numeric(7, 2);

 update phones
 set final_price =
 case
        when tax is null then starting_price
   else starting_price + starting_price * tax / 100
```

```
 end;


 -- Функция COALESCE принимает список значений и возвращает первое из них, которое не равно NULL:

update phones

set final_price = coalesce(starting_price + starting_price * tax / 100, starting_price);


select * from phones;


-- Функции для работы с числами

select -1 as value;


-- ABS: возвращает абсолютное значение числа.

select abs(-23) as abs;


-- POW(m, n): возвращает m, возведенную в степень n.

select pow(4, 2) as pow;


-- ROUND: округляет число. В качестве первого параметра передается число. Второй параметр указывает на длину.

-- Если длина представляет положительное число, то оно указывает, до какой цифры после запятой идет округление.

-- Если длина представляет отрицательное число, то оно указывает, до какой цифры с конца числа до запятой идет округление


select round(123.567, 2) as round_1;

select round(123.567, -2) as round_2;


-- SQRT: получает квадратный корень числа.

select sqrt(225) as sqrt;
```

```
-- RAND: генерирует случайное число с плавающей точкой в диапазоне от 0 до 1.

select rand() as rand;


-- https://metanit.com/sql/mysql/6.2.php



-- Функции для работы со строками

select 'Tom Smith' as full_name;


-- CONCAT: объединяет строки. В качестве параметра принимает от 2-х и более строк, которые надо соединить:

select concat('Tom', ' ' , 'Smith') as full_name;


-- CONCAT_WS: также объединяет строки, но в качестве первого параметра принимает разделитель, который будет соединять строки:

select concat_ws(' ', 'Tom', 'Smith', 'age', '34');


-- LENGTH: возвращает количество символов в строке. В качестве параметра в функцию передается строка, для которой надо найти длину:

select length('Tom Smith');

select length('Том');


-- LTRIM: удаляет начальные пробелы из строки. В качестве параметра принимает строку:

select ltrim('    apple');


-- RTRIM: удаляет конечные пробелы из строки. В качестве параметра принимает строку:

select rtrim('apple      ');


-- TRIM: удаляет начальные и конечные пробелы из строки. В качестве параметра принимает строку:

select trim('    apple   ');
```

-- LEFT: вырезает с начала строки определенное количество символов.

-- Первый параметр функции - строка, а второй - количество символов, которые надо вырезать сначала строки:

```
select left('apple', 3);
```

-- RIGHT: вырезает с конца строки определенное количество символов.

-- Первый параметр функции - строка, а второй - количество символов, которые надо вырезать сконца строки:

```
select right('apple', 3);
```

-- SUBSTRING(str, start [, length]): вырезает из строки str подстроку, начиная с позиции start.

-- Третий необязательный параметр передает количество вырезаемых символов:

```
select substring('galaxy S8 plus', 8);

select substring('galaxy S8 plus', 8, 2);
```

-- REPLACE(search, find, replace): заменяет в строке search подстроку find на подстроку replace.

-- Первый параметр функции - строка, второй - подстрока, которую надо заменить, а третий - подстрока, на которую надо заменить:

```
select replace('galaxy S8 plus', 'S8', 'A9');
```

-- LOCATE(find, search [, start]): возвращает позицию первого вхождения подстроки find в строку search.

-- Дополнительный параметр start позволяет установить позицию в строке search, с которой начинается поиск подстроки find.

-- Если подстрока search не найдена, то возвращается 0:

```
select locate('m','Tom Smith');

select locate('m','Tom Smith', 4);

select locate('a','Tom Smith');
```

-- REVERSE: переворачивает строку наоборот:

```
select reverse('12345');
```

```
-- LOWER: переводит строку в нижний регистр:
select lower('APPLE');

-- UPPER: переводит строку в верхний регистр
select upper('ApPlE');

-- https://metanit.com/sql/mysql/6.1.php

-- Пример:
create table user_bremen(
id integer primary key auto_increment,
first_name varchar(128),
last_name varchar(128),
age integer);

create table user_berlin(
id integer primary key auto_increment,
first_name varchar(128),
last_name varchar(128),
age integer,
phone varchar(128));

insert into user_bremen(first_name, last_name, age)
values('Linda', 'Johnson', 35),
            ('Barbara', 'Jones', 21),
   ('Laura', 'Thomas', 40),
   ('Sarah', 'Garcia', 30),
   ('Kimberly', 'Wans', 43);

insert into user_berlin(first_name, last_name, age, phone)
```

```
values('Carol', 'Robinson', 32, '+8129808937'),
        ('Barbara', 'Jones', 21, '+8129823454'),
   ('Melissa', 'King', 20, '+8129380234'),
   ('Sarah', 'Garcia', 30, '+812980223232'),
   ('Amy', 'Wans', 43, '+812932323');

select * from user_bremen;
select * from user_berlin;

-- UNION / UNION ALL - "вертикальное" объединение
-- Оператор UNION ALL используется для объединения наборов результатов из 2 или более предложений SELECT.
-- Он возвращает все строки из запроса и не удаляет повторяющиеся строки между различными предложениями SELECT.
-- Оператор UNION позволяет обединить две однотипных выборки. Эти выборки могут быть из разных таблиц или из одной и той же таблицы.

select id, first_name, last_name, age
from user_bremen
union all
select id, first_name, last_name, age
from user_berlin;

select id, first_name, last_name, age
from user_bremen
union
select id, first_name, last_name, age
from user_berlin;

-- Оператор ORDER BY сортируют значения по одному или нескольких столбцам.
select id, first_name, last_name, age
```

```
from user_bremen
union
select id, first_name, last_name, age
from user_berlin
order by first_name;


 -- По умолчанию данные сортируются по возрастанию, однако с помощью оператора DESC можно задать сортировку по убыванию.
 select id, first_name, last_name, age
from user_bremen
union
select id, first_name, last_name, age
from user_berlin
order by first_name desc;


--  https://metanit.com/sql/mysql/4.3.php
```

```
-- Практика
select * from employees;

-- 1.	Используем таблицу employees.
-- Нужно повысить зарплаты для отдела sales на 20%, для finance 15%, для shipping 10%, для marketing 25%, для Human Resources 20% и для IT 35%.
-- Записать данные в поле new_salary. Поле создавать не нужно, используем AS.

Select first_name, last_name, salary, department,
case
	when department = 'Sales' or department = 'Human Resources' then salary + salary * 20 / 100
	when department = 'Finance' then salary + salary * 15 / 100
	when department = 'shipping' then salary + salary * 10 / 100
	when department = 'marketing' then salary + salary * 25/ 100
	when department = 'IT' then salary + salary * 35 / 100
end as new_salary
from employees;

-- 2.	Создать поле new_salary numeric(7, 2), заполнить.
alter table employees
add new_salary numeric(7, 2);
update employees
set new_salary =
 case
	when department = 'Sales' or department = 'Human Resources' then salary + salary * 20 / 100
	when department = 'Finance' then salary + salary * 15 / 100
	when department = 'shipping' then salary + salary * 10 / 100
	when department = 'marketing' then salary + salary * 25/ 100
	when department = 'IT' then salary + salary * 35 / 100
end;
```

```
-- 3.	Используем таблицу customers.

-- Создать поле sale integer.

alter table customers

add sale integer;


-- 4.	Заполнить поле sale:

-- Если sum_price < 5000 скидка null, если больше/равно 5000 и меньше 10000, скидка 5, если больше/равно 10000, скидка 10.

update customers

set sale =

case

	when sum_price < 5000 then null

   when sum_price < 10000 then 5

   else 10

end;


-- 5.	Создать поле final_price numeric(7, 2).

-- Заполнить поле соответственно сделав скидки.

alter table customers

add final_price numeric(7, 2);


update customers

set final_price = coalesce(sum_price - sum_price * sale / 100, sum_price);


-- 6.	В order_name поменять math_book на mathematics_book, eng_book на english_book.

update customers

set order_name =  replace(order_name, 'math_book', 'mathematics_book'),

	order_name = replace(order_name, 'eng_book', 'english_book');
```

-- Д/З

-- 7. Вывести случайные числа от 1 до 10.

-- 8. Используем таблицу employees.

-- Имя и фамилию соединить в новое поле full_name.

-- 9. Поле email перевести в нижний регистр.

-- 10. Используем бд sakila

-- 11. Используем таблицу film

-- В description слово beautiful заменить словом amazing.

-- 12. Вывести поле title_len (количество символов поле title).

-- 13. Объединить таблицы customer и staff, вывести поля first_name, last_name, email, address_id и active.

-- 14. Используем бд fe37_38

-- Объединить таблицы customers и employees, вывести поля full_name, email (если нет, то null).

-- 15. Добавить сортировку по email.