

01001  
00101  
01100



# РАЗРАБОТКА НА C++

## УРОК 6. ГРУППИРОВКА ДАННЫХ И ОБЪЕДИНЕНИЕ ТАБЛИЦ



01001  
00101  
01100



## ГРУППИРОВКА ДАННЫХ И ОБЪЕДИНЕНИЕ ТАБЛИЦ

### DISTINCT

**DISTINCT** – это команда, с помощью которой можно выбрать уникальные значения таблицы

#### Пример

```
SELECT DISTINCT имя_столбца FROM имя_таблицы
```

### GROUP BY

Группировка данных в SQL проводится с помощью оператора **GROUP BY**. Эта команда группирует данные при выборке, имеющие одинаковые значения в столбце

#### Пример

```
SELECT * FROM имя_таблицы  
GROUP BY имя_столбца;
```

Т.е. на выходе мы получаем таблицу, в которой есть только уникальные значения в столбце `имя_столбца`

#### Пример использования оператора **GROUP BY**:

Но когда мы используем оператор **GROUP BY**, у нас все строки, у которых в заданном столбце совпадает значение, должны собраться в одну. Следовательно, если в остальных столбцах значения различаются, то нам нужно их тоже как-то объединить. Мы можем их добавить в оператор **GROUP BY**:

```
SELECT * FROM имя_таблицы  
GROUP BY имя_столбца1, имя_столбца2;
```

Групповая функция **SUM()** суммирует все значения в столбце. При работе с оператором **GROUP BY** она суммирует все значения в столбце в рамках группы

Групповая функция **COUNT()** считает, сколько записей относится к группе

```
SELECT столбец1, SUM(столбец2), COUNT(столбец3) FROM таблица  
GROUP BY столбец1;
```

Групповая функция **MIN()** вычисляет минимальное значение элементов столбца, относящихся к группе

Групповая функция **MAX()** вычисляет максимальное значение элементов столбца, относящихся к группе

Групповая функция **AVG()** вычисляет среднее значение элементов столбца, относящихся к группе

```
SELECT столбец1, MIN(столбец2), MAX(столбец3), AVG(столбец4) FROM таблица  
GROUP BY столбец1
```



Кроме того, мы уже прошли с вами условие отбора строк с помощью оператора WHERE. В запросах с оператором GROUP BY вместо WHERE используется ключевое слово HAVING

```
SELECT столбец1, MIN(столбец2) FROM таблица  
GROUP BY столбец1  
HAVING условие;
```

## UNION

Объединение двух или более таблиц – это объединение данных из разных таблиц в одну

**Важно.** При объединении количество столбцов во всех таблицах должно совпадать, иначе будет ошибка. Имена столбцов будут такие же, как в основной таблице, в которую добавляются данные из других таблиц.

Для объединения данных нужно воспользоваться командой UNION:

```
SELECT * FROM имя_таблицы1 WHERE условие  
UNION SELECT * FROM имя_таблицы2 WHERE условие
```

Так у нас выведется таблица, в которой все строки будут уникальными (т.е. ни одна строка не будет полностью совпадать с другой), при этом мы получим выборку из двух таблиц

Если мы хотим получить все строки, в т.ч. и повторяющиеся, то нужно использовать служебное слово ALL:

```
SELECT * FROM имя_таблицы1 WHERE условие  
UNION ALL SELECT * FROM имя_таблицы2 WHERE условие
```

**Важно.** Имена колонок берутся из первой таблицы

## JOIN

Соединение таблиц – это операция, когда таблицы сравниваются между собой построчно и появляется возможность вывода столбцов из всех таблиц, участвующих в соединении

Есть несколько видов оператора JOIN

**INNER JOIN** – внутреннее соединение. Выводит строки, только если условие соединения выполняется. Условие прописывается после ключевого слова ON. В запросах необязательно прописывать INNER – если написать только JOIN, то СУБД по умолчанию выполнит именно внутреннее соединение

```
SELECT * FROM таблица1  
JOIN таблица2  
ON условие;
```



**LEFT JOIN** и **RIGHT JOIN** называются левым и правым соединениями, или внешними. При таких соединениях из левой (для **LEFT JOIN**) или из правой таблицы (для **RIGHT JOIN**) попадает в результаты в любом случае

Левая таблица – та, название которой указывается перед написанием ключевых слов **[LEFT | RIGHT | INNER] JOIN**, правая таблица – та, название которой указываем после них

```
SELECT * FROM  
левая_таблица LEFT JOIN правая_таблица  
ON условие;
```

**FULL JOIN** – полное внешнее соединение. Возвращает все строки из всех таблиц, участвующих в соединении, соединив между собой те, которые подошли под условие **ON**.

```
SELECT * FROM таблица1  
FULL JOIN таблица2  
ON условие;
```



01001  
00101  
01100