

# Простая выборка данных с помощью языка SQL

# Простая выборка данных

```
SELECT [ALL | DISTINCT] [TOP n [PERCENT]
```

*списокВыборки*

```
FROM ИмяТаблицы
```

```
WHERE УсловиеОтбора
```

*СписокВыборки определяет поля, включаемые в итоговый набор данных,*

*ИмяТаблицы указывает таблицу БД, из которой возвращаются записи,*

*УсловиеОтбора позволяет ограничить число возвращаемых записей с помощью логических операторов.*

# Ключевые слова

- ▶ *DISTINCT* – возвращает уникальные записи
- ▶ *ALL* - возвращает *все записи, включая дубликаты,*
- ▶ *TOP n* – возвращает *n первых записей*
- ▶ *Percent* – возвращает *определенный процент от всех строк*

# Список выборки

- ▶ Список выборки может содержать включать следующие один или несколько элементов:

**\* | ИмяПоля | Выражение [AS Псевдоним], [...n].**

Для выборки всех полей из таблицы в списке выборки необходимо указать звездочку (\*).

Ключевое слово *AS* позволяет заменить в итоговом наборе данных обычные имена полей псевдонимами

# Пример

```
SELECT Studentid AS 'Код',[Name] AS 'Фамилия',  
[BirthDate] AS 'Дата рождения'  
FROM Students
```

# Выражение в запросе

*Выражение* задает выражение, которое включается в итоговый набор данных. Выражение может содержать константы, имена полей, функции и их комбинации. По умолчанию имя колонки с выражением не определено, поэтому можно указать псевдоним.

Например, список студентов с указанием фамилии и первого символа имени и идентификационного номера может быть получен в результате запроса:

```
SELECT [Фамилия]+' '+Substring([Имя],1,1)+'.' AS  
[Студент], StudentID  
FROM Students
```

# Сортировка

Сортировка возможна по имени поля (даже если оно и не указано в списке выборки), по псевдониму или по позиции в списке выборки, которые указываются в разделе `ORDER BY` `ИмяПоля` `[,...n]` `[ASC | DESC]`.

По умолчанию сортировка осуществляется по возрастанию, что соответствует зарезервированному слову *ASC*, которое может опускаться, для сортировки в убывающем порядке указывается – *DESC*

```
SELECT [Фамилия]+'`'+Substring([Имя],1,1)+'.' AS  
[Студент], StudentID  
FROM Students  
ORDER BY [Сотрудник]
```

# Условие отбора

Условие отбора определяет критерий отбора записей, включаемых в итоговый набор. В результат будут включены только те строки, которые соответствуют наложенным условиям.

Условие может включать выражения, образованные с помощью операторов сравнения или логических операторов и с помощью логических операндов AND, OR и NOT.

```
SELECT [Фамилия]+' '+Substring([Имя],1,1)+'.' AS  
[Студент], StudentID  
FROM Students  
WHERE studentID >=2 and studentID<43  
ORDER BY [Сотрудник]
```

# Оператор BETWEEN

С помощью оператора BETWEEN можно получить ответ на вопрос, лежит ли величина в указанном диапазоне.

Данный оператор предназначен лишь для того, чтобы облегчить логику восприятия алгоритма.

```
SELECT [Фамилия]+' '+Substring([Имя],1,1)+'.' AS  
[Студент], StudentID  
FROM Students  
WHERE studentID Between 2 and 43  
ORDER BY [Сотрудник]
```

# Оператор LIKE

Для поиска по шаблону символьных строк используется логический оператор LIKE, который чаще всего используется в ситуациях, когда неизвестно точное совпадение

В шаблоне могут использоваться следующие символы:

- ▶ \* (%) – подразумевает любую строку, состоящую из 0 и более символов;
- ▶ ? (\_) – ровно один символ;
- ▶ [ ] – любой символ из заданного множества (например, [adfh]) или диапазона (например, [0-9]),
- ▶ [!a-p] – любой символ, не попадающий в заданный диапазон или множество.
- ▶ #- одна цифра

# Пример LIKE

```
SELECT [Фамилия]+'`'+Substring([Имя],1,1)+'.' AS  
[Студент], StudentID  
FROM Students  
WHERE [Фамилия ] like "*ОВ"
```

# Оператор IN

Для определения соответствия выражения одному из перечисленных в заданном списке значений применяется логический оператор IN. Данный оператор всегда может быть записан и в виде группы условий, объединенных операндом OR

```
SELECT [Фамилия]+'`'+Substring([Имя],1,1)+'.' AS  
[Студент], StudentID  
FROM Students  
WHERE [Фамилия ] IN ('Иванов','Петров')
```

# Оператор NULL

Однако в список значений нельзя включать неопределенное значение NULL, для работы с такими значениями используется функция выборки IS NULL.

Например, следующий запрос возвращает студентов, у которых не указан год рождения

```
SELECT [Фамилия]+' '+Substring([Имя],1,1)+'.' AS  
[Студент], StudentID  
FROM Students  
WHERE [birthDate] is NULL
```

# Выборка данных из нескольких таблиц

Такая выборка данных предполагает *соединение* нескольких таблиц для получения единого набора результатов, включающих записи и поля каждой таблицы. Соединение позволяет собрать данные, разделенные в процессе нормализации.

Существует три вида соединений: внутреннее, внешнее и перекрестное.

# Аналитическая выборка данных

Аналитическая выборка данных из базы данных неразрывно связанных с агрегатными функциями:

- ▶ *Avg* (*[all | distinct] выражение*) – среднее арифметическое всех значений.
- ▶ *Count* (*[all | distinct] выражение | \**) – количество значений в списке, отличных от NULL. При использовании символа \* подсчитывается количество значений, включая значения NULL или повторяющиеся значения.
- ▶ *Sum* (*[all | distinct] выражение*) – сумма всех значений списка.
- ▶ *Max* (*[all | distinct] выражение*) – максимальное значение.
- ▶ *Min* (*[all | distinct] выражение*) – минимальное значение.

Ключевое слово **all** предписывает выполнять агрегирование всех записей в результирующем наборе данных, **distinct** – агрегирование только уникальных записей. По умолчанию используется **all**.

# Пример запроса с агрегатными функциями

Например, вычисление средней цены товаров осуществляется с помощью следующего запроса:

```
SELECT AVG([Отметка]) FROM [Студент_Предмет]
```

При выполнении агрегатной функции осуществляется объединение значений отдельного поля таблицы или части записей, после чего выполняется указанное агрегирование.

Агрегатная функция возвращает одно единственное значение, поэтому использование других имен полей в списке выборки запрещено.

# Группировка записей

Для группировки записей по полям или выражениям применяется раздел *GROUP BY* оператора *SELECT*, что позволяет применять для каждой группы функции агрегирования.

Синтаксис данной части следующий:

[*GROUP BY* ВыражениеГруппировки, [...n]]

Определяем количество студентов в каждом городе

```
SELECT Count([Фамилия])
```

```
FROM Students
```

```
WHERE [birthDate] is NULL
```

```
GROUP BY [CITY]
```

При использовании *GROUP BY* для каждой определенной группы значений выводится только одна запись в итоговом наборе данных.

# Ключевое слово Having

раздел *HAVING* – какие группы должны быть выведены в итоговый набор данных. Ключевое слово *HAVING* можно использовать только в разделе *GROUP BY*

```
SELECT Count([Фамилия])  
FROM Students  
WHERE [birthDate] is NULL  
GROUP BY [CITY]  
HAVING Count([Фамилия])>10
```