

# Оператор выборки Select

## Лекция №4

Бутенко И.В. 2017 год

# Введение

- **SQL = DDL**(Data definition Lang) + **DML**  
(Data Manipulation Lang)
- **DDL: CREATE, ALTER, DROP**
- **DML: SELECT, INSERT, UPDATE, DELETE**

# Таблицы

Таблица – специальный тип данных, который может использоваться для сохранения данных для дальнейшей обработки.

Ограничения запрещают вносить в таблицу недопустимые данные.

Ключом называется множество атрибутов, задание значений которых позволяет однозначно определить значения остальных атрибутов.

# Пример

```
CREATE TABLE students
```

```
(  
    id    int identity(1,1) PRIMARY KEY,  
    name  varchar(30) not null,  
    lastname varchar(30) not null,  
    birthday datetime null  
)
```

```
CREATE TABLE subjects
```

```
(  
    id    int identity(1,1),  
    name  varchar(30) not null,  
    hours smallint null  
)
```

```
CREATE TABLE marks
```

```
(  
    stud_id int FOREIGN KEY REFERENCES students (id),  
    subj_id int,  
    ddate  datetime default getdate(),  
    mark   tinyint CHECK (mark > 1 and mark <= 5)  
)
```

# Определение

- **Оператор выборки SELECT** извлекает информацию из базы данных и возвращает ее в виде таблицы результатов запроса (производит выборку строк и столбцов из таблиц).

# Общий вид SELECT

- **SELECT** [ALL | DISTINCT] <select\_list>
- [INTO [new\_table\_name]]
- [**FROM** {<table\_source>}  
[...,<table\_source>]
- [**WHERE** <search condition>]
- [**GROUP BY** <group list>]
- [**HAVING** < search condition>]
- [**ORDER BY** <sort list>]

# Возможности SELECT 1

- `select * from students`
- `select name, lastname, birthday from students`
- `select lastname as 'Фамилия', birthday 'Дата рождения' from students`
- `select 'Студент: ' + name + ' ' + lastname from students`

# Возможности SELECT 2

- `select lastname as 'Фамилия',  
isnull(convert(varchar,birthday,103),'???'  
) 'Дата рождения' from students`
- `select avg(mark) from marks (abs, sign,  
sqrt, round)`
- `select ddate, getdate() curdate,  
datediff(d, ddate, getdate()) diff from  
marks`



# Возможности SELECT 3

- `select * from marks where mark < 3`
- `select * from marks where ddate between '30/04/2010' and '01/06/2010'`
- `select * from students where lastname like 'П%' or name like 'M_____'`

# ВОЗМОЖНОСТИ SELECT 4

- `select * from students where birthday is null`
- `select * from students order by lastname asc, birthday desc`
- `select lastname + ' ' + left(name,1) name from students where birthday is null`  
Union `select name from subjects where hours is null`

# Расширенные возможности 1

- **GROUP BY** организует группы данных
  - группировка выполняется по столбцу
  - используются с функциями группировки
  - образует одно значение для группы
- **HAVING** фильтрует группы по условию
  - можно использовать по столбцу или выражению
  - то же самое, что и блок WHERE
- **COMPUTE**
  - Образует общие итоговые значения.

# Расширенные возможности 2

- `select stud_id,  
avg(convert(decimal(5,2),mark)) from  
marks group by stud_id`
- `select stud_id,  
avg(convert(decimal(5,2),mark)) from  
marks group by stud_id having  
avg(convert(decimal(5,2),mark)) >= 4`
- `select * from marks where ddate >  
'01/10/2010' compute max(mark)`

# Выборка с подзапросом 1

- Подзапрос в блоке WHERE ссылается на внешнюю таблицу
- Подзапрос выполняется однократно для каждой строки внешнего запроса
- Если в подзапросе условие выполняется, то внешний запрос выдает строку

# Выборка с подзапросом 2

- `select * from students s where exists (select * from marks m where m.stud_id = s.id)`
- `select * from marks main where ddate = (select max(ddate) from marks sub where main.stud_id = sub.stud_id)`

# Многотабличная выборка 1

- `select m.mark, s.lastname from marks m  
join students s on m.stud_id = s.id`
- Старый синтаксис:  
`select m.mark, s.lastname from marks m,  
students s where m.stud_id = s.id`

# CROSS JOIN 1

- Декартово произведение двух таблиц представляет собой таблицу (называемую *таблицей произведения*), состоящую из всех возможных пар строк обеих таблиц. Столбцами таблицы произведения являются все столбцы первой таблицы, за которыми следуют все столбцы второй таблицы.



# CROSS JOIN 2

=

```
select authors.au_id, au_name,  
       b_id, b_name  
from authors cross join books
```

```
select authors.au_id, au_name,  
       b_id, b_name  
from authors, books
```

au_id	au_name	b_id	b_name
1	Иванов	1	Сказка
2	Петров	1	Сказка
3	Сидоров	1	Сказка
1	Иванов	2	Детектив
2	Петров	2	Детектив
3	Сидоров	2	Детектив
1	Иванов	3	Учебник
2	Петров	3	Учебник
3	Сидоров	3	Учебник



Books		
b_id	au_id	b_name
1	1	Сказка
2	1	Детектив
3	2	Учебник
4	4	ФЭНТЕЗИ

Authors	
au_id	au_name
1	Иванов
2	Петров
3	Сидоров

# INNER JOIN 1

- При этом типе связывания каждая из 2х участвующих в связывании таблиц будет включать только те строки, для которых есть соответствие во второй таблице.

# INNER JOIN 2

```
select au_name, b_name  
from authors [inner] join books  
on authors.au_id = books.au_id
```

=

```
select au_name, b_name  
from authors, books  
where authors.au_id = books.au_id
```

au_name	b_name
Иванов	Сказка
Иванов	Детектив
Петров	Учебник

Books		
b_id	au_id	b_name
1	1	Сказка
2	1	Детектив
3	2	Учебник
4	4	Фэнтези

Authors	
au_id	au_name
1	Иванов
2	Петров
3	Сидоров

# LEFT OUTER JOIN 1

- При этом типе связывания в левой таблице будут оставлены все строки независимо от того, есть ли для них соответствие в правой таблице.

# LEFT OUTER JOIN 2

```
select au_name, b_name  
from authors left outer join books  
on authors.au_id = books.au_id
```

=

```
select au_name, b_name  
from authors, books  
where authors.au_id *= books.au_id
```

au_name	b_name
Иванов	Сказка
Иванов	Детектив
Петров	Учебник
Сидоров	NULL

Books		
b_id	au_id	b_name
1	1	Сказка
2	1	Детектив
3	2	Учебник
4	4	фэнтези

Authors	
au_id	au_name
1	Иванов
2	Петров
3	Сидоров

# RIGHT OUTER JOIN 1

- При этом типе связывания в правой таблице будут оставлены все строки независимо от того, есть ли для них соответствие в левой таблице.

# RIGHT OUTER JOIN 2

```
select au_name, b_name  
from authors right outer join books  
on authors.au_id = books.au_id
```

=

```
select au_name, b_name  
from authors, books  
where authors.au_id =* books.au_id
```

au_name	b_name
Иванов	Сказка
Иванов	Детектив
Петров	Учебник
NULL	Фэнтези

Books		
b_id	au_id	b_name
1	1	Сказка
2	1	Детектив
3	2	Учебник
4	4	Фэнтези

Authors	
au_id	au_name
1	Иванов
2	Петров
3	Сидоров

# FULL OUTER JOIN 1

- Этот тип связывания разрешает использование всех строк связываемых таблиц. Можно представить этот тип связывания как одновременное применение типов LEFT и RIGHT (что не разрешено).



# FULL OUTER JOIN 2

```
select au_name, b_name  
from authors full outer join books  
on authors.au_id = books.au_id
```

Books		
b_id	au_id	b_name
1	1	Сказка
2	1	Детектив
3	2	Учебник
4	4	Фэнтези

Authors	
au_id	au_name
1	Иванов
2	Петров
3	Сидоров

au_name	b_name
Иванов	Сказка
Иванов	Детектив
Петров	Учебник
NULL	Фэнтези
Сидоров	NULL

# Создание таблиц

- `select * into #stud from students where birthday > '01/01/1985'`
- `select * from #stud`