SQL - язык проектирования РБД

Стандарты SQL

- ANSI – Американский национальный институт стандартов, ISO – Международная организация стандартов
- Стандарт SQL1 был впервые опубликован в 1986 г. - обеспечивал минимальную функциональность, обновлялся в 1989 – механизм поддержания ссылочной целостности
- в 1992 (SQL2) - расширенная функциональность
- в 1999 (SQL3) – интеграция с объектно-ориентированным подходом

SQL-серверы

СУБД	Производитель	URL
Oracle	Oracle Corp.	www.oracle.com
MS SQL	Server- Microsoft	www.microsoft.com
Informix	Informix	www.informix.com
Sybase	Sybase	www.sybase.com
DB2	IBM	www.4.ibm.com

Словарь SQL

Два типа запросов:

- Возвращающий строки: **SELECT**

SELECT список полей или *

FROM список таблиц

WHERE условие отбора

GROUP BY выражение_группирования

HAVING условие_включения_группы

ORDER BY столбец | выражение **[ASC | DESC],... ;**

- Не возвращающие строки: **Action Queries**
  - Update : изменение записей
  - Insert : вставка новой записи
  - Delete : удаление записи

Рассмотрим БД, которая моделирует сдачу сессии в некотором учебном заведении, Пусть она состоит из трех отношении

R1 =(ФИО, Дисциплина, Оценка)

R2 = (ФИО, Группа);

R3 = (Группы, Дисциплина)

R1

ФИО	Дисциплина	Оценка
Петров Ф. И.	Базы данных	5
Сидоров К. А.	Базы данных	4
Миронов А. В.	Базы данных	2
Степанова К. Е.	Базы данных	2
Крылова Т. С.	Базы данных	5
Сидоров К. А.	Теория информации	4
Степанова К. Е.	Теория информации	2
Крылова Т. С.	Теория информации	5

R2

ФИО	Группа
Петров Ф. И.	4906
Сидоров К. А.	4906
Миронов А. В.	4906
Крылова Т. С.	4906
Владимиров В. А.	4906
Трофимов П. А.	4807
Иванова Е. А.	4807
Уткина Н. В.	4807

R3

Группа	Дисциплина
4906	Базы данных
4906	Теория информации
4906	Английский язык
4807	Английский язык
4807	Сети и телекоммуникации

Посчитать количество двоек за экзамен «БД»

SELECT “количество двоек =” **count**(*)

FROM R1

WHERE Дисциплина =“БД” **AND**

“Оценка” = 2

Результат

Количество двоек = 3

Например, можно вычислить количество студентов, сдававших экзамены по каждой дисциплине. Для этого надо выполнить запрос с группировкой по полю «Дисциплина» и вывести в качестве результата название дисциплины и количество строк в группе по данной дисциплине. Применение символа * в качестве аргумента функции COUNT означает подсчет всех строк в группе.

```
SELECT R1.Дисциплина, COUNT(*)
```


Дисциплина	COUNT(*)
Базы данных	6
Теория информации	4
Сети и телекоммуникации	3
Английский язык	4

Типы данных

- Для указания даты используется знак **#** (в стандарте ANSI – апостроф, т.е. **'2/17/94 13:00'**: **#5/2/62#** **#4:12 am#**
- Значение NULL обозначает отсутствие данных в поле. NULL это не 0 и не пустая строка. Сравнение выполняется с помощью оператора IS NULL.

Примеры:

1. Все строки таблицы Authors

```
SELECT * FROM Authors
```

2. Все столбцы и те строки, для которых в столбце PubID = 1213

```
SELECT * FROM Publishers WHERE PubID = 1213
```

3. Два столбца и те строки, для которых верно условие

```
SELECT LastName, PlaceofBirth FROM Customers
   WHERE ((AGE > 30) and (SEX = 'M'))
      ORDER BY LastName, PlaceOfBirth
```

Для выборки данных по шаблону можно использовать оператор **LIKE** с заменителями - **%** или *****.

Примеры оператора LIKE

- (MS **Access использует для указания любого символа знак *, ANSI SQL - %**):

...Where ((LastName Like ‘SM*’) or (Name Like ‘sm*’) or (Name Like ‘Sm*’))

- Оператор LIKE выполняется быстрее, если указан в конце оператора WHERE.
- ANSI SQL использует круглые скобки ( ). MS Access использует также [ ], поэтому желательно для преемственности кода заменить скобки на круглые.
- Для указания в запросе источника данных используется символ точка(.)

Database.Table.Field

- Оператор IN используется в операторе WHERE для указания подмножества, к которому может относиться проверяемое поле записи.
- Подмножеством может быть список или результат выполнения запроса (в этом случае подзапрос должен возвратить список значений одного поля)

SELECT Name, YearBorn

FROM Authors **Where** YearBorn IN (1962, 1963, 1964)

SELECT Name, YearBorn

FROM Authors **Where** YearBorn **IN (SELECT** Year **FROM** HoleInOne)

- Asterisk (*)
  - **SELECT** authorID, firstName, lastName **FROM** Authors **WHERE** lastName **LIKE** 'D*'
- Question mark (?)
  - **SELECT** authorID, firstName, lastName **FROM** Authors WHERE lastName **LIKE** '?I*'
  - **DELETE FROM** Authors **WHERE** firstName **Like** 'Chan%' (ANSI SQL)
  - **DELETE FROM** Authors **WHERE** firstName **Like** 'Chan*' (MS Access)

Книги, авторы и издательства. Таблицы ИЗДАТЕЛЬСТВА и КНИГИ связаны по полю pubID. Таблицы АВТОРЫ и СВЕДЕНИЯ ОБ АВТОРАХ связаны по полю authorID. Таблицы КНИГИ и СВЕДЕНИЯ ОБ АВТОРАХ связаны по полям Titles.isbn и AuthorISBN.isbn

```
SELECT Titles.title, Authors.Name, Publishers.publisherName
  FROM (Publishers INNER JOIN Titles ON Publishers.pubID = Titles.pubID)
    INNER JOIN
      (Authors INNER JOIN AuthorISBN ON Authors.authorID =
                                          AuthorISBN.authorID)
    ON Titles.isbn = AuthorISBN.isbn
      ORDER BY Titles.title;
```

Оптимизация команды SELECT

- Не указывайте лишние столбцы в запросе
- Используйте не перечисление полей, а символ * (все поля).

Команда DELETE

DELETE * FROM таблица WHERE условие

Примеры:

Delete * FROM Authors Where Dead = TRUE
Delete * FROM Publishers Where PubID > 30
Delete * FROM MooCows

Команда UPDATE

UPDATE таблица SET поле = значение [, поле = значение ...] WHERE условие

Примеры:

Update Authors Set Commissions = (Sales * 0.1)
Update Authors Set Address = '123 Maple' Where (AuID = 3121)
Update Authors Set Dead=False, Stupid=True Where ((Sales>100000) and (Commissions=0))

Если команда содержит вычисления, то они будут выполняться на стороне сервера и это хорошо.

Команда INSERT

INSERT INTO таблица (поле, поле) VALUES (значение, значение)

Примеры:

INSERT INTO authors (Name, Address, Sales) VALUES (‘Smith, Frank’, ‘123 Main St’, 35232.06)

INSERT INTO publishers (Name, ABACODE, Paperbacks) VALUES (‘Smith Books’, 1311, TRUE)

ANSI SQL: True это не ноль, обычно –1, False это ноль

Связывание таблиц

Для выборки из связанных таблиц используется оператор JOIN.
Связи между таблицами бывают двух типов:

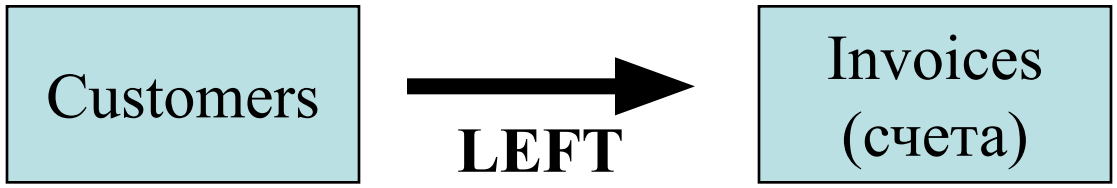
- внутренние **INNER**: запрос содержит совпадающие по ключевым полям строки.
- внешнее **OUTER**: запрос может включать пустые (NULL) поля.

Некоторые СУБД используют слово FULL

SELECT [поля]
FROM таблицаA {**INNER | LEFT | RIGHT**} **JOIN** таблицаB
ON (таблицаA.поле1 = таблицаB.поле2)
WHERE [условие]
ORDER BY [поля]

Выбор внешнего соединения – левое или правое?

- Внешнее соединение используется для
  - Выявления несовпадений в ключевых полях таблиц
  - Выявления пустых полей
- Левое соединение LEFT JOIN выбирает все записи левой таблицы и совпадающие по ключевому полю записи правой таблицы. Правое соединение – наоборот. Соединение **Full** JOIN выберет все записи в обеих таблицах, в том числе с совпадающими ключевыми полями



Все Покупатели с учетом и без учета Счетов

Все Счета с учетом и без учета Покупателей

Операторы GROUP BY и HAVING

- Используется для группировки записей
- Все поля, перечисляемые в части **SELECT**, должны упоминаться и в части **GROUP BY**, кроме тех полей, что участвуют в вычислениях в операторе SELECT.
- С помощью оператора **AS** можно дать имена вычисляемым полям, в которых могут использоваться агрегатные функции: **COUNT** (количество), **SUM** (сумма), **AVG** (среднее значение), **MIN, MAX**
- Оператор WHERE фильтрует записи.
- Оператор HAVING фильтрует результаты после их группировки, т.к. фильтрует группы.

```
SELECT Titles.PubID, Titles.[Year Published], Count(Titles.Title) AS Count
  FROM Titles
   GROUP BY Titles.PubID, Titles.[Year Published]
```

PubID	Year Published	Count
3	1994	31
3	1995	46
3	1996	27
4	1980	1
4	1986	1

НЕПРАВИЛЬНО:

```
SELECT dept_id, SUM(salary) FROM emp
    WHERE SUM(salary)>2500
      GROUP BY dept_id;
```

Результат:

```
WHERE SUM(salary)>2500
ERROR at line 3: ORA-00934: group function is not allowed here
```

Как фильтровать группы правильно:

1. Записи фильтровать по WHERE
2. Создать группы GROUP BY
3. Результат фильтровать по HAVING

ПРАВИЛЬНО:

```
SELECT dept_id, SUM(salary) FROM emp
  GROUP BY dept_id
    HAVING SUM(salary)>2500;
```

Результат:

```
DEPT_ID   SUM(SALARY)
---------      -----------
31     2800
41          4990
42          3245
```

```
SELECT dept_id, SUM(salary) FROM emp
  WHERE s_date=DATE('31121990','ddmmyyyy')
    GROUP BY dept_id;
```

Результат:

```
DEPT_ID   SUM(SALARY)
------------   -----------
31     2800
41     4990
```

Наиболее полно преимущества ключевого слова IN проявляются во вложенных запросах. Предположим, нам нужно найти все издания, выпущенные компанией "Oracle Press". Наименования издательских компаний содержатся в таблице **publishers**, названия книг в таблице **titles**. Ключевое слово NOT IN позволяет объединить обе таблицы и извлечь при этом нужную информацию:

```
SELECT title FROM titles WHERE pub_id IN
  (SELECT pub_id FROM publishers WHERE publisher='Oracle Press');
```

При выполнении этой команды СУБД вначале обрабатывает вложенный запрос по таблице **publishers**, а затем его результат передает на вход основного запроса по таблице **titles**.

Некоторые задачи нельзя решить с использованием только операторов сравнения. Например, мы хоти найти web-site издательства "Wiley", но не знаем его точного наименования. Для решения этой задачи предназначено ключевое слово LIKE, его синтаксис имеет вид:

```
WHERE <имя_столбца> LIKE <образец> [ ESCAPE <ключевой_символ> ]
```

Образец заключается в кавычки и должен содержать шаблон подстроки для поиска:

- % (знак процента) - заменяет любое количество символов
- _ (подчеркивание) - заменяет одиночный символ.

Подзапросы

SELECT список_выбора
FROM таблица, …
WHERE выражение *оператор сравнения*
(**SELECT** список_выбора **FROM** таблица, …);

где *оператор сравнения* **=, <, >** (для одной записи в подзапросе), **IN, ANY, ALL** (для нескольких записей в подзапросе).

- Могут быть вложены в SELECT, FROM, WHERE, HAVING
- Полезны для сложной выборки данных, могут быть именованными и использоваться для поиска дубликатов записей:
- В подзапросе нельзя использовать оператор сортировки ORDER BY

```
SELECT DISTINCTROW Titles.Title, Titles.[Year Published], Titles.PubID
    FROM Titles
      WHERE (Titles.Title IN
      (SELECT Title FROM Titles GROUP BY Title HAVING Count(*)>1 )
          ORDER BY Titles.Title
```

Книги с неоднократным упоминанием их названий в таблице TITLE.

ALL: Найти служащих, которые были приняты на работу раньше всех служащих в должности ‘Warehouse Manager’:

```
SELECT last_name FROM emp
  WHERE start_date <ALL
    (SELECT start_date  FROM emp WHERE title='Warehouse Manager');
```

LAST_NAME
Velasquez
Ngao
Ropeburn
Smith

FROM: Выбрать три региона, в которых больше всего фирм клиентов:

```
SELECT TOP 3 *
FROM (SELECT region_id, COUNT(*) FROM customer
   GROUP BY region_id ORDER BY 2 DESC) AS s;
```

```
REGION_ID COUNT(*) REGION
--------- -------- -------
1         4        1
5         4        2
4         3        3
```

HAVING: Найти должность с самой низкой средней заработной платой:

```
SELECT title, AVG(salary) FROM emp
   GROUP BY title
   HAVING AVG(salary)=
    (SELECT MIN(AVG(salary)) FROM emp GROUP BY title);

TITLE           AVG(SALARY)
------------    -----------
Stock Clerk     949
```

SELECT: Для каждого служащего получить его номер, номер отдела, в котором он работает, зарплату и процент, который составляет его зарплата в суммарной зарплате его отдела:

```
SELECT id, dept_id, salary,salary/
    (SELECT SUM(salary) FROM emp WHERE dept_id=e.dept_id)*100 "%"
    FROM emp e
        ORDER BY 2;
```

ID	DEPT_ID	SALARY	%
4	10	1450	100
3	31	1400	50
23	34	795	34,26
15	35	1450	100
2	41	1450	29,05

Виды вложенных запросов

- Однострочные
- Многострочные
- **Квантифицированные**
  - EXISTS, NOT EXISTS

Служащие, у которых зарплата такая же, как у служащего по фамилии Ngao

```
SELECT last_name FROM emp one
 WHERE EXISTS
 (SELECT * FROM emp
   WHERE salary=one.salary AND last_name='Ngao');
```

Названия регионов, в которых нет ни одного отдела

```
SELECT name FROM region
  WHERE NOT EXISTS
  (SELECT id  FROM dept
   WHERE dept.region_id=region.region.id);
```

Отдельно по должности и по году

```
SELECT id FROM emp
 WHERE title IN
 (SELECT title FROM emp WHERE dept_id=34)
   AND d_date IN
       (SELECT d_date FROM emp WHERE dept_id=34)
   AND dept_id<>34;
```

Результат: 11, 22, 17, 12, 16

Извлечение данных

```
Use NorthWind
SELECT ProductName AS [Название продукта]
from Products
```

	Название продукта
1	Alice Mutton
2	Aniseed Syrup

Указание на обращение к таблицам БД может быть указано явно командой **USE**. Полям таблицы можно задать псевдонимы, т.е. заголовки.

```
Use NorthWind
SELECT productname AS [название товара],
unitprice AS [Цена] FROM Products
WHERE unitprice BETWEEN 10 AND 12
```

	название товара	Цена
	Aniseed Syrup	10.0000
	Sir Rodney's Scones	10.0000
	Spegesild	12.0000
	Longlife Tofu	10.0000

Выбор товаров, цена которых лежит в пределах от 10 до 12.

```
Use NorthWind
SELECT productname AS [название товара],
unitprice AS [Цена] FROM Products
WHERE unitprice in (10,18)
```

Выбор товаров, цена которых или 10 или 18

	название товара	Цена
	Chai	18.0000
	Aniseed Syrup	10.0000
	Sir Rodney's Scones	10.0000
	Steeleye Stout	18.0000
	Chartreuse verte	18.0000
	Longlife Tofu	10.0000
	Lakkalikööri	

```
Use NorthWind
SELECT DISTINCT Unitprice from Products
```

	Unitprice
1	2.5000
2	4.5000

Для исключения повторов в **столбце** используется ключ **DISTINCT**

Функции

Для работы с датами используются функции извлечения года (**YEAR**), месяца (**MONTH**), дня (**DAY**)…

```
USE northwind
SELECT firstname,birthdate FROM employees
WHERE MONTH(birthdate)=7
```

firstname	birthdate
Michael	1963-07-02 00:00:00.000

Выбрать компании, у которых не указан регион

```
USE northwind
SELECT region, companyname FROM suppliers
WHERE region IS NOT NULL
```

region	companyname
LA	New Orleans Cajun Delights
MI	Grandma Kelly's Homestead

Сравнение со строкой - оператор **LIKE** со знаками **%** или **?** (т.е. любое количество символов. В MS Access знак “*”.)

```
USE northwind
SELECT productname FROM products
WHERE productname LIKE '%TOU%'
```

productname
Steeleye Stout
Tourtière

_ один любой символ;

[-…] один символ из диапазона:

```
USE northwind
SELECT productname FROM products
WHERE productname LIKE '%Ch_g%'
```

productname
Queso Manchego La Pastora
Schoggi Schokolade

```
USE northwind
SELECT productname FROM products
WHERE productname LIKE '%gu[ld]%'
```

productname
Gudbrandsdalsost
Gula Malacca

[…] один из символов **в скобках**;

Товары, в названии которых встречается комбинация букв “gu”, за которой может стоять буква “l” или “d”

[^…] любой символ **не в скобках**;

Товары, в названии которых есть комбинация букв “gu“, после которых не следует буква “a”

```
USE northwind
SELECT productname FROM products
WHERE productname LIKE '%gu[^a]%'
```

productname
Chef Anton's Gumbo Mix
Gudbrandsdalsost
Gula Malacca

Упорядочение записей, подсчет итогов

Упорядочение записей по значению поля (полей) выполняется с помощью оператора **ORDER BY**. Для упорядочения по возрастанию используется ключ **ASC** (по умолчанию), по убыванию - **DESC**.

```
USE northwind
SELECT productname FROM products
ORDER BY productname
```

productname
Alice Mutton
Aniseed Syrup
Boston Crab Meat
Camembert Pierrot

Наименование товара в алфавитном порядке

```
USE northwind
SELECT productname, Unitprice FROM products
ORDER BY unitprice DESC
```

productname	Unitprice
Côte de Blaye	263.5000
Thüringer Rostbratwurst	123.7900
Mishi Kobe Niku	97.0000

Список из наименования и цены товара, отсортированные по убыванию цены.

Выборка первых N записей с помощью ключа **TOP**. Отсортировав записи можно выбрать наилучшую (наихудшую) выборку товаров.

Десятка наиболее дорогих товаров

```
USE northwind
SELECT TOP 10 productname, Unitprice
FROM products ORDER BY unitprice DESC
```

productname	Unitprice
Côte de Blaye	263.5000
Thüringer Rostbratwurst	123.7900
Mishi Kobe Niku	97.0000
Sir Rodney's Marmalade	81.0000
Carnarvon Tigers	62.5000
Raclette Courdavault	55.0000
Manjimup Dried Apples	53.0000
Tarte au sucre	49.3000
Ipoh Coffee	46.0000
Rössle Sauerkraut	45.6000

Подсчет статистики по столбцам - функции: **Max, Min, SUM, AVG** (ср. знач.), **COUNT** (количество), **STDEV** (стандартное отклонение), **VAR** (дисперсия)

```
USE northwind
SELECT sum(Unitprice)
FROM products
```

(No column name)
2222.7100

```
USE northwind
SELECT max(Unitprice)
FROM products
```

(No column name)
263.5000

Количество товара, цена которого менее 50

```
USE Northwind
SELECT COUNT(*) AS [Кол-во товара]
FROM products WHERE unitprice<50
```

Кол-во товара
70

При выполнении оператора SELECT результирующее отношение может иметь несколько записей с одинаковыми значениями всех полей. Чтобы исключить повторяющиеся записи из выборки используется **DISTINCT**. Если указан вместо **DISTINCT** оператор **ALL**, то результат включит все строки вместе с дублями.

Выборка из нескольких таблиц.

Очень часто возникает ситуация, когда выборку данных надо производить из отношения, которое является результатом слияния (join) двух других отношений. Например, нам нужно получить из базы данных **publications** информацию о всех печатных изданиях в виде следующей таблицы:

|название_книги | год_выпуска | издательство |

Для этого СУБД предварительно должна выполнить слияние таблиц **titles** и **publishers.**
Для выполнения операции такого рода в операторе SELECT после ключевого слова FROM указывается список таблиц, по которым производится поиск данных. После ключевого слова WHERE указывается условие, по которому производится слияние.

Для выполнить данный запрос, нужно дать команду:

```
SELECT titles.title,titles.yearpub,publishers.publisher
    FROM titles,publishers
        WHERE titles.pub_id=publishers.pub_id;
```

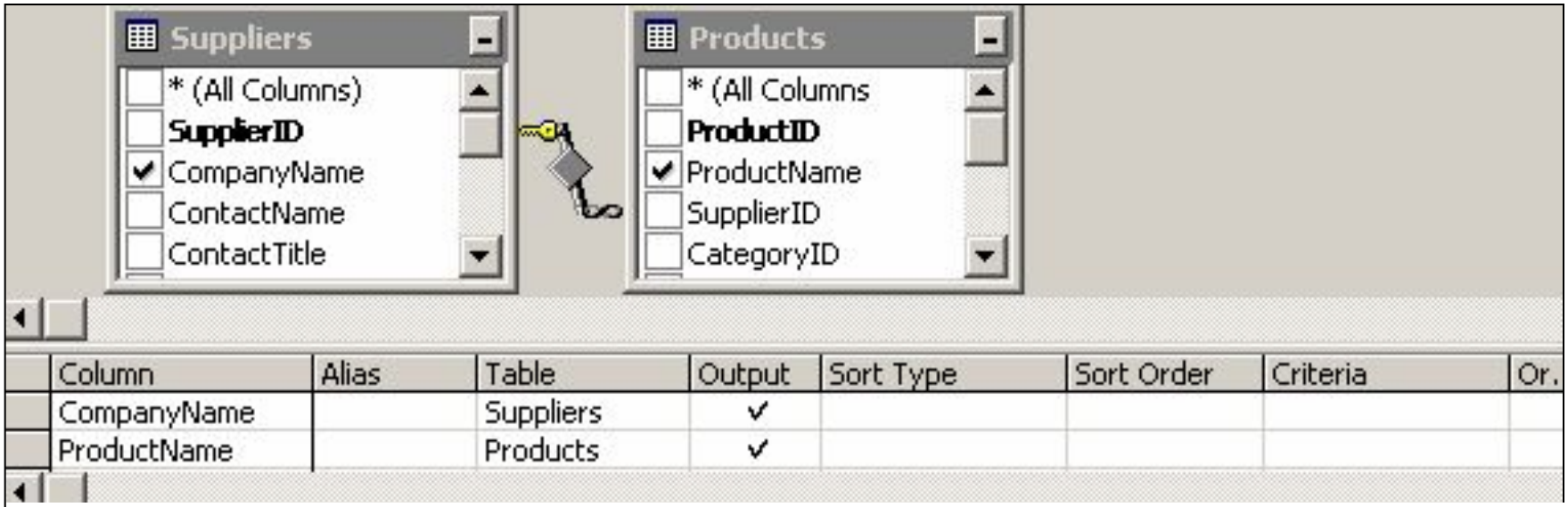
Пример, где одновременно задаются условия и слияния, и выборки (результат предыдущего запроса ограничивается изданиями после 1996 года):

```
SELECT titles.title,titles.yearpub,publishers.publisher
    FROM titles,publishers
        WHERE titles.pub_id=publishers.pub_id AND titles.yearpub>1996;
```

Имеется возможность производить слияние и более чем двух таблиц. Например, чтобы дополнить описанную выше выборку именами авторов книг необходимо составить оператор следующего вида:

```
SELECT authors.author,titles.title,titles.yearpub,publishers.publisher
  FROM titles,publishers,titleauthors
    WHERE titleauthors.au_id=authors.au_id AND
    titleauthors.title_id=titles.title_id  AND  titles.pub_id=publishers.pub_id AND
    titles.yearpub > 1996;
```

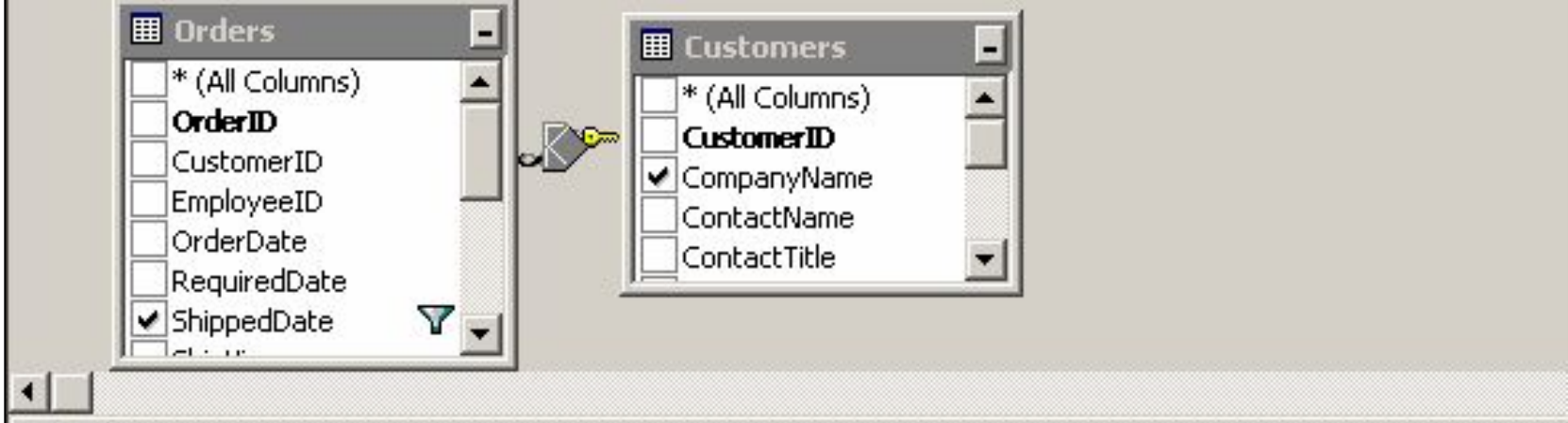
Соединение таблиц задается в секции **FROM**. Условия выборки задаются в конструкции **WHERE** (при группировке **GROUP BY -** в конструкции **HAVING)**. Типы соединений: внутреннее (выбираются те строки, которые соответствуют условию соединения), внешнее (возвращаются все строки главной таблицы - левой или правой или обоих, участвующих в соединении с учетом условия выборки).



Column	Alias	Table	Output	Sort Type	Sort Order	Criteria	Or.
CompanyName		Suppliers	✓				
ProductName		Products	✓				

```
SELECT dbo.Suppliers.CompanyName, dbo.Products.ProductName
    FROM dbo. Suppliers INNER JOIN dbo.Products
        ON dbo.Suppliers.SupplierID = dbo.Products.SupplierID
```

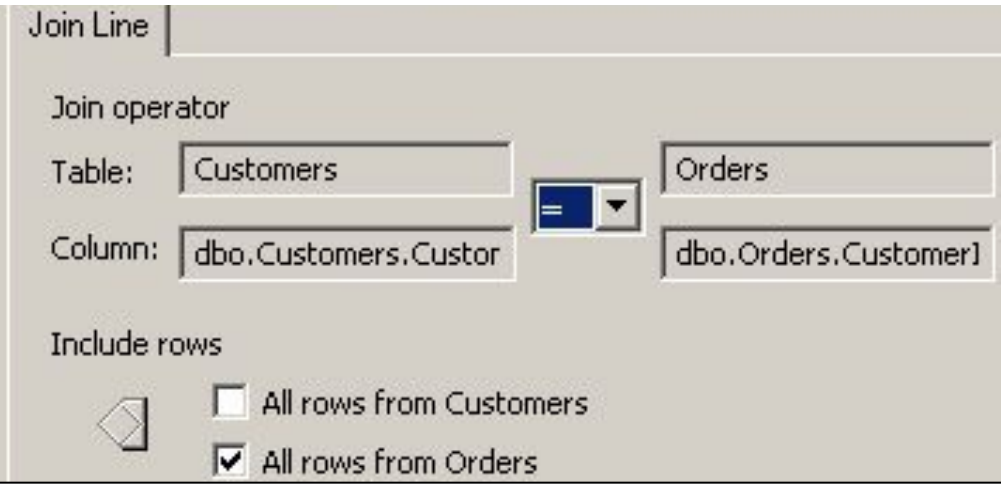
CompanyName	ProductName
Exotic Liquids	Chai
Exotic Liquids	Chang
Exotic Liquids	Aniseed Syrup
New Orleans Cajun	Chef Anton's Cajur
New Orleans Cajun	Chef Anton's Gumb
Grandma Kelly's Ho	Grandma's Boysent
Grandma Kelly's Ho	Uncle Bob's Organic



Column	Alias	Table	Output	Sort Type	Sort Order	Criteria	Or...
CompanyName		Customers	✓				
ShippedDate		Orders	✓			IS NULL	

```
SELECT dbo.Customers.CompanyName, dbo.Orders.ShippedDate
    FROM dbo. Customers RIGHT JOIN dbo.Orders
        ON dbo.Customers.CustomerID = dbo.Orders.CustomerID
            WHERE (dbo.Orders.ShippedDate IS NULL)
```

CompanyName	ShippedDate
Ernst Handel	<NULL>
Rancho grande	<NULL>
LINO-Delicateses	<NULL>
Great Lakes Food M	<NULL>
Bottom-Dollar Mark	<NULL>
La maison d'Asie	<NULL>
Cactus Comidas pa	<NULL>
Blauer See Delikate	<NULL>
Ricardo Adocicados	<NULL>
Great Lakes Food M	<NULL>
Reggiani Caseifici	<NULL>



Правое соединение – RIGHT JOIN, левое – LEFT JOIN, полное – FULL JOIN.

Таблицы Поставщики и Продукты связаны условием **INNER JOIN** по полю **SupplierID** (код поставщика).

```
USE Northwind
SELECT suppliers.companyname, products.productname
FROM suppliers INNER JOIN products
ON suppliers.supplierID=products.supplierID
```

companyname	productname
Exotic Liquids	Chai
Exotic Liquids	Chang
Exotic Liquids	Aniseed Syrup
New Orleans Cajun Delights	Chef Anton's Cajun

Для уникальных полей названия таблиц указывать не обязательно.
Можно также использовать псевдонимы **таблиц**

```
USE Northwind
SELECT s.companyname, p.productname
FROM suppliers AS s INNER JOIN products AS p
ON s.supplierID=p.supplierID
```

```
USE Northwind
SELECT Customers.CompanyName, Orders.ShippedDate
FROM Customers RIGHT JOIN Orders
ON Customers.CustomerID = Orders.CustomerID
WHERE Orders.ShippedDate Is Null
```

CompanyName	ShippedDate
Ernst Handel	NULL
Rancho grande	NULL

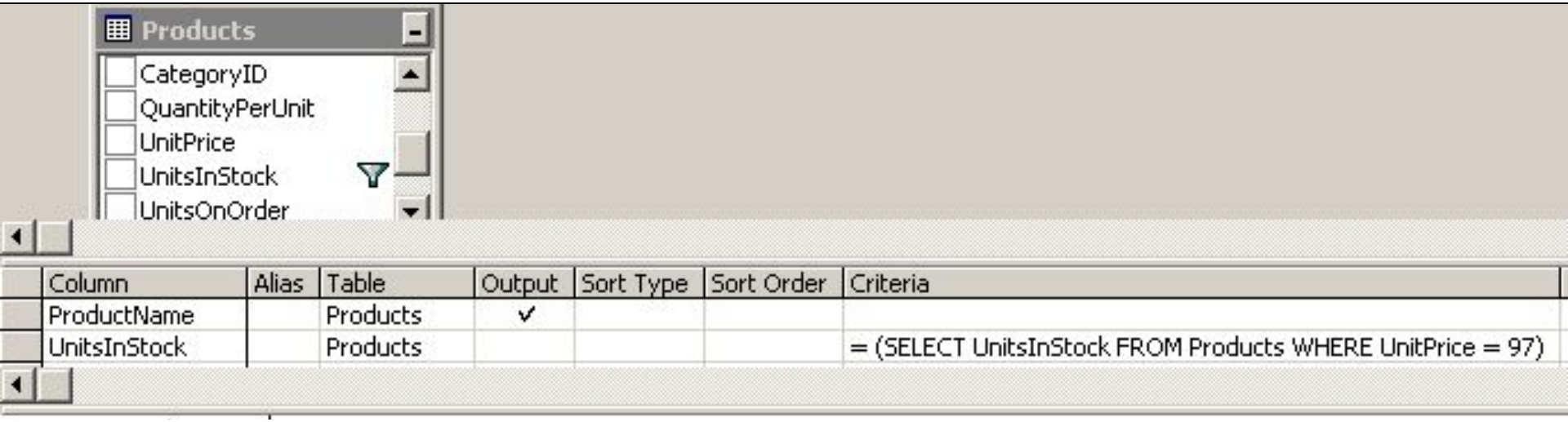
Покупатели, кому товар еще не доставлен (дата доставки – поле ShippedDate таблицы Orders (счета)).

```
SELECT DISTINCTROW Authors.Au_ID, Authors.Author
  FROM Authors LEFT JOIN [Title Author]
    ON Authors.Au_ID = [Title Author].Au_ID
      WHERE ((([Title Author].Au_ID) IS NULL))
```

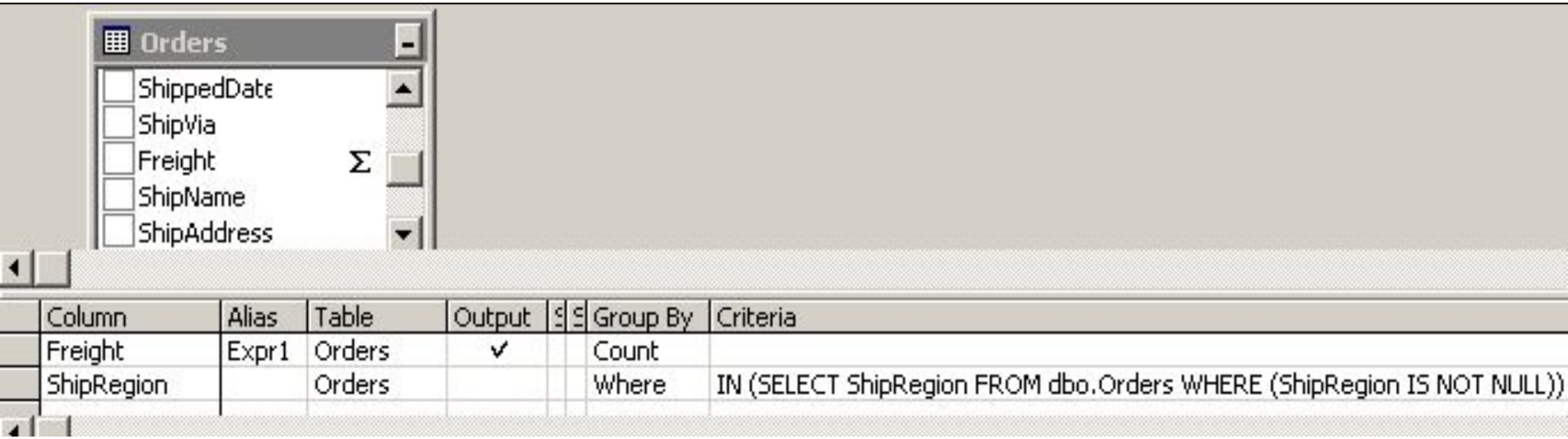
Авторы без книг

Подзапрос - запрос, вложенный во внешний оператор **SELECT, INSERT, UPDATE, DELETE.** Возвращает **одно** значение. Подзапросы могут содержать ключи **IN, ANY, ALL, EXISTS**.

Найти товар, запас (UnitsInStock) которого совпадает с выборкой товаров с ценой=97.



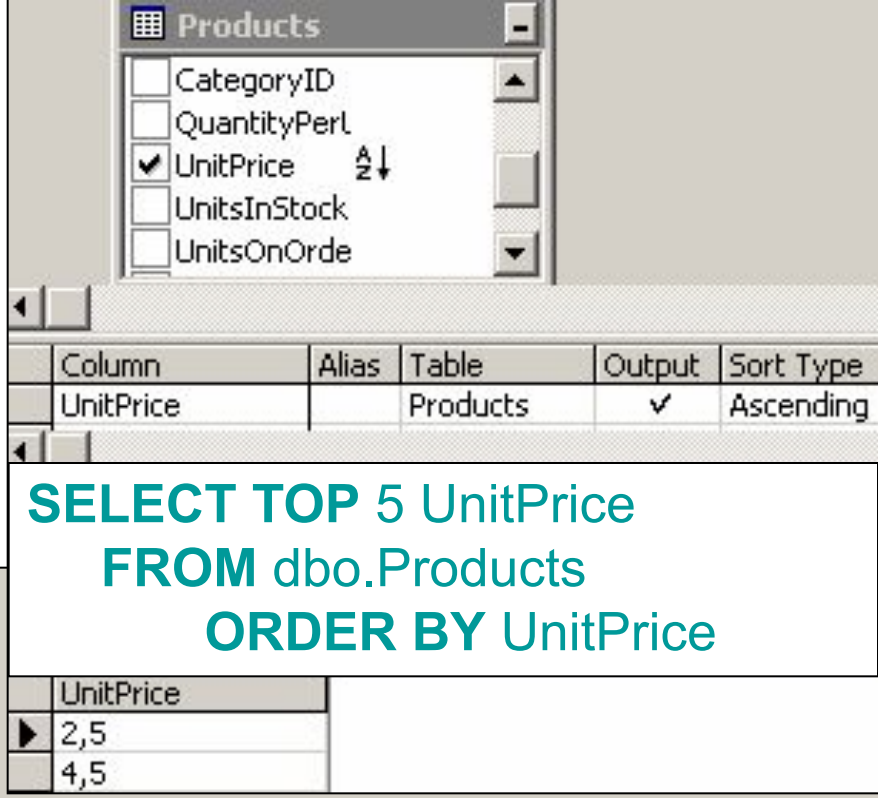

```
SELECT  ProductName FROM dbo. Products
     WHERE  (UnitsInStock =
     (SELECT UnitsInStock FROM Products WHERE UnitPrice = 97))
```

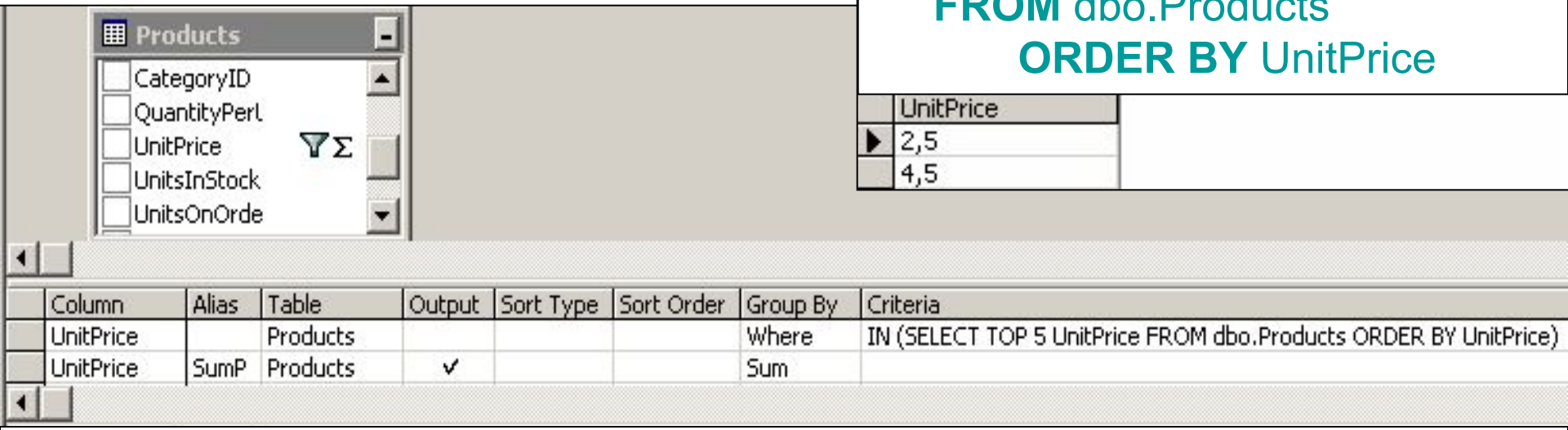



```
SELECT  COUNT(Freight) AS Expr1 FROM dbo.Orders
   WHERE (ShipRegion IN
      (SELECT ShipRegion FROM dbo.Orders
         WHERE (ShipRegion IS NOT NULL)))
```

Количество элементов в поле *Freight* таблицы *Orders* для которых в поле *ShipRegion* нет пустых значений.

Найти сумму цен 5 дешевых товаров (создадим подзапрос, затем в основной запрос включим текст подзапроса):

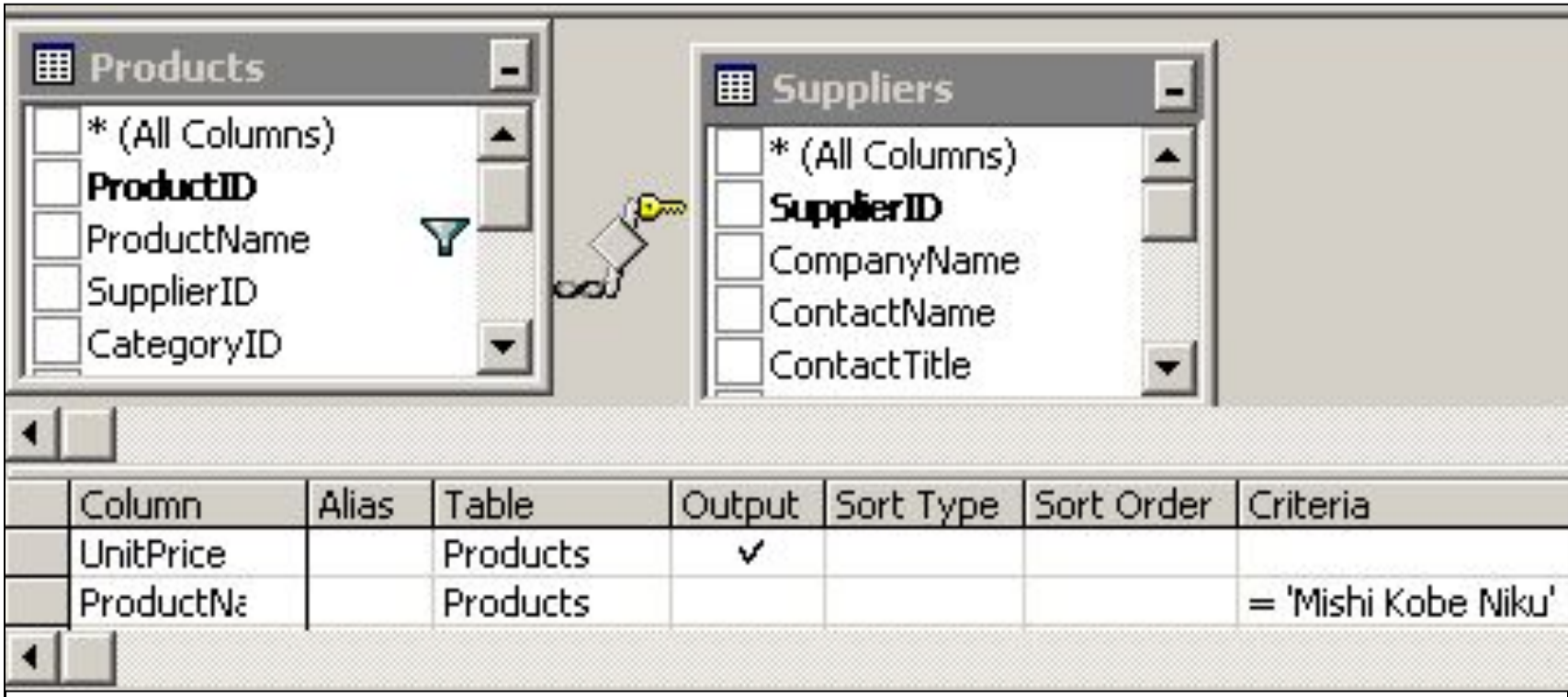





```
SELECT SUM(UnitPrice) AS SumP
    FROM dbo.Products
        WHERE (UnitPrice IN
            (SELECT TOP 5 UnitPrice FROM dbo. Products ORDER BY UnitPrice))
```

SumP
27,45

Найти товары, цена за единицу которых больше, чем у продукта “Mishi Kobe Niku” (создадим подзапрос, затем включим его в запрос)



Column	Alias	Table	Output	Sort Type	Sort Order	Criteria
UnitPrice		Products	✓			
ProductNa		Products				= 'Mishi Kobe Niku'

```
SELECT dbo.Products.UnitPrice
    FROM dbo.Products INNER JOIN dbo.Suppliers
        ON dbo.Products.SupplierID = dbo.Suppliers.SupplierID
            WHERE (dbo.Products.ProductName = 'Mishi Kobe Niku')
```

UnitPrice
97

Ключевые слова **ALL** и **ANY** сравнивают скалярное значение с набором значений **одного столбца**.

Ключ **ALL** применяется ко всем значениям, **ANY** – как минимум к одному.

Column	Table	Output	Sort	Criteria
ProductName	Product:	✓		
UnitPrice	Product:			> ANY (SELECT dbo.Products.UnitPrice FROM dbo.Pro

```
SELECT dbo.Priducts.ProductName FROM  dbo.Products
WHERE  (dbo.Products.UnitPrice > ANY
  (SELECT dbo.Products.UnitPrice
    FROM dbo.Products INNER JOIN dbo.Suppliers
      ON dbo.Products.SupplierID = dbo.Suppliers.SupplierID
        WHERE  (dbo.Products.ProductName = 'Mishi Kobe Niku')))
```

ProductName
Thüringer Rostbrat
Côte de Blaye

Товары, цена за единицу которых больше, чем у продукта “Mishi Kobe Niku”

Ключевое слово **EXISTS** проверяет наличие атрибута.

Оператор **WHERE** внешнего запроса проверяет, существуют ли строки, соответствующие подзапросу. Результат конструкции WHERE – **TRUE** или **FALSE.**

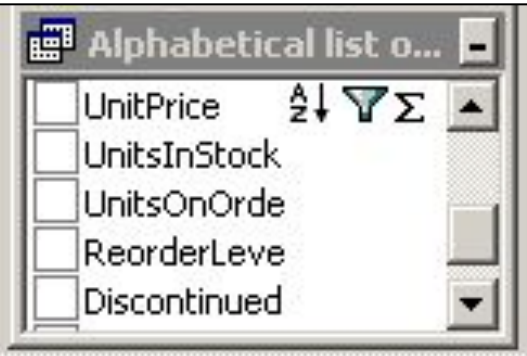
Column	Table	Output	Criteria
CompanyName	Suppliers	✓	
EXISTS (SELECT * FROM products WHERE suppliers.supplierid = products.supplierid AND unitprice = 1			= TRUE

```
SELECT DISTINCT dbo.Suppliers.CompanyName
  FROM dbo.Products INNER JOIN dbo.Suppliers
    ON dbo.Products.SupplierID = dbo.Suppliers.SupplierID
      WHERE EXISTS (SELECT * FROM products
        WHERE suppliers.supplierID = products.supplierID AND unitprice = 14)
```

CompanyName
Bigfoot Breweries
Heli Süßwaren Gmb
Leka Trading

Поставщики, цена товара которых = 14. Попутно выполняется проверка наличия таких продуктов от поставщиков.

Группировка строк



Выбрать 100 категорий тех товаров из запроса *Alphabetical list of products,* у которых средняя (**AVG**) цена находится в интервале от 10 до 50.

Column	Alias	Table	Output	Sort Type	Sort Order	Group By
CategoryName		[Alphabetical list of products]	✓			Group By
UnitPrice	AVGPrice	[Alphabetical list of products]	✓	Ascending	1	Avg

```
SELECT TOP 100 CategoryName, AVG(UnitPrice) AS AVGPrice
    FROM dbo.[Alphabetical list of products]
        WHERE  (UnitPrice BETWEEN 10 AND 50)
            GROUP BY CategoryName ORDER BY AVG(UnitPrice)
```

CategoryName	AVGPrice
Beverages	20
Seafood	20,0675
Produce	21,0833
Confections	22,738
Condiments	23,2181
Meat/Poultry	24
Grains/Cereals	27,9375
Dairy Products	28,725

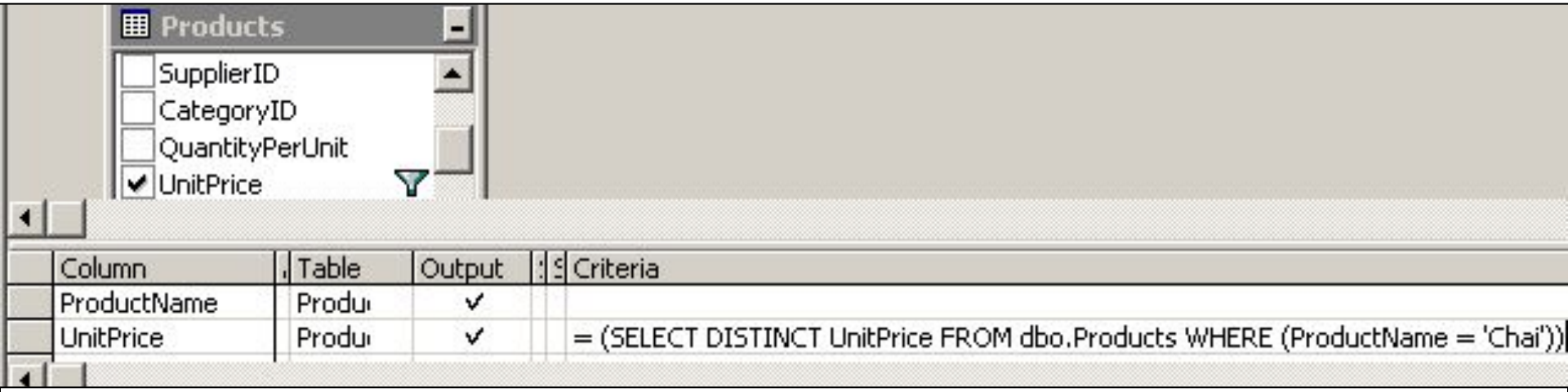
Дополнительную фильтрацию выполним с помощью выражения **HAVING**.

Column	Table	Output	S	S	Group By	Criteria
CategoryName	[Alphabetical list of product:	✓			Group By	
UnitPrice	[Alphabetical list of product:	✓			Avg	< 25

```
SELECT CategoryName, AVG(UnitPrice) AS AVGPrice
    FROM  dbo.[Alphabetical list of products]
        GROUP BY CategoryName
            HAVING (AVG(UnitPrice) < 25)
```

CategoryName	AVGPrice
Condiments	23,2181
Grains/Cereals	21,2916
Meat/Poultry	15,725
Seafood	20,6825

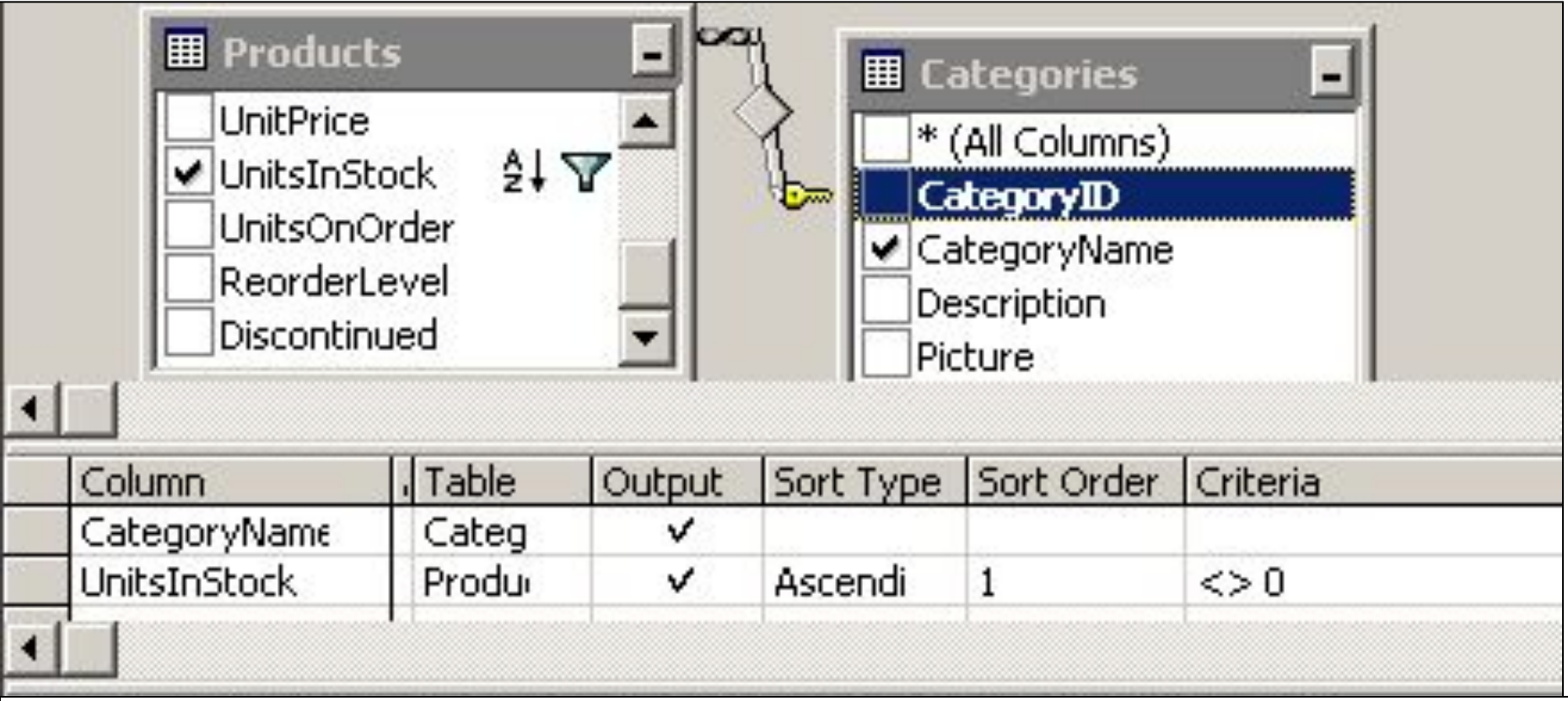
Товары, средняя цена которых < 25




```
SELECT  ProductName, UnitPrice  FROM  dbo.Products
  WHERE  (UnitPrice =
     (SELECT DISTINCT UnitPrice  FROM  dbo.Products
         WHERE  ProductName = 'Chai'))
```

ProductName	UnitPrice
Chai	18
Steeleye Stout	18
Chartreuse verte	18
Lakkalikööri	18

Названия и цена продуктов, совпадающих по цене с товаром “Chai”.

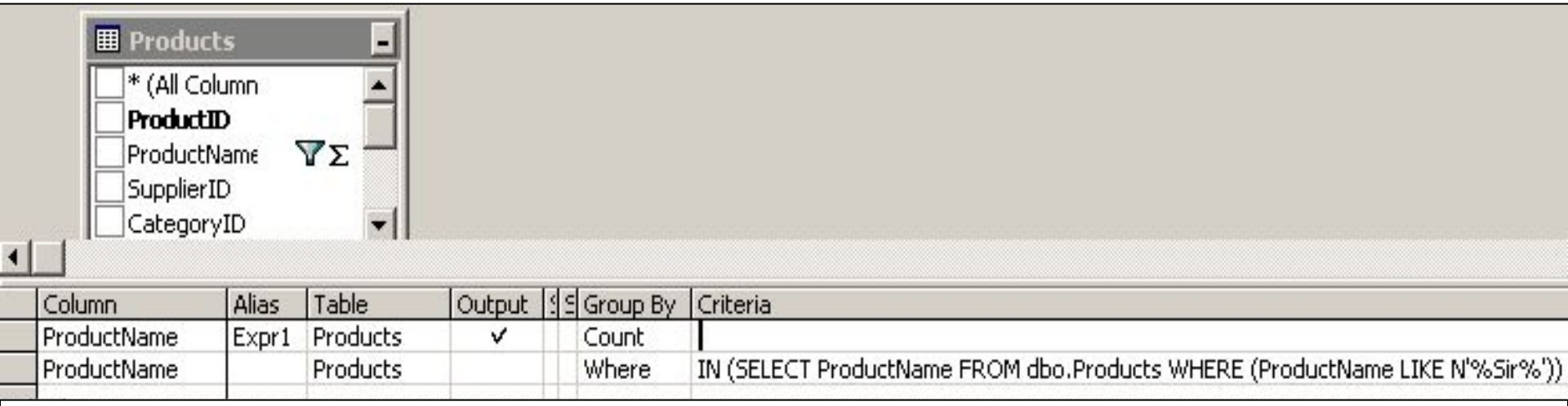


Column	Table	Output	Sort Type	Sort Order	Criteria
CategoryName	Categ	✓			
UnitsInStock	Produ	✓	Ascendi	1	<> 0

```
SELECT TOP 3 dbo.Categories. CategoryName, dbo.Products.UnitsInStock
  FROM dbo.Products INNER JOIN dbo.Categories
    ON dbo.Products.CategoryID = dbo.Categories.CategoryID
      WHERE (dbo.Products.UnitsInStock <> 0)
        ORDER BY dbo.Products.UnitsInStock
```

CategoryName	UnitsInStock
Confections	3
Condiments	4
Produce	4

3 категории товаров, запасы которых минимальны (сортировка по убыванию значения запаса)



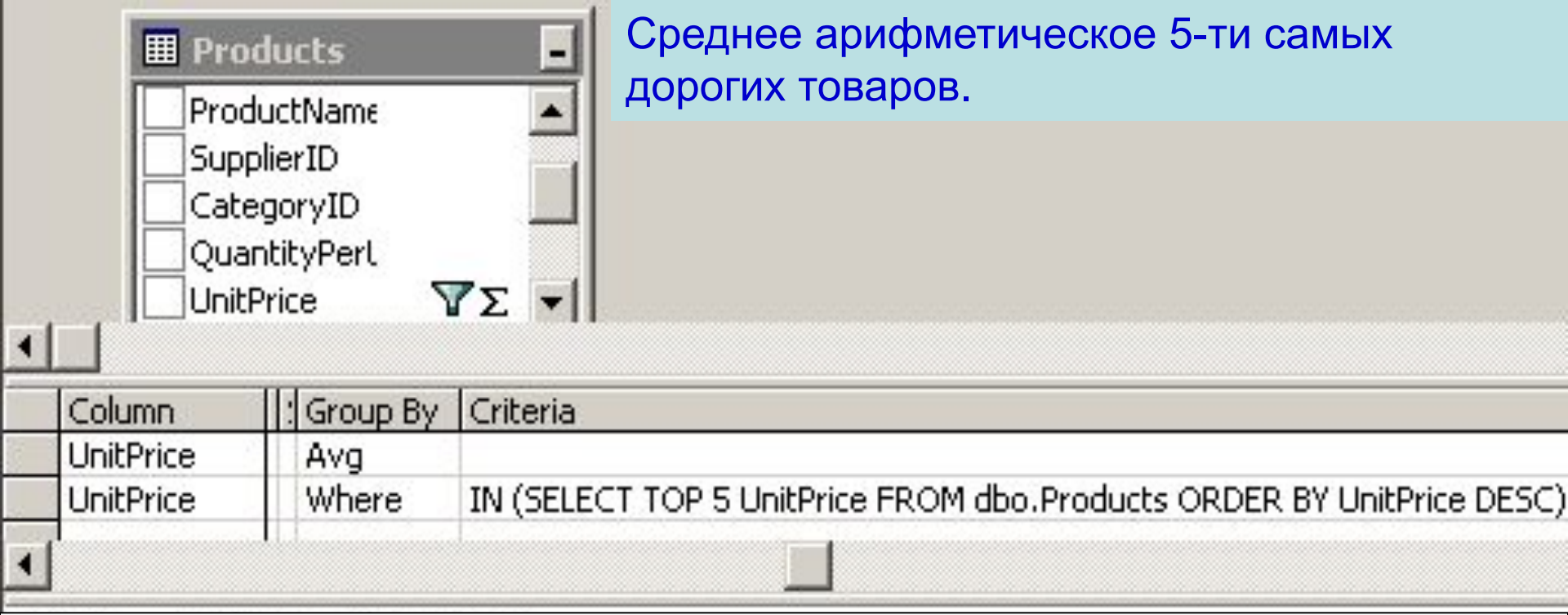
Column	Alias	Table	Output	Group By	Criteria
ProductName	Expr1	Products	✓	Count	
ProductName		Products		Where	IN (SELECT ProductName FROM dbo.Products WHERE (ProductName LIKE N'%Sir%'))

```
SELECT COUNT(ProductName) AS Expr1 FROM dbo.Products
  WHERE (ProductName IN
     (SELECT ProductName FROM dbo.Products
     WHERE ProductName LIKE '%S%'))
```

Expr1
3

Количество продуктов, в названии которых встречается слово ‘Sir’

Среднее арифметическое 5-ти самых дорогих товаров.




```
SELECT  AVG(UnitPrice)  AS  Expr1
   FROM   dbo.Products
      WHERE  (UnitPrice  IN
         (SELECT TOP  5  UnitPrice  FROM  dbo.Products
            ORDER  BY  UnitPrice  DESC))
```

Expr1
125,558

Запрос после конструктора можно подкорректировать вручную.
Необходимо проверять результат, например, в MS Excel

5 кафедр, число сотрудников которых максимально

```
SELECT TOP 5
сотрудник.[Код кафедры], Count(сотрудник.ФИО) AS число_сотрудников
        FROM сотрудник
            GROUP BY сотрудник.[Код кафедры]
                ORDER BY Count(сотрудник.ФИО) DESC;
```

ФИО и премии сотрудников

```
SELECT сотрудник.ФИО, [оклад]*0.5 AS Премия FROM сотрудник;
```

Средний оклад сотрудников

```
SELECT AVG(сотрудник.оклад) AS Ср_оклад FROM сотрудник;
```

Список сотрудников, у которых нет детей

```
SELECT DISTINCTROW сотр.ФИО FROM сотр LEFT JOIN дети
      ON сотрудник.Код=дети.Код_сотр
         WHERE (((дети.Код_сотр) IS NULL));
```

Сотрудники с окладами в диапазоне от 1000 до 2000

```
SELECT  сотрудник.ФИО, сотрудник.оклад FROM сотрудник
     WHERE (((сотрудник.оклад) Between 1000 And 2000));
                    или Between 2000 And 1000
```

Сотрудники с окладами менее 1000 и более 15000

```
SELECT  сотрудник.ФИО, сотрудник.оклад FROM сотрудник
   WHERE (((сотрудник.оклад)<1000 Or (сотрудник.оклад)>15000));
```

Сотрудники с окладами вне диапазона от 1000 до 2000

```
SELECT  сотрудник.ФИО, сотрудник.оклад FROM сотрудник
   WHERE (((сотрудник.оклад) Not Between 1000 And 2000));
```

Сотрудники с ФИО, начинающимися на букву “Д”

```
SELECT  сотрудник.ФИО FROM сотрудник
      WHERE (((сотрудник.ФИО) Like "Д*"));
```

Кафедры с числом сотрудников более 5 человек

```
SELECT  сотр.[Код кафедры], Count(сотр.ФИО) AS [Число_сотрудников]
    FROM сотр
       GROUP BY сотр.[Код кафедры] HAVING (((Count(сотр.ФИО))>5));
```

Таблицы фирма, сотрудники и экзамены. Сколько аттестованных сотрудников есть на **каждой** фирме (один сотрудник может быть аттестован по нескольким пунктам, поэтому **DISTINCT**. Предварительно для каждой фирмы проверяется наличие аттестованных сотрудников).

```
SELECT сотр.фирма, Count(сотр.сотр) AS аттест_сотр  FROM сотр
  WHERE (EXISTS (SELECT DISTINCT сотр.фирма, сотр.сотр
      FROM сотр INNER JOIN экзам ON сотр.сотр = экзам.сотр)) <>False
        GROUP BY сотр.фирма;
```

Удалить записи с ФИО “Бурлак”

```
DELETE * FROM студент WHERE студент.ФИО="Бурлак";
```

Минимальная, максимальная и средняя зарплата

```
SELECT MIN(зарплата), MAX(зарплата), AVG(зарплата)
    FROM сотрудники;
```

Самая короткая фамилия

```
SELECT MIN(ФИО) FROM сотрудники;
```

Самая короткая фамилия

```
SELECT COUNT(*), COUNT(налог) FROM сотрудники;
```

Вместо хранения вычисляемых полей в таблицах …

```
SELECT OrderID, Сумма FROM Orders
```

… их можно вычислить в запросе, т.е. “на лету”

```
SELECT OrderID, SUM(Цена* Количество * (1.0 - скидка)) AS Сумма
  FROM Order Details
    GROUP BY OrderID
```

Счета, имеющие по крайней мере одну запись со скидкой (поле Discount) – второй вариант без подзапроса выполняется в 3 раза быстрее

```
SELECT OrderID FROM Orders O
  WHERE EXISTS
    (SELECT OrderID FROM OrderDetails OD
        WHERE O.OrderID = OD.OrderID AND Discount >= 0.25)

SELECT DISTINCT O.OrderID
  FROM Orders O INNER JOIN OrderDetails OD
    ON O.OrderID = OD.OrderID
    WHERE Discount >= 0.25
```

- Не имеет значения **порядок перечисления операндов** в операторах команды SELECT
  - Порядок таблиц в операторе FROM
    - “… **FROM** Foo, Bar …” ==
    - “… **FROM** Bar, Foo …”
  - Порядок операндов при проверке условия
    - “… **WHERE** Col1 = 2 **AND** Col2 > 10 …” ==
    - “… **WHERE** Col2 > 10 **AND** Col1 = 2 …”
  - Порядок таблиц во внутреннем соединении
    - “… **FROM** Foo **INNER JOIN** Bar …” ==
    - “… **FROM** Bar **INNER JOIN** Foo …”

Оператор **HAVING** выполняется в 3 раза медленнее, чем оператор **WHERE**

```
SELECT CustomerID, COUNT(CustomerID)  FROM Orders
   GROUP BY CustomerID
      HAVING CustomerID >= 'A' AND CustomerID < 'B‘

SELECT CustomerID, COUNT(CustomerID)  FROM Orders
   WHERE CustomerID >= 'A' AND CustomerID < 'B'
      GROUP BY CustomerID
```

Количество покупателей, коды которых начинаются с “A”

Можно сцеплять поля выборки (таблица СТУДЕНТ имеет поля ФАМ_СТУД и НОМ_ЗАЧ):

```
SELECT фам_студ & " имеет " & ном_зач AS Список FROM студент;
```

```
Список
------------------------
Velasquez получает 2500
Ngao получает 1450
Nagayama получает 1400
```

Операции над датами

Операция	Результат	Описание
дата + число	**Дата**	**Прибавление количества дней к дате.**
дата - число	**Дата**	**Вычитание количества дней из даты.**
дата - дата	**Число**	**Вычитание одной даты из другой, результат – количество дней между датами.**
Дата + число/24	**Дата**	**Прибавление часов к дате.**

```
SELECT DATE, DATE-7, DATE+18 FROM dual;
```

Результат:

```
DATE      DATE-7    DATE+18
--------  --------  --------
27.09.11  20.09.11  08.11.11
```

СОЗДАНИЕ ТАБЛИЦ

CREATE TABLE <имя_таблицы>(<имя_столбца><тип_столбца>
[NOT NULL] [UNIQUE | PRIMARY KEY]
[REFERENCES <имя_главной_таблицы> [<имя_столбца>]] , ...)

Для каждого столбца обязательно указываются имя и тип, а также опционально могут быть указаны следующие параметры:

- **NOT NULL** - элементы столбца всегда должны иметь определенное значение (не NULL)
- один из взаимоисключающих параметров:
  - **UNIQUE** - значение каждого элемента столбца должно быть уникальным или
  - **PRIMARY KEY** - столбец является первичным ключом.
- **REFERENCES** <имя_главной_таблицы> [<имя_столбца>] - эта конструкция определяет, что данный столбец является внешним ключом и указывает на ключ какой главной таблицы он ссылается.

Создание БД **publications (ПУБЛИКАЦИИ) из таблиц** authors, publishers, titles, titleautors, wwwsites, wwwsiteauthors

```
CREATE DATABASE publications;
CREATE TABLE authors (au_id INT PRIMARY KEY,
                      author VARCHAR(25) NOT NULL);
CREATE TABLE publishers (pub_id INT PRIMARY KEY,
                         publisher VARCHAR(255) NOT NULL,
                         url VARCHAR(255));
CREATE TABLE titles (title_id INT PRIMARY KEY,
                     title VARCHAR(255) NOT NULL, yearpub INT,
                     pub_id INT REFERENCES publishers(pub_id));
CREATE TABLE titleautors (au_id INT REFERENCES authors(au_id),
                          title_id INT REFERENCES titles(title_id));
CREATE TABLE wwwsites (site_id INT PRIMARY KEY,
                       site VARCHAR(255) NOT NULL,
                       url VARCHAR(255));
CREATE TABLE wwwsiteauthors (au_id INT REFERENCES authors(au_id),
                             site_id INT REFERENCES wwwsites(site_id));
```

Удаление таблицы:

DROP TABLE <имя_таблицы>

Модификация таблицы:

- Добавить столбцы

ALTER TABLE <имя_таблицы>
ADD (<имя_столбца> <тип_столбца> **[NOT NULL]**
[UNIQUE | PRIMARY KEY]
[REFERENCES <имя_главной_таблицы> [<имя_столбца>]] ,...)

- Удалить столбцы

ALTER TABLE <имя_таблицы>
DROP (<имя_столбца>,...)

- Модификация типа столбцов

ALTER TABLE <имя_таблицы>
MODIFY (<имя_столбца> <тип_столбца> **[NOT NULL]**
[UNIQUE | PRIMARY KEY]
[REFERENCES <имя_главной_таблицы> <имя_столбца>]] ,...)

4. Команды модификации данных.

К этой группе относятся операторы добавления, изменения и удаления записей.

Добавить новую запись в таблицу:

INSERT INTO <имя_таблицы>[(<имя_столбца>,<имя_столбца>,...)]
VALUES (<значение>,<значение>,..)

Пример с указанием списка столбцов:

```
INSERT INTO publishers (publisher, pub_id)
        VALUES ("Super Computer Publishing",17);
```

Список столбцов в команде не является обязательным параметром. В этом случае должны быть указаны значения для всех полей таблицы в том порядке, как эти столбцы были перечислены в команде CREATE TABLE, например:

```
INSERT INTO publishers
    VALUES (16,"Microsoft Press", "http://www.microsoft.com");
```

Модификация записей:

UPDATE <таблица> **SET** столбец>=<значение>,...[**WHERE** <условие>]

Если условие не задано, UPDATE применяется ко всем записям.

UPDATE publishers **SET** url="http://www.superpub.com" **WHERE** pub_id=17;

Логические выражения над константами и полями:

- операции сравнения: > , < , >= , <= , = , <> , != могут применяться не только к числовым значениям, но и к строкам и датам.
- операции проверки поля **IS NULL, IS NOT NULL**
  - операции проверки на вхождение в диапазон: BETWEEN и NOT BETWEEN.
  - операции проверки на вхождение в список: IN и NOT IN
  - операции проверки на вхождение подстроки: LIKE и NOT LIKE
  - отдельные операции соединяются связями AND, OR, NOT и группируются с помощью скобок.

Пример: Найти в таблице **publishers** все неопределенные значения столбца **url** и заменить их строкой "url not defined".

UPDATE publishers **SET** url="url not defined" **WHERE** url **IS NULL**;

Удаление записей

DELETE FROM <имя_таблицы> **[WHERE** <условие> **]**

Если ключевое слово WHERE и условие отсутствуют, из таблицы удаляются все записи.

Пример: удаляет запись об издательстве Super Computer Publishing

```
DELETE FROM publishers WHERE publisher = "Super Computer Publishing";
```