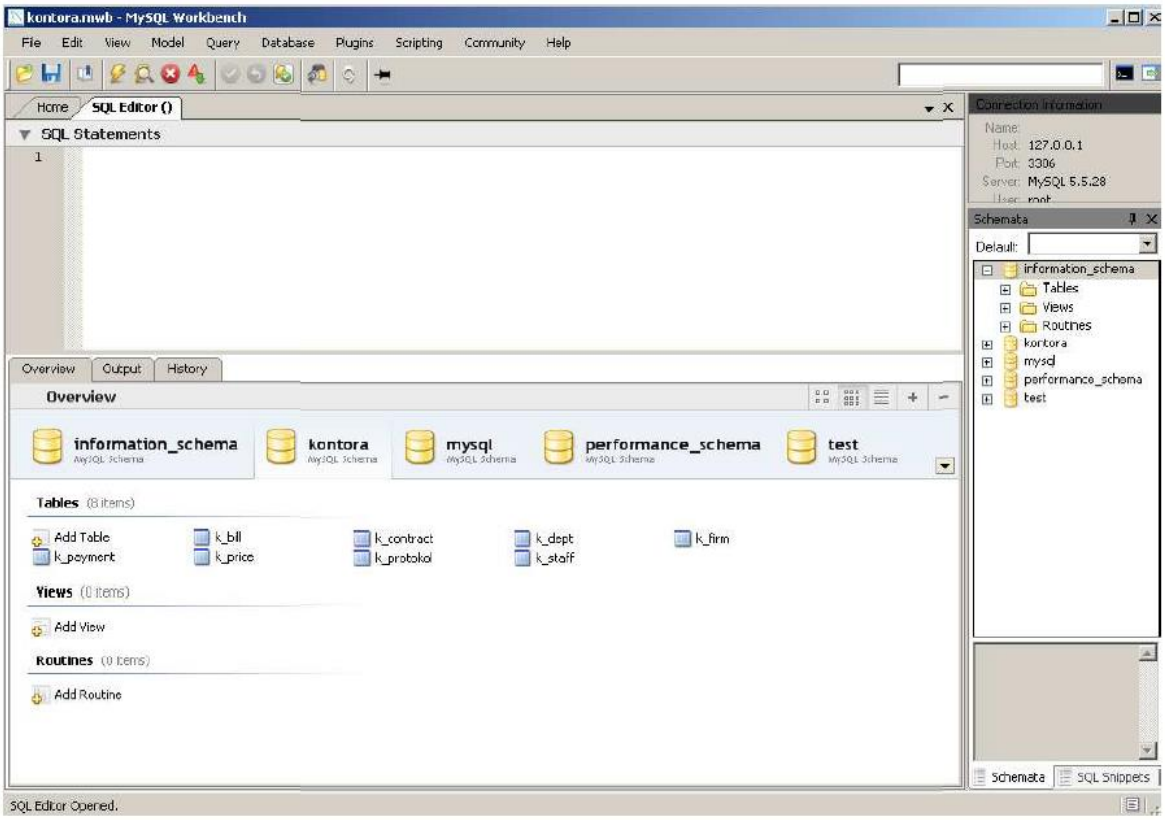
Запросы к базе данных

На предыдущем этапе мы заполнили таблицы базы данных. Рассмотрим теперь команду SELECT, позволяющую выполнять запросы к базе данных. По поводу этой команды написаны целые книги, здесь мы кратко на примерах рассмотрим ее основные возможности.

Как и в предыдущем задании, подключимся к серверу. Для этого в секции

щелкнем по ссылке Open Connection to Start Querying. В открывшемся окне нужно задать **username** и **password**, и нажать на кнопку «ОК».

Команды SELECT нужно набирать в окне **SQL Statements:**



Самой первой командой в сеансе данных должна быть команда выбора текущей базы данных:

```
USE kontora;
```

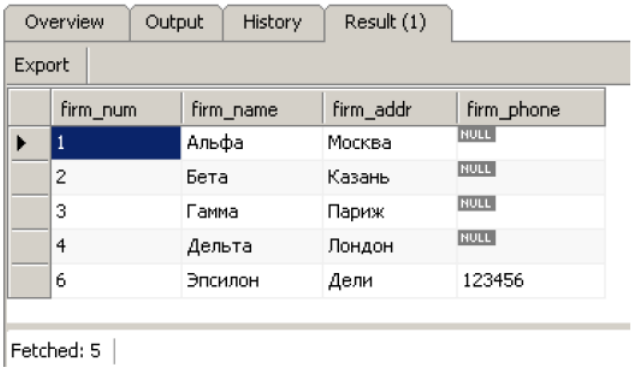
Выборка из одной таблицы

Обязательные ключевые слова команды – SELECT и FROM.

Тривиальная выборка всех полей и всех строк одной таблицы выполняется следующим образом.

Получить полную информацию обо всех предприятиях:

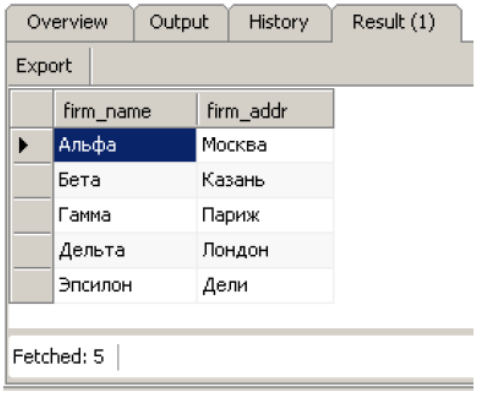
```
SELECT * FROM  k_firm
```



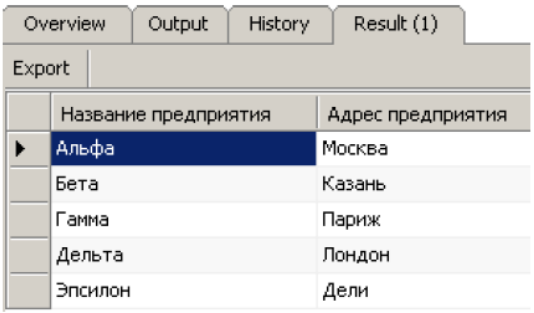
Выбор отдельных полей таблицы.

Получить названия и адреса всех предприятий:

```
SELECT firm_name, firm_addr  FROM  k_firm
```



Поля выборки можно переименовывать. Если новое название состоит из нескольких слов, помещайте его в кавычки.



Overview | Output | History | Result (1)

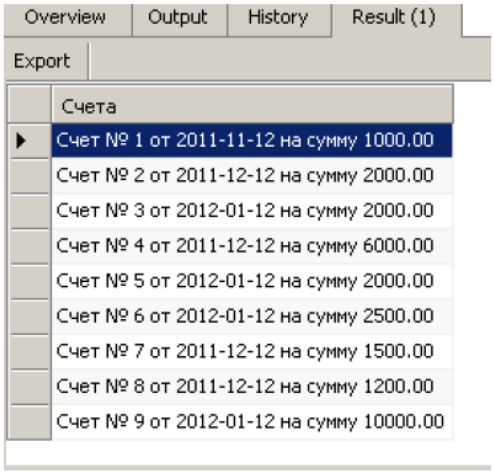
Export

	Название предприятия	Адрес предприятия
▶	Альфа	Москва
	Бета	Казань
	Гамма	Париж
	Дельта	Лондон
	Эпсилон	Дели

```
SELECT firm_name
  AS "Название предприятия",
    firm_addr
  AS "Адрес предприятия"
FROM  k_firm
```

В списке полей выборки можно использовать выражения. В этом случае часто приходится преобразовывать данные из одного типа в другой с помощью функции CONVERT. Строковые константы следует помещать в одинарные кавычки. Функция CONCAT служит для сцепления строк.

Распечатать информацию о счетах:



Overview | Output | History | Result (1)

Export

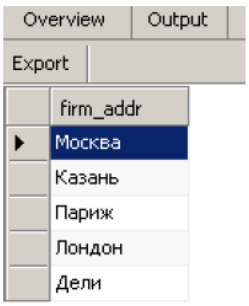
	Счета
▶	Счет № 1 от 2011-11-12 на сумму 1000.00
	Счет № 2 от 2011-12-12 на сумму 2000.00
	Счет № 3 от 2012-01-12 на сумму 2000.00
	Счет № 4 от 2011-12-12 на сумму 6000.00
	Счет № 5 от 2012-01-12 на сумму 2000.00
	Счет № 6 от 2012-01-12 на сумму 2500.00
	Счет № 7 от 2011-12-12 на сумму 1500.00
	Счет № 8 от 2011-12-12 на сумму 1200.00
	Счет № 9 от 2012-01-12 на сумму 10000.00

```
SELECT CONCAT("Счет № ",
CONVERT(bill_num, CHAR),
  " от ", CONVERT(bill_date, CHAR),
  " на сумму ",
CONVERT(bill_sum,CHAR)) AS Счета
  FROM k_bill
```

Для того чтобы исключить дубликаты строк, нужно использовать ключевое слово DISTINCT.

Напечатать список городов, в которых находятся предприятия-клиенты:

```
SELECT DISTINCT firm_addr FROM k_firm
```



Overview | Output

Export

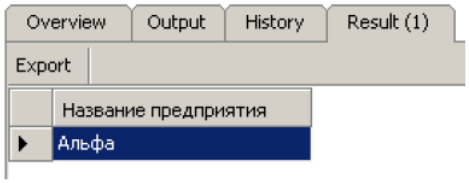
	firm_addr
▶	Москва
	Казань
	Париж
	Лондон
	Дели

Использование условий отбора

Для выбора отдельных строк по некоторому критерию используется ключевое слово WHERE

Получить список предприятий, расположенных в Москве:

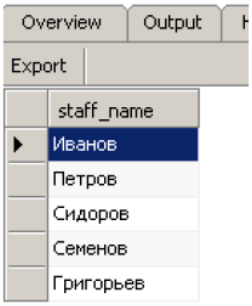
```
SELECT    firm_name as "Название предприятия"
    FROM  k_firm
    WHERE firm_addr='Москва'
```



Для сравнения поля со значением NULL нельзя использовать операции = и !=, вместо них нужно использовать выражения IS NULL и IS NOT NULL.

Получить список постоянно работающих сотрудников, т.е., таких, у которых staff_termdate равно NULL:

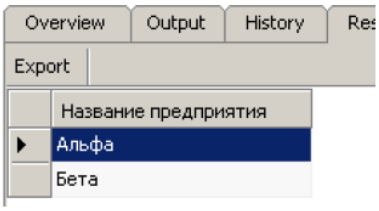
```
SELECT staff_name FROM k_staff
    WHERE staff_termdate IS NULL
```



Условия могут быть сложные, представляющие собой комбинацию нескольких операций сравнения. В них можно использовать логические связки AND и OR, а также отрицание NOT.

Получить список предприятий, расположенных в Москве или Казани:

```
SELECT  firm_name as "Название предприятия" FROM  k_firm
    WHERE firm_addr='Москва' OR firm_addr='Казань'
```



Если условие заключается в сравнении поля со **списком** значений, удобно использовать ключевое слово IN.

Получить список предприятий, расположенных в Москве или Казани:

```
SELECT    firm_name as "Названиe предприятия"
     FROM  k_firm
WHERE firm_addr IN ('Москва','Казань')
```

Если условие заключается в сравнении поля с **диапазоном** значений, удобно использовать ключевое слово BETWEEN.

Получить список договоров, заключенных в ноябре 2011 г.:

```
SELECT  * FROM  k_contract
WHERE contract_date BETWEEN '2011-11-01' AND '2011-11-30'
```

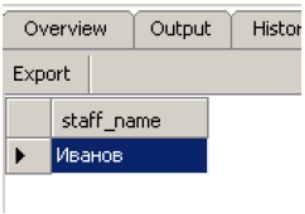
Overview | Output | History | Result (1)

Export

	contract_num	contract_date	contract_type	k_staff_staff_num	k_firm_firm_num
▶	1	2011-11-01	A	1	1
	4	2011-11-15	A	2	2
	7	2011-11-12	A	1	4

Для полей строкового типа можно применять сравнение с подстрокой.

Получить список сотрудников, фамилия которых начинается на И:



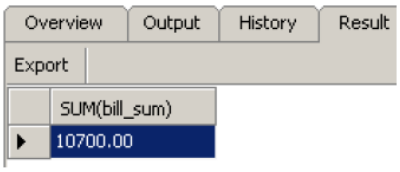

```
SELECT  staff_name FROM  k_staff
     WHERE staff_name LIKE 'И%'
```

Использование агрегирующих функций

Для подсчета **итоговых** значений служат функции SUM, COUNT, MAX, MIN, AVG. Если не используется группировка строк, запрос с применением итоговой функции вернет *ровно* одну строку.

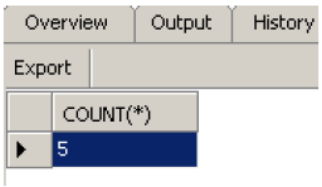
Подсчитать, на какую сумму выставлены счета в декабре 2011 года.

```
SELECT  SUM(bill_sum) FROM  k_bill
     WHERE bill_date
          BETWEEN '2011-12-01' AND '2011-12-31'
```



Функция COUNT позволяет подсчитать, сколько строк в таблице имеется вообще.

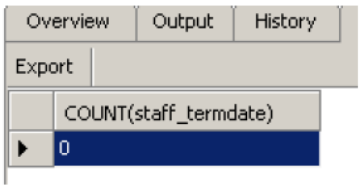
Подсчитать количество сотрудников.




```
SELECT COUNT(*) FROM k_staff
```

А также эта функция позволяет подсчитать, сколько строк с не-NULL-значениями в определенном поле.

Подсчитать количество временно работающих сотрудников (у них заполнен срок окончания трудового договора – поле staff_termdate). Предполагается, что даты все разные (точнее говоря, здесь подсчитывается количество различных не-null значений).



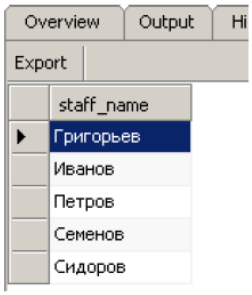

```
SELECT COUNT(staff_termdate)
FROM k_staff
```

Сортировка

Для сортировки используется ключевое слово ORDER BY и имя поля или его номер в списке полей выборки.

Напечатать список сотрудников, отсортированный по алфавиту:

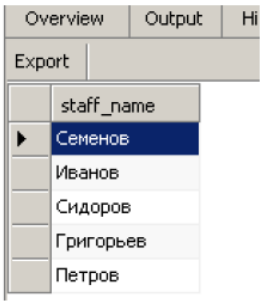
```
SELECT staff_name FROM k_staff ORDER BY 1
```



Можно сортировать строки даже по такому полю, которое не входит в список полей выборки.

Напечатать список сотрудников, отсортированный по дате поступления на работу:

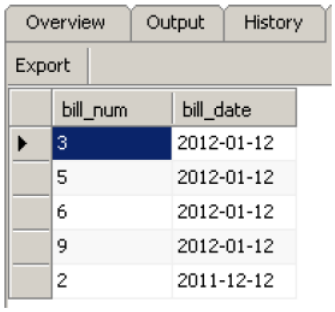
```
SELECT staff_name FROM k_staff ORDER BY staff_hiredate
```



Сортировать данные можно и по убыванию. Кроме того, можно ограничить количество строк в результате.

Напечатать информацию о 5 последних выписанных счетах в порядке убывания даты счета:

```
SELECT bill_num, bill_date
FROM k_bill ORDER BY bill_date DESC LIMIT 5
```



	bill_num	bill_date
▶	3	2012-01-12
	5	2012-01-12
	6	2012-01-12
	9	2012-01-12
	2	2011-12-12

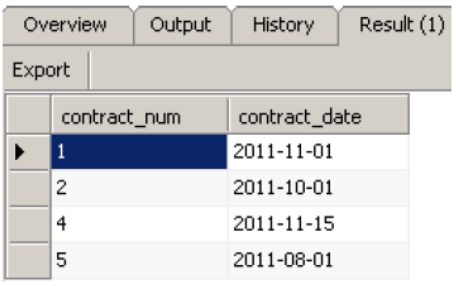
Подзапросы

Для более сложных формулировок иногда удобно использовать подзапросы. Подзапрос всегда указывается в скобках.

Подзапрос может быть **несвязанным**, т.е. в формулировке подзапроса нет ссылки на главный запрос. В этом случае подзапрос выполняется один раз при выполнении главного запроса. В данном примере используется ключевое слово IN, так как подзапрос может возвращать несколько значений.

Получить список договоров, по которым в декабре 2011 года выписаны счета:

```
SELECT contract_num, contract_date FROM k_contract
     WHERE contract_num IN
          (SELECT k_contract_contract_num FROM k_bill
          WHERE bill_date
          BETWEEN '2011-12-01' AND '2011-12-31')
```



	contract_num	contract_date
▶	1	2011-11-01
	2	2011-10-01
	4	2011-11-15
	5	2011-08-01

Тот же самый запрос с использованием ключевого слова ANY:

```
SELECT contract_num, contract_date FROM k_contract
     WHERE contract_num =ANY
          (SELECT k_contract_contract_num FROM k_bill
          WHERE bill_date
          BETWEEN '2011-12-01' AND '2011-12-31')
```

Тот же самый запрос можно выполнить и с помощью **связанного** подзапроса, т.е., подзапроса, в котором есть ссылка на главный запрос. Для ссылки на таблицу главного запроса нужно указать псевдоним. Такой подзапрос будет выполняться **заново** для **каждой** строки главного запроса.

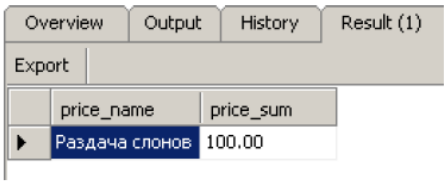
Кроме того, в данном примере иллюстрируется использование ключевого слова EXISTS:

```
SELECT contract_num, contract_date FROM k_contract c
     WHERE EXISTS
          (SELECT * FROM k_bill b
               WHERE bill_date
          BETWEEN '2011-12-01' AND '2011-12-31'
          AND c.contract_num=b.k_contract_contract_num)
```

Пример использования ключевого слова ALL.

Напечатать информацию о товаре (товарах) с наименьшей ценой.

```
SELECT price_name, price_sum FROM k_price
     WHERE price_sum <= ALL
          (SELECT price_sum FROM k_price)
```

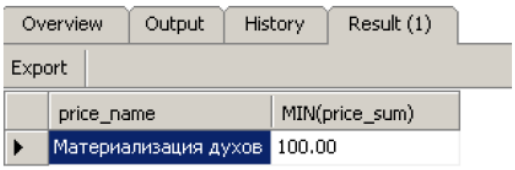


Этот запрос можно сформулировать и по-другому. В этом примере мы можем использовать операцию сравнения =, т.к. подзапрос возвращает ровно одну строку и один столбец.

```
SELECT price_name, price_sum FROM k_price
     WHERE price_sum =
          (SELECT MIN(price_sum) FROM k_price)
```

А так, как в следующем примере, запрос формулировать нельзя. При запуске ошибок **не будет**, просто получится неверный результат:

```
SELECT price_name, MIN(price_sum) FROM k_price
```

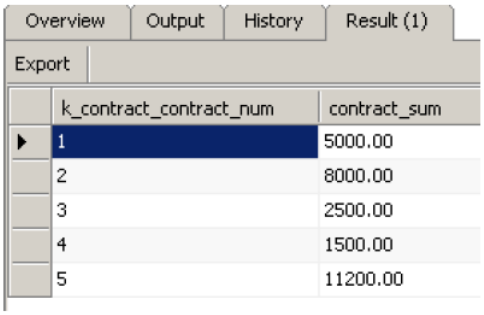


Как видите, значение столбца price_name просто было взято из первой строки таблицы.

Группировка

Для подведения итога по группе данных используется комбинация ключевого слова GROUP BY и агрегирующих функций. Причем в списке полей для выборки обычно присутствуют *только* поля группировки и агрегирующие функции. При необходимости можно добавить дополнительные поля, которые функционально зависят от «ключа группировки».

Получить список договоров и общую сумму счетов по каждому договору:

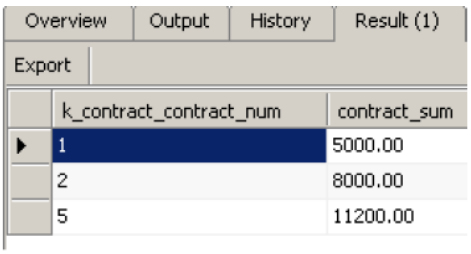



```
SELECT contract_num,
SUM(bill_sum) AS contract_sum
    FROM k_bill
GROUP BY contract_num
```

В том случае, когда нужно выбрать не все группы, а только некоторые из них, используется ключевое слово HAVING:

Получить список договоров, имеющих 2 или более счетов, и общую сумму счетов по каждому договору:

```
SELECT k_contract_contract_num, SUM(bill_sum) AS contract_sum
FROM k_bill
     GROUP BY k_contract_contract_num
HAVING COUNT(bill_num)>=2;
```



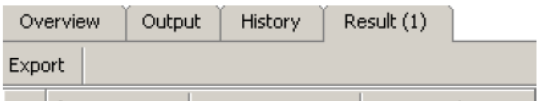
	k_contract_contract_num	contract_sum
▶	1	5000.00
	2	8000.00
	5	11200.00

Выборка из нескольких таблиц

Для связи таблиц можно использовать то же ключевое слово WHERE, как и для условий отбора. При выборке из нескольких таблиц рекомендуется всегда использовать псевдонимы таблиц. Дело в том, что если в разных таблицах имеются одинаковые поля, то всегда нужно уточнять, к какой таблице они относятся, т.е., использовать синтаксис имя_таблицы.имя_поля. А так как имена таблиц обычно длинные, удобно заменять их псевдонимами.

Напечатать список договоров с указанием названия предприятия.

```
SELECT firm_name, contract_num, contract_date
     FROM k_firm f, k_contract c
     WHERE f.firm_num=c.k_firm_firm_num
```



	firm_name	contract_num	contract_date
▶	Альфа	1	2011-11-01
	Альфа	2	2011-10-01
	Альфа	3	2011-09-01
	Бета	4	2011-11-15
	Бета	5	2011-08-01
	Гамма	6	2011-07-15
	Дельта	7	2011-11-12

То же самое можно получить, если использовать синтаксис JOIN...ON. Это так называемое *внутреннее* (INNER) соединение. Строки соединяются, если совпадают значения полей в условии ON.

```
SELECT firm_name, contract_num, contract_date
FROM k_firm f JOIN k_contract c ON f.firm_num=c.k_firm_firm_num
```

Для соединения трех и более таблиц синтаксис в этом формате следующий:

Напечатать список сотрудников, номера и даты договоров, которые они заключили, с указанием названия предприятия.

```
SELECT staff_name, contract_num, contract_date, firm_name
FROM k_firm f JOIN k_contract c ON f.firm_num=c.k_firm_firm_num
              JOIN k_staff s ON  s.staff_num=c.k_staff_staff_num
```

Overview | Output | History | Result (1)

Export

	staff_name	contract_num	contract_date	firm_name
▶	Иванов	1	2011-11-01	Альфа
	Иванов	3	2011-09-01	Альфа
	Иванов	6	2011-07-15	Гамма
	Иванов	7	2011-11-12	Дельта
	Петров	2	2011-10-01	Альфа
	Петров	4	2011-11-15	Бета
	Петров	5	2011-08-01	Бета

Кроме внутреннего, бывают еще левое (LEFT), правое (RIGHT) и полное (FULL) соединения.

Рассмотрим, например, левое соединение. В результат попадут строки, в которых совпадают значения полей в условии ON, и те строки из левой таблицы, для которых не нашлось соответствующих строк в правой таблице. Поля из правой таблицы будут заполнены значениями NULL.

Напечатать список договоров с указанием названия предприятия плюс список предприятий, у которых нет договоров.

Overview | Output | History | Result (1)

Export

	firm_name	contract_num	contract_date
▶	Альфа	1	2011-11-01
	Альфа	2	2011-10-01
	Альфа	3	2011-09-01
	Бета	4	2011-11-15
	Бета	5	2011-08-01
	Гамма	6	2011-07-15
	Дельта	7	2011-11-12
	Эпсилон	NULL	NULL

```
SELECT firm_name,
       contract_num,
       contract_date
FROM k_firm f
LEFT JOIN k_contract c
ON f.firm_num=c.k_firm_firm_num
```

А что будет в том случае, если условие связи вообще не указывать? Получится так называемое *декартово произведение* таблиц, в котором *каждая* строка первой таблицы будет сцеплена с *каждой* строкой второй таблицы. Результат получается обычно очень большим и не имеющим смысла.

```
SELECT firm_name, contract_num, contract_date
     FROM k_firm f, k_contract c

# Предыдущий запрос вернул 35 строк,
# т.е. 5 предприятий умножить на 7 договоров.
```

Разумеется, в одном и том же запросе можно связывать не только две, а три и более таблицы, использовать в этих запросах подзапросы, группировки и т.п. Например, запрос к 4 таблицам:

Напечатать информацию о платежах с указанием названия предприятия.

```
SELECT firm_name, payment_date, payment_sum
 FROM k_firm f, k_contract c, k_bill b, k_payment p
 WHERE f.firm_num=c.k_firm_firm_num  AND
       c.contract_num=b.k_contract_contract_num AND
       b.bill_num=p.k_bill_bill_num
```

Overview | Output | History | Result (1)

Export

firm_name	payment_date	payment_sum
Альфа	2011-12-15	1000.00
Альфа	2012-01-13	1500.00
Альфа	2012-01-15	500.00
Альфа	2012-01-12	1000.00
Бета	2012-01-05	100.00
Бета	2012-01-12	900.00
Бета	2011-12-25	1000.00

Объединение запросов

Для объединения результатов двух и более запросов нужно использовать ключевое слово UNION. Объединяемые запросы должны иметь **одинаковое** количество и тип полей. Параметр ORDER BY , если он нужен, следует указывать только в **последнем** запросе.

Получить список договоров и общую сумму счетов по каждому договору, а также строку с итоговой суммой.

```
SELECT CONCAT('Договор № ',
              CONVERT(k_contract_contract_num, CHAR),
              ' на сумму ') AS "Номер",
       SUM(bill_sum) AS "Сумма" FROM k_bill
GROUP BY k_contract_contract_num
UNION
SELECT 'ИТОГО: ', SUM(bill_sum) FROM k_bill ORDER BY 1
```

Overview | Output | History | Result (1)

Export

Номер	Сумма
Договор № 1 на сумму	5000.00
Договор № 2 на сумму	8000.00
Договор № 3 на сумму	2500.00
Договор № 4 на сумму	1500.00
Договор № 5 на сумму	11200.00
ИТОГО:	28200.00

И еще несколько примеров

Получить прайс-лист с суммой заказов по каждому товару. Обратите внимание, что название и цена товара могут использоваться в списке полей для выбора, поскольку они функционально (однозначно) зависят от номера товара, по которому проводится группировка.

```
SELECT pr.price_name, pr.price_sum,
       SUM(prot.kolvo*prot.price_sum)
FROM k_price pr, k_protokol prot
WHERE pr.price_num=prot.k_price_price_num
GROUP BY pr.price_num ORDER BY 1
```

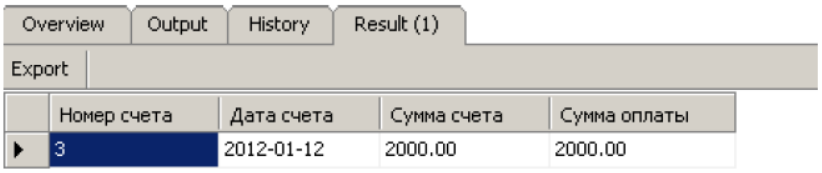
Overview | Output | History | Result (1)

Export

price_name	price_sum	SUM(prot.kolvo*prot.price_sum)
Материализация духов	1000.00	7000.00
Раздача слонов	100.00	3700.00
Слоновий бивень	3000.00	6000.00
Моржовый клык	1500.00	1500.00
Копыто Пегаса	5000.00	10000.00

Полностью оплаченные счета, т.е., счета, сумма платежей по которым больше или равна сумме счета. Обратите внимание на применение подзапроса.

```
SELECT    b.bill_num AS "Номер счета",
     b.bill_date AS "Дата счета",
     b.bill_sum AS "Сумма счета",
     SUM(p.payment_sum) AS "Сумма оплаты"
FROM k_bill b, k_payment p
WHERE b.bill_num=p.k_bill_bill_num AND
     b.bill_sum<=
   (SELECT SUM(payment_sum) FROM k_payment p2
    WHERE b.bill_num=p2.k_bill_bill_num)
GROUP BY b.bill_num
```



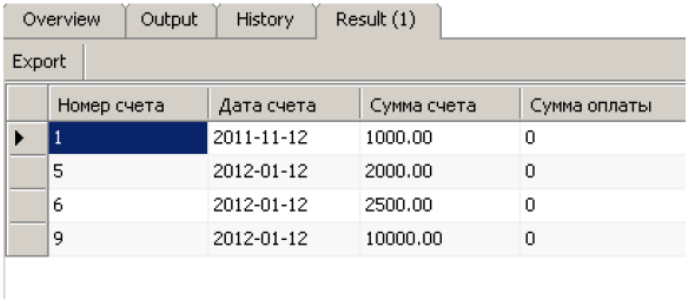
Overview | Output | History | Result (1)

Export

Номер счета	Дата счета	Сумма счета	Сумма оплаты
3	2012-01-12	2000.00	2000.00

Полностью неоплаченные счета, по которым вообще нет платежей.

```
SELECT    b.bill_num AS "Номер счета",
     b.bill_date AS "Дата счета",
     b.bill_sum AS "Сумма счета",
     0 AS "Сумма оплаты"
FROM k_bill b
WHERE b.bill_num NOT IN (SELECT k_bill_bill_num FROM k_payment)
```



Overview | Output | History | Result (1)

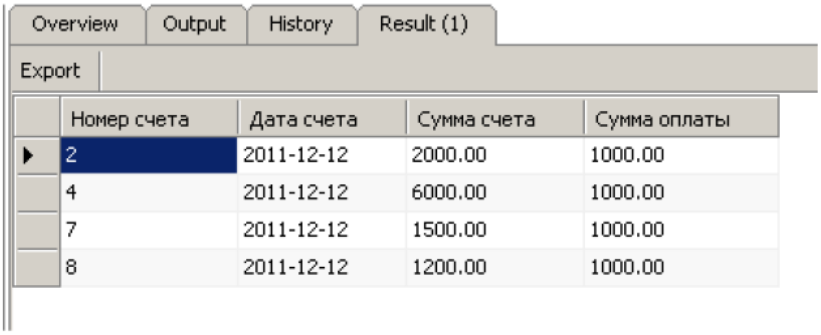
Export

Номер счета	Дата счета	Сумма счета	Сумма оплаты
1	2011-11-12	1000.00	0
5	2012-01-12	2000.00	0
6	2012-01-12	2500.00	0
9	2012-01-12	10000.00	0

Частично оплаченные счета. Обратите внимание, что в этом примере в параметре FROM вместо второй таблицы используется вложенный SELECT

```
SELECT    b.bill_num AS "Номер счета",
     b.bill_date AS "Дата счета",
     b.bill_sum AS "Сумма счета",
```

```
     p.pay_sum AS "Сумма оплаты"
FROM k_bill b,
(SELECT k_bill_bill_num, SUM(payment_sum) as pay_sum
FROM k_payment
GROUP BY k_bill_bill_num) p
WHERE b.bill_sum >p.pay_sum AND b.bill_num=p.k_bill_bill_num
```



Номер счета	Дата счета	Сумма счета	Сумма оплаты
2	2011-12-12	2000.00	1000.00
4	2011-12-12	6000.00	1000.00
7	2011-12-12	1500.00	1000.00
8	2011-12-12	1200.00	1000.00