ПОДЗАПРОСЫ В SQL

Вложенные подзапросы

SQL позволяет использовать одни запросы внутри других запросов, то есть вкладывать запросы друг в друга.

Как работает запрос SQL со связанным подзапросом?

- Выбирается строка из таблицы, имя которой указано во внешнем запросе.
- Выполняется подзапрос и полученное значение применяется для анализа этой строки в условии предложения WHERE внешнего запроса.
- По результату оценки этого условия принимается решение о включении или не включении строки в состав выходных данных.
- Процедура повторяется для следующей строки таблицы внешнего запроса.

Правила организации подзапросов

1. Подзапросы должны быть заключены в круглые скобки.
2. Команда ORDER BY не может использоваться в подзапросе, хотя в основном запросе она использоваться может.
3. В подзапросе может использоваться команда GROUP BY для выполнения той же функции, что и ORDER BY.
4. Подзапросы, которые возвращают более одной строки, могут использоваться только с несколькими операторами значений, такими как оператор IN.

Оператор IN

1. Используется для сравнения проверяемого значения поля с заданным списком. Этот список значений указывается в скобках справа от оператора IN.
2. Построенное с использованием IN условие считается истинным, если значение поля, имя которого указано слева от IN, совпадает с одним из значений, перечисленных в списке, указанном в скобках справа от IN.

Вложенные подзапросы

Пример:

Извлечь данные обо всех оценках студента Петрова.

```
SELECT *
FROM EXAM_MARKS
WHERE STUDENT_ID =
(SELECT  STUDENT_ID
FROM  STUDENT WHERE SURNAME= ‘Петров’);
```

Вложенные подзапросы

Результат выполнения запроса:

Запрос1

Exam_id	Student_id	Subj_id	Mark	Exam_date
43	Петров	22	3	18.01.2000
0		0	0	

Вышеуказанный запрос действителен только в том случае, если в результате выполнения подзапроса, указанного в скобках, возвращается одно значение.

Если в результате подзапроса возвращается более одного значения, применяется оператор IN.

Вложенные подзапросы

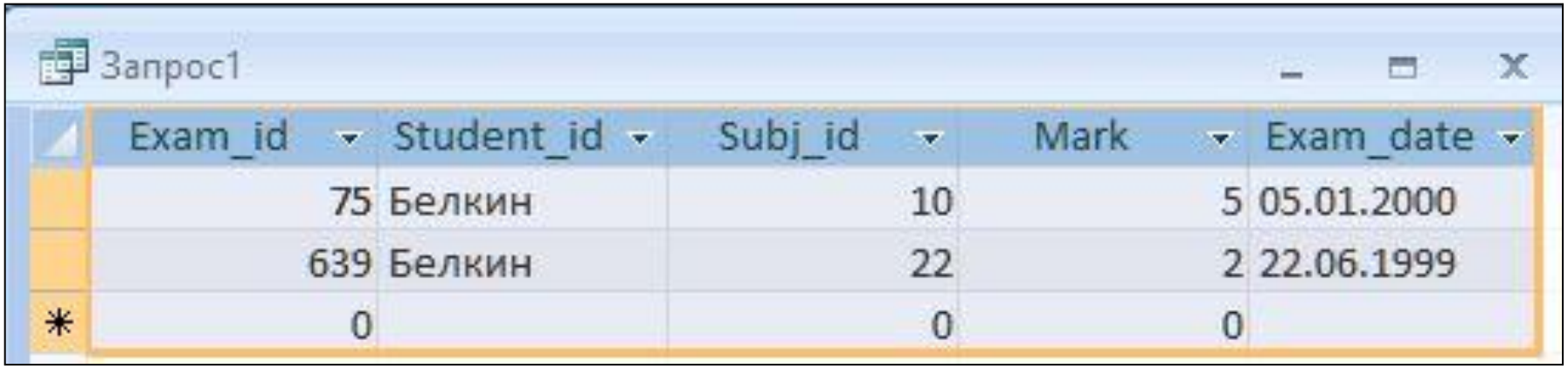
Пример:

Выбрать данные обо всех оценках студентов из Воронежа.

```
SELECT *
FROM EXAM_MARKS

WHERE STUDENT_ID IN

(SELECT  STUDENT_ID

FROM  STUDENT

WHERE CITY = 'Воронеж');
```

Вложенные подзапросы

Результат выполнения запроса:



Exam_id	Student_id	Subj_id	Mark	Exam_date
75	Белкин	10	5	05.01.2000
639	Белкин	22	2	22.06.1999
0		0	0	

Использование алиасов вместо имен таблиц

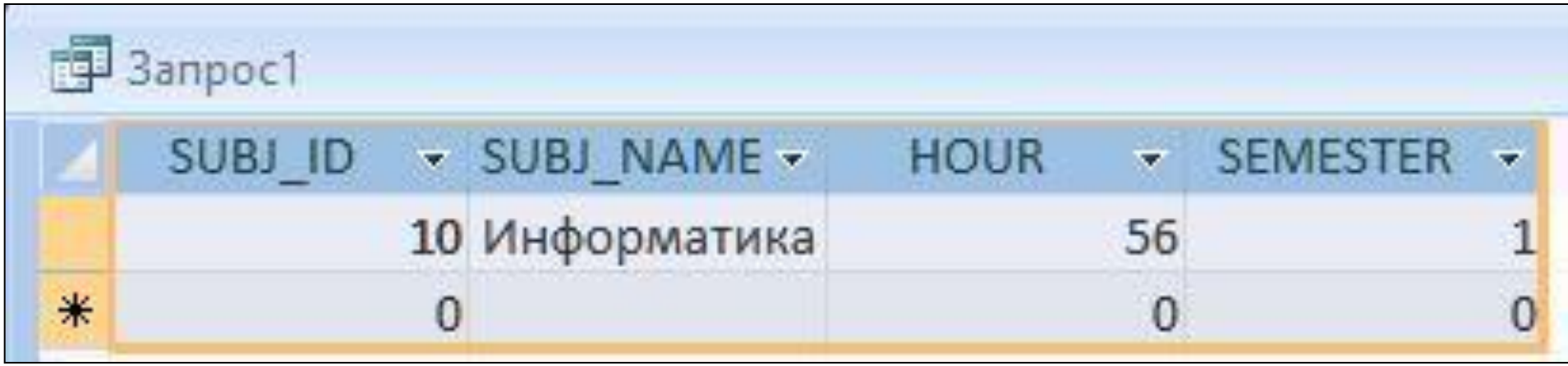
Пример:

Выбрать сведения обо всех предметах обучения, по которым проводился экзамен 12 января 2000 г.

```
 SELECT *
FROM SUBJECT AS SU

WHERE '12.01.2000' IN

( SELECT  EXAM_DATE

FROM  EXAM_MARKS AS EX

WHERE SU.SUBJ_ID = EX.SUBJ_ID );
```

Использование алиасов вместо имен таблиц

Результат выполнения запроса:



Запрос1

SUBJ_ID	SUBJ_NAME	HOUR	SEMESTER
10	Информатика	56	1
0		0	0

В некоторых СУБД для выполнения этого запроса может потребоваться преобразование значения даты в символьный тип.

Формирование связанных подзапросов

При использовании подзапросов во внутреннем запросе можно ссылаться на таблицу, имя которой указано в предложении FROM внешнего запроса. В этом случае связанный подзапрос выполняется один раз для каждой строки таблицы основного запроса.

Формирование связанных подзапросов

Исходя из того, что предложение GROUP BY позволяет группировать выводимые SELECT- запросом записи по некоторому полю, то предложение HAVING позволяет осуществлять при выводе фильтрацию этих групп.

Предикат предложения HAVING оценивается не для каждой строки результата, а для группы выходных записей, сформированной предложением GROUP BY внешнего запроса.

Формирование связанных подзапросов

Пример:

Необходимо по данным из таблицы EXAM_MARKS определить сумму полученных студентами оценок, сгруппировав значения оценок по датам экзаменов и включив те дни, когда число студентов, сдававших в течение дня экзамены, было равно 1.

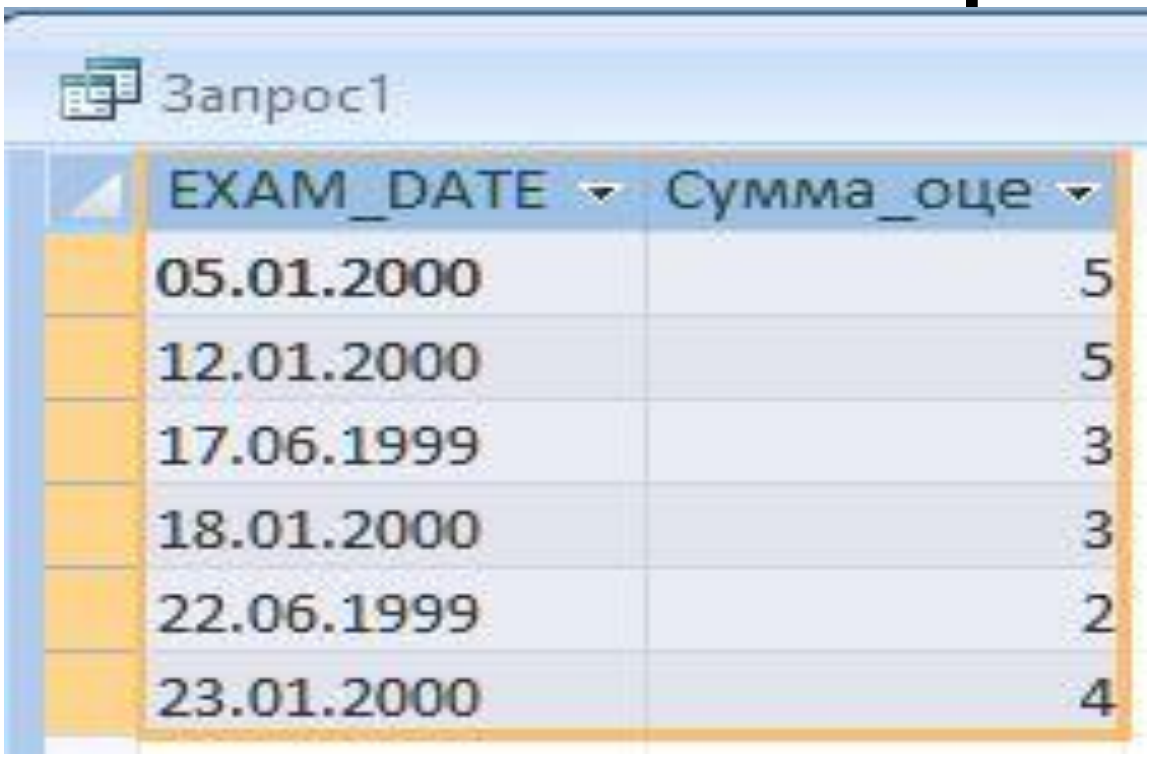
Формирование связанных подзапросов

Пример:

```
SELECT EXAM_DATE, SUM(MARK) AS
Сумма_оценок
FROM  EXAM_MARKS AS A
GROUP BY EXAM_DATE
HAVING 1=
(SELECT COUNT (MARK)
FROM EXAM_MARKS AS B
WHERE A. EXAM_DATE = B. EXAM_DATE);
```

Формирование связанных подзапросов

Результат выполнения запроса:


Запрос1

EXAM_DATE	Сумма_оце
05.01.2000	5
12.01.2000	5
17.06.1999	3
18.01.2000	3
22.06.1999	2
23.01.2000	4

Подзапрос вычисляет количество строк с одной и той же датой, совпадающей с датой, для которой сформирована очередная группа основного запроса.

Оператор EXISTS

Оператор EXISTS генерирует значение истина или ложь.

Используя подзапросы в качестве аргумента, этот оператор оценивает результат выполнения подзапроса как true, если этот подзапрос генерирует выходные данные.

В противном случае результат подзапроса будет false.

Оператор EXISTS не может принимать значение UNKNOWN(неизвестно).

Оператор EXISTS

При использовании связанных вложенных запросов предложение EXISTS анализирует каждую строку таблицы, на которую ссылается внешний запрос. Основной запрос извлекает строки из кандидатов, проверяющих условия. Для каждой строки-кандидата выполняется подзапрос.

- ***Нельзя использовать функции агрегирования в подзапросе, указанном в инструкции EXISTS.***

Оператор EXISTS

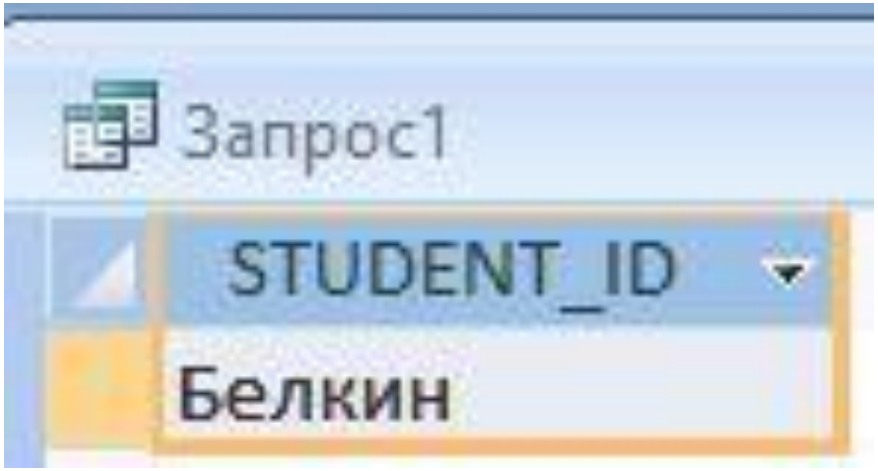
Пример :

Извлечь из таблицы EXAM_MARKS данные о студентах, получивших хотя бы одну неудовлетворительную оценку.

```
 SELECT DISTINCT  STUDENT_ID
FROM EXAM_MARKS AS A
WHERE  EXISTS
(SELECT *
FROM EXAM_MARKS AS B
WHERE MARK < 3
AND B. STUDENT_ID =A. STUDENT_ID);
```

Оператор EXISTS

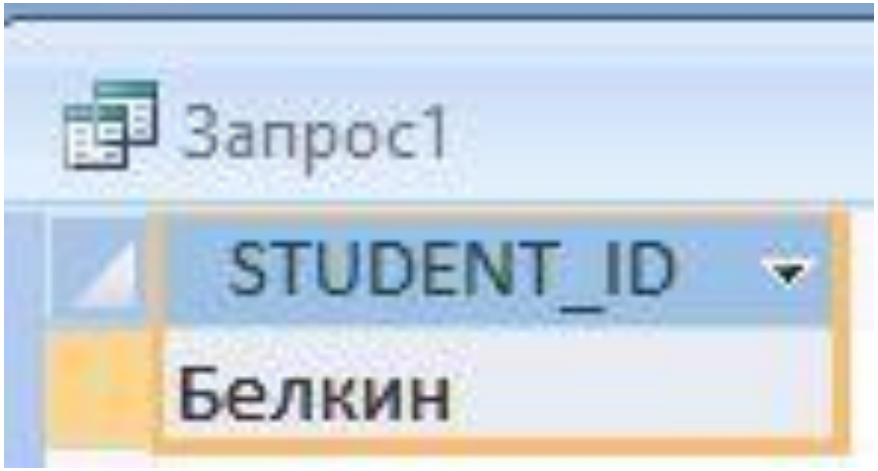
Результат выполнения запроса:



Как только подзапрос находит строку, в которой значение столбца MARK удовлетворяет условию, он останавливает выполнение и возвращает true внешнему запросу, который затем анализирует его строку-кандидата.

Оператор EXISTS

Результат выполнения запроса:



Как только подзапрос находит строку, в которой значение столбца MARK удовлетворяет условию, он останавливает выполнение и возвращает true внешнему запросу, который затем анализирует его строку-кандидата.

Операторы сравнения с множеством значений IN, ANY,ALL

Операторы сравнения

Оператор ALL, как правило, эффективно используется с неравенствами. В SQL выражение <> ALL реально означает *не равно ни одному* из результатов подзапроса.

Операторы сравнения

IN	Равно любому из значений, полученных в подзапросе.
NOT IN	Не равно ни одному из значений, полученных в подзапросе.
= ANY	Равно любому из значений, полученных в подзапросе. Соответствует логическому оператору OR.
>ANY, >=ANY	Больше, чем (больше или равно) любое полученное число.
<ANY, <=ANY	Меньше, чем (меньше или равно) любое полученное число.

Операторы сравнения

= ALL	Равно всем полученным значениям. Соответствует логическому оператору AND.
> ALL, >=ALL	Больше, чем (больше или равно) все полученные числа.
<ALL, <=ALL	Меньше, чем (меньше или равно) все полученные числа.

Операторы сравнения

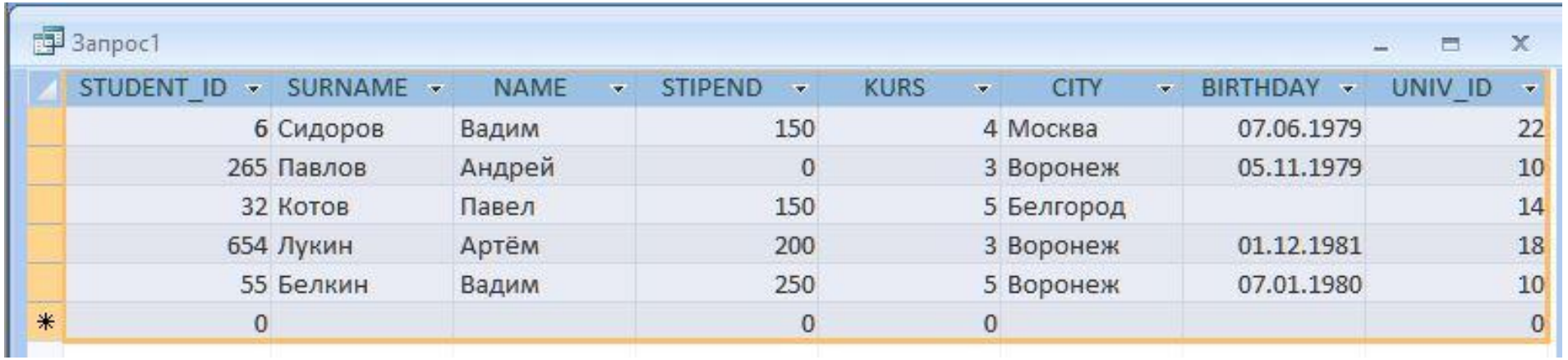
Пример 1:

Выбрать сведения о студентах, проживающих в городе, где расположен университет, в котором они учатся.

```
SELECT *
FROM STUDENT AS S
WHERE  CITY=ANY
(SELECT CITY
FROM UNIVERSITY AS U
WHERE U.UNIV_ID=S.UNIV_ID);
```

Операторы сравнения

Результат выполнения запроса:


Запрос1

STUDENT_ID	SURNAME	NAME	STIPEND	KURS	CITY	BIRTHDAY	UNIV_ID
6	Сидоров	Вадим	150	4	Москва	07.06.1979	22
265	Павлов	Андрей	0	3	Воронеж	05.11.1979	10
32	Котов	Павел	150	5	Белгород		14
654	Лукин	Артём	200	3	Воронеж	01.12.1981	18
55	Белкин	Вадим	250	5	Воронеж	07.01.1980	10
0			0	0			0

Операторы сравнения

Пример 2:

Выбрать данные о названиях всех университетов с рейтингом более высоким, чем рейтинг любого университета Воронежа:

```
 SELECT *
FROM UNIVERSITY
WHERE RATING>ALL
(SELECT RATING
FROM UNIVERSITY
WHERE CITY= 'Воронеж');
```

Операторы сравнения

Результат выполнения запроса:

Запрос1

UNIV_ID	UNIV_NAME	RATING	CITY
22	МГУ	606	Москва
11	НГУ	345	Новосибирск
32	РГУ	416	Ростов
15	ТГУ	368	Томск
0		0	