

# Область использования вложенных запросов

Ряд конструкций языка DML предусматривает использование вложенных запросов:

- некоторые формы конструкции WHERE в операторе SELECT;
- теоретико-множественные операции (например, UNION);
- вставка данных в таблицу (INSERT)
- любая таблица в запросе может быть представлена в виде вложенного запроса.

# Формы с подзапросами в конструкции **WHERE** оператора **SELECT**

- expression [ NOT ] IN (SELECT-statement)
- expression relation-operator ( SELECT-statement )
- [NOT ] EXISTS ( SELECT-statement )
- expression relation-operator { ANY | ALL | SOME } ( SELECT-statement )
- (здесь ключевые слова ANY и SOME — синонимы).

# Оператор UNION

SELECT-statement

UNION [ALL]

SELECT-statement

[UNION SELECT-statement...]

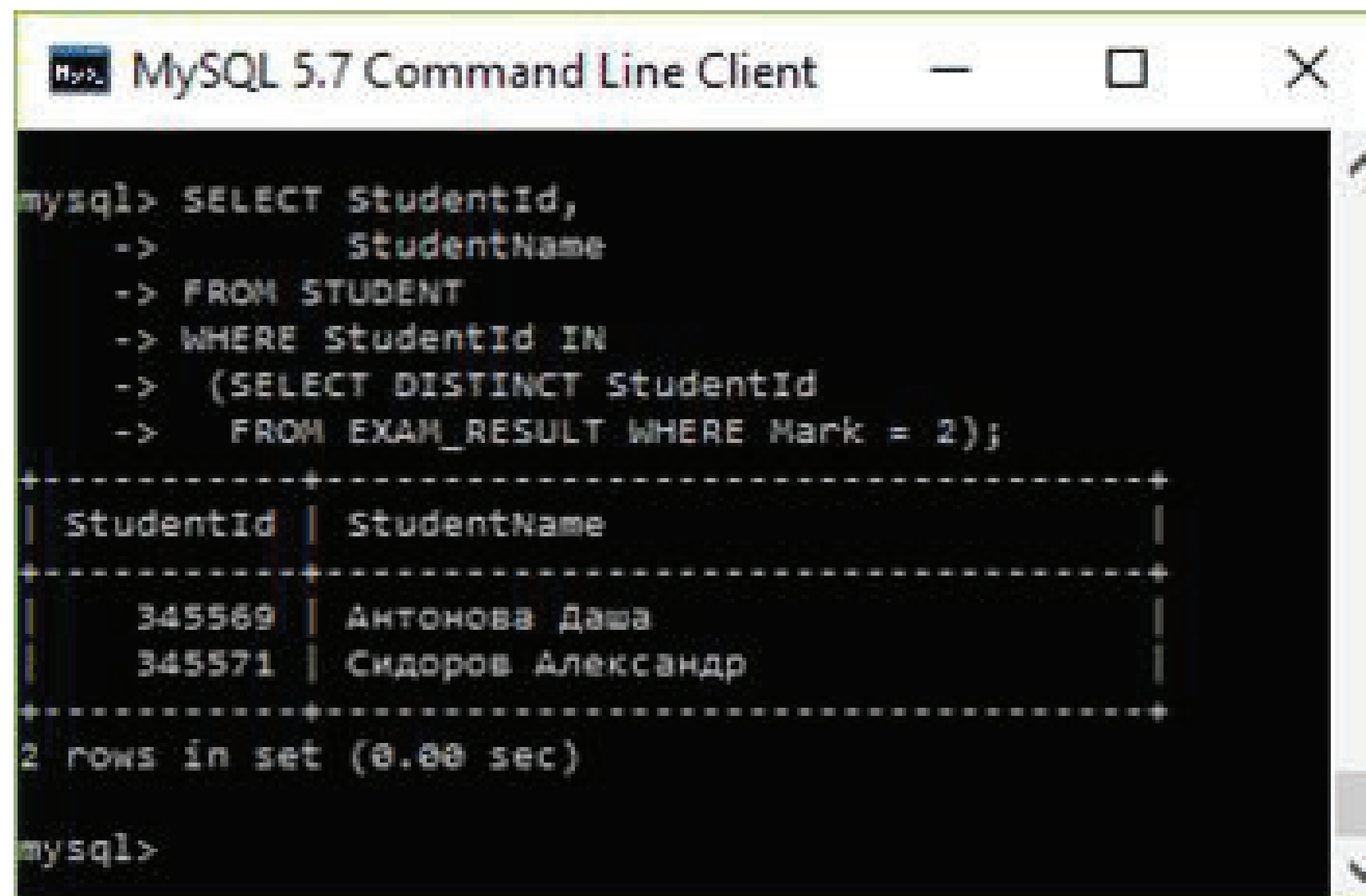
[ORDER BY ...]

[LIMIT n[,m]]

# Пример: запрос с проверкой на вхождение в множество

Номера зачетов и ФИО студентов, получивших 2 на экзаменах.

```
SELECT StudentId,  
       StudentName  
FROM STUDENT  
WHERE StudentId IN  
      (SELECT DISTINCT StudentId  
       FROM EXAM_RESULT  
       WHERE Mark = 2);
```



```
mysql> SELECT StudentId,  
->      StudentName  
-> FROM STUDENT  
-> WHERE StudentId IN  
->      (SELECT DISTINCT StudentId  
->      FROM EXAM_RESULT WHERE Mark = 2);
```

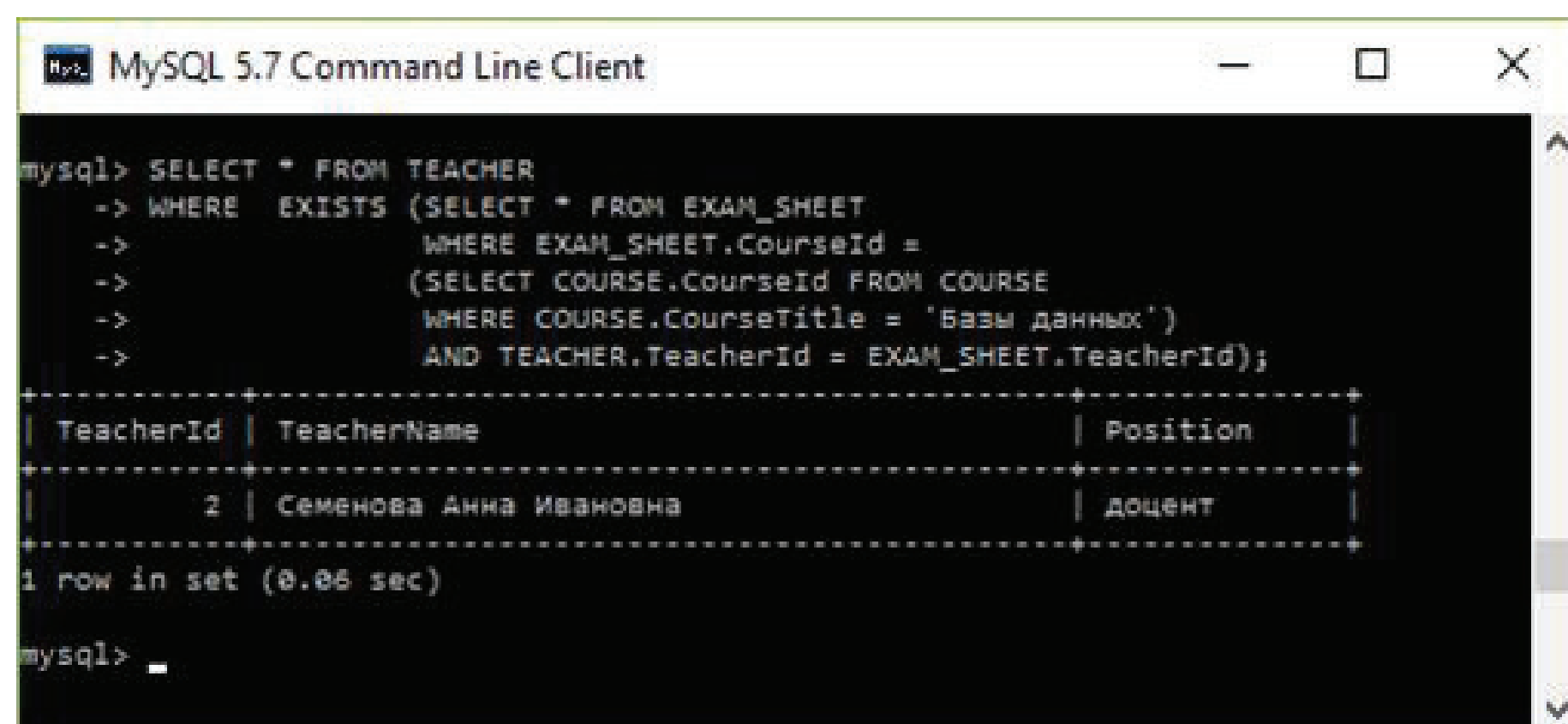
StudentId	StudentName
345569	Антонова Даша
345571	Сидоров Александр

```
2 rows in set (0.00 sec)  
  
mysql>
```

# Пример: запрос с проверкой существования записей подзапроса

Преподаватели, которые будут принимать экзамен по Базам данных.

```
SELECT * FROM TEACHER
WHERE EXISTS (SELECT * FROM EXAM_SHEET WHERE EXAM_SHEET.CourseId =
              (SELECT COURSE.CourseId FROM COURSE
               WHERE COURSE.CourseTitle = 'Базы данных')
              AND TEACHER.TeacherId = EXAM_SHEET.TeacherId);
```



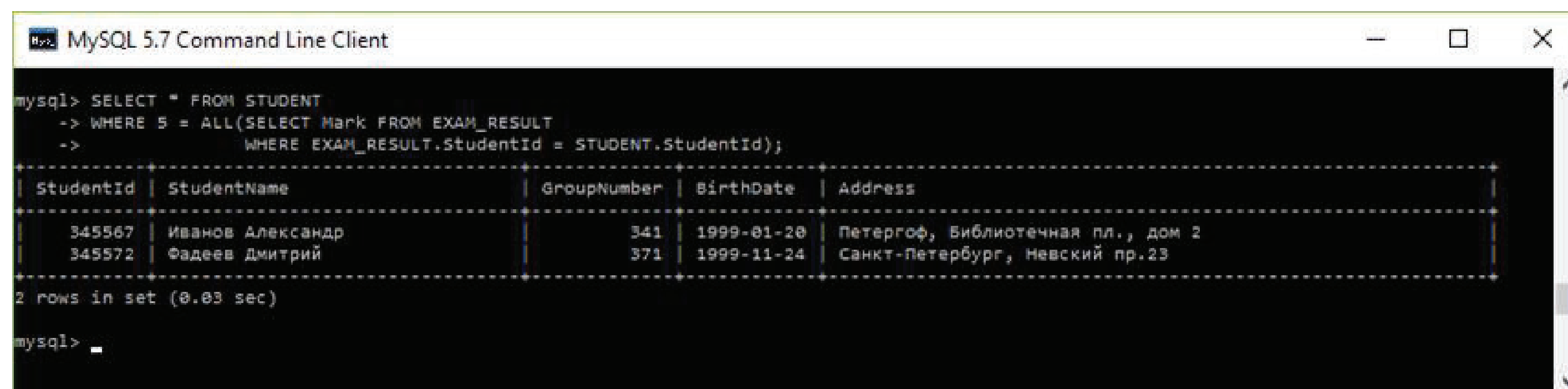
```
mysql> SELECT * FROM TEACHER
-> WHERE EXISTS (SELECT * FROM EXAM_SHEET
->                WHERE EXAM_SHEET.CourseId =
->                (SELECT COURSE.CourseId FROM COURSE
->                WHERE COURSE.CourseTitle = 'Базы данных')
->                AND TEACHER.TeacherId = EXAM_SHEET.TeacherId);
+-----+-----+-----+
| TeacherId | TeacherName          | Position |
+-----+-----+-----+
| 2         | Семенова Анна Ивановна | доцент  |
+-----+-----+-----+
1 row in set (0.06 sec)

mysql> _
```

# Пример: запрос с проверкой значения всех записей подзапроса

Вся информация о студентах-отличниках.

```
SELECT * FROM STUDENT
WHERE 5 = ALL(SELECT Mark FROM EXAM_RESULT
              WHERE EXAM_RESULT.StudentId = STUDENT.StudentId);
```



```
mysql> SELECT * FROM STUDENT
-> WHERE 5 = ALL(SELECT Mark FROM EXAM_RESULT
->                WHERE EXAM_RESULT.StudentId = STUDENT.StudentId);
```

StudentId	StudentName	GroupNumber	BirthDate	Address
345567	Иванов Александр	341	1999-01-20	Петергоф, Библиотечная пл., дом 2
345572	Фадеев Дмитрий	371	1999-11-24	Санкт-Петербург, Невский пр.23

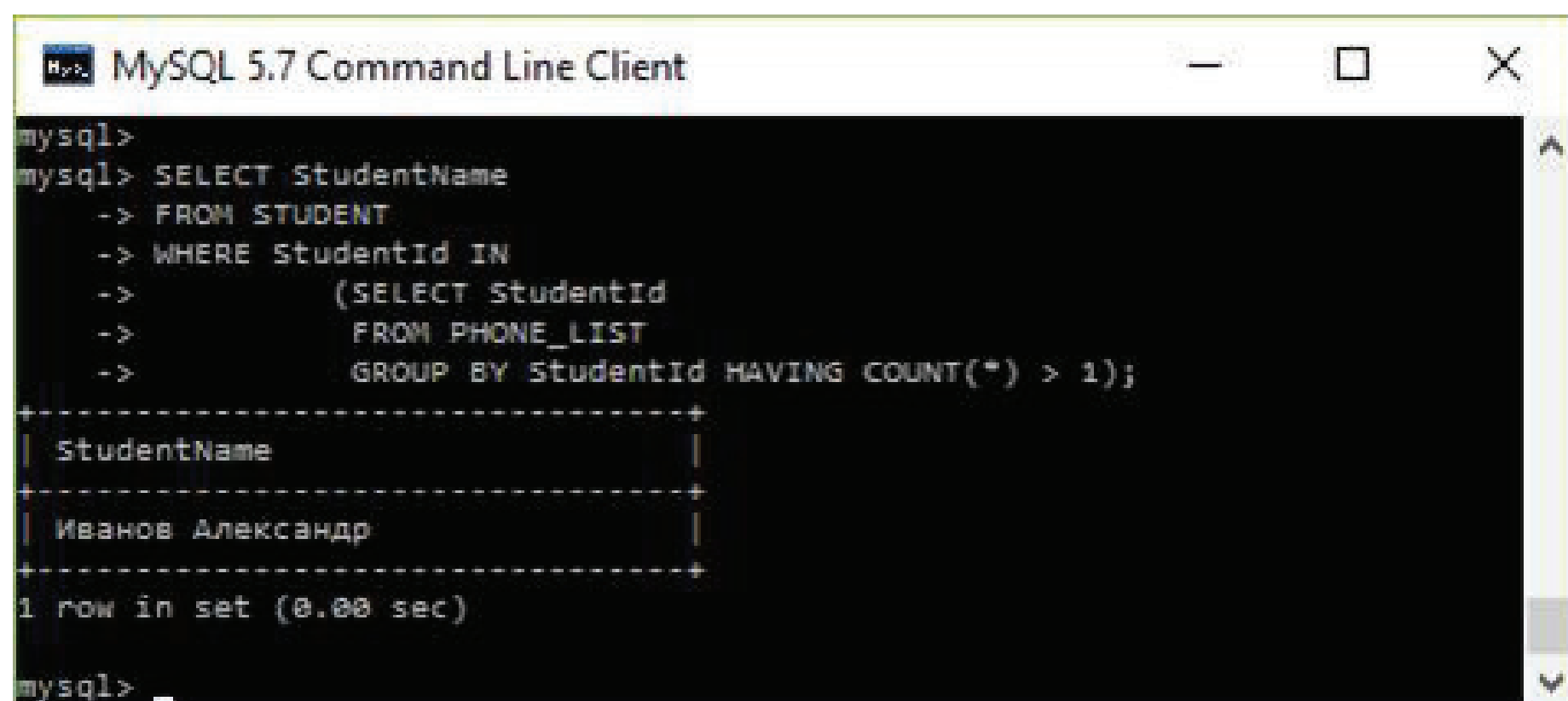
```
2 rows in set (0.03 sec)

mysql>
```

# Пример: агрегирование с условием для группировки

Студенты, для которых указано более одного номера телефона.

```
SELECT StudentName FROM STUDENT
WHERE StudentId IN (SELECT StudentId
                    FROM PHONE_LIST
                    GROUP BY StudentId HAVING COUNT(*) > 1);
```



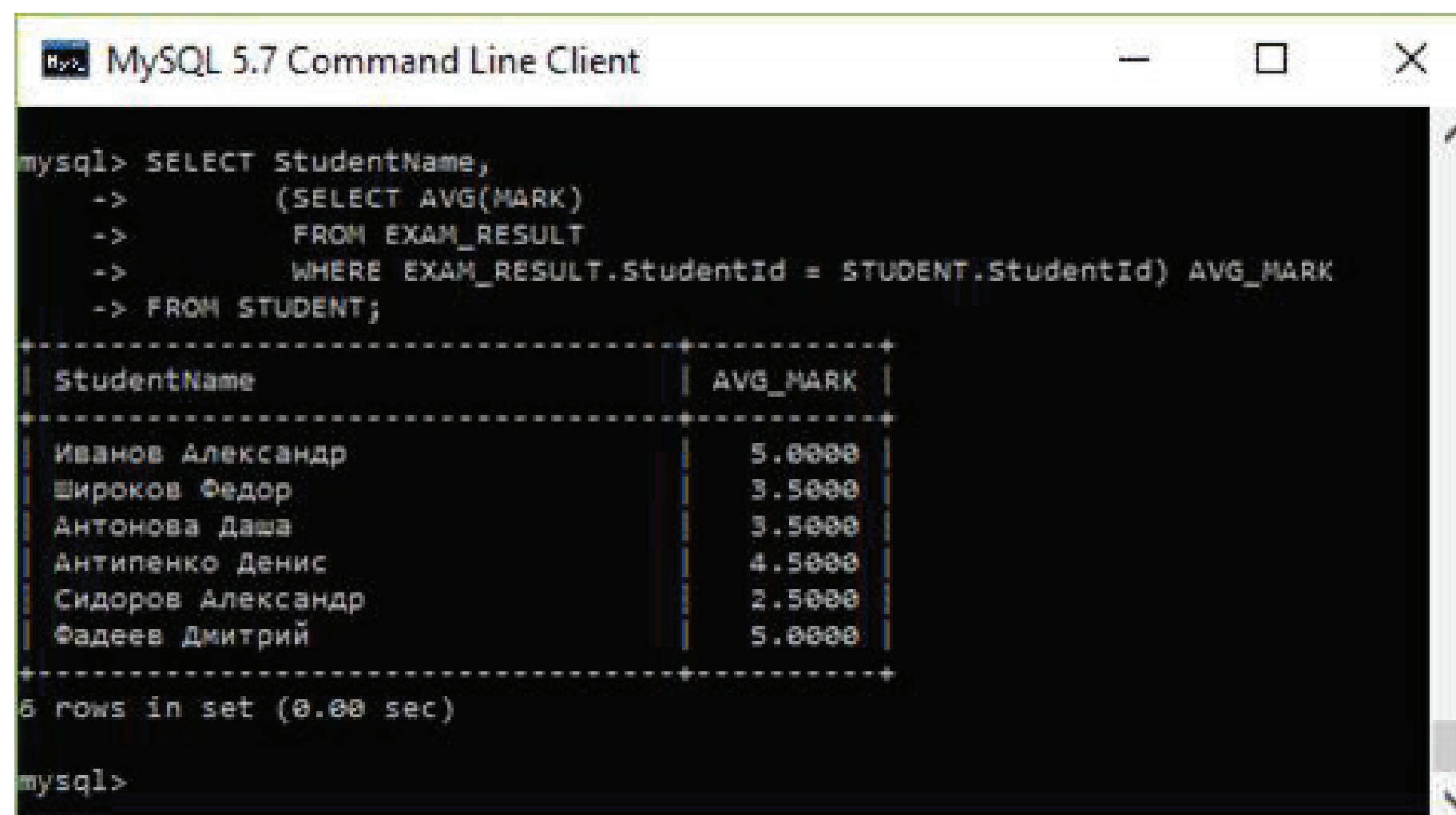
```
mysql>
mysql> SELECT StudentName
-> FROM STUDENT
-> WHERE StudentId IN
-> (SELECT StudentId
-> FROM PHONE_LIST
-> GROUP BY StudentId HAVING COUNT(*) > 1);
+-----+
| StudentName |
+-----+
| Иванов Александр |
+-----+
1 row in set (0.00 sec)

mysql> _
```

# Пример: использование вложенного запроса в списке полей

Вычисление среднего балла студента.

```
SELECT StudentName,  
       (SELECT AVG(MARK) FROM EXAM_RESULT  
        WHERE EXAM_RESULT.StudentId = STUDENT.StudentId) AVG_MARK  
FROM STUDENT;
```

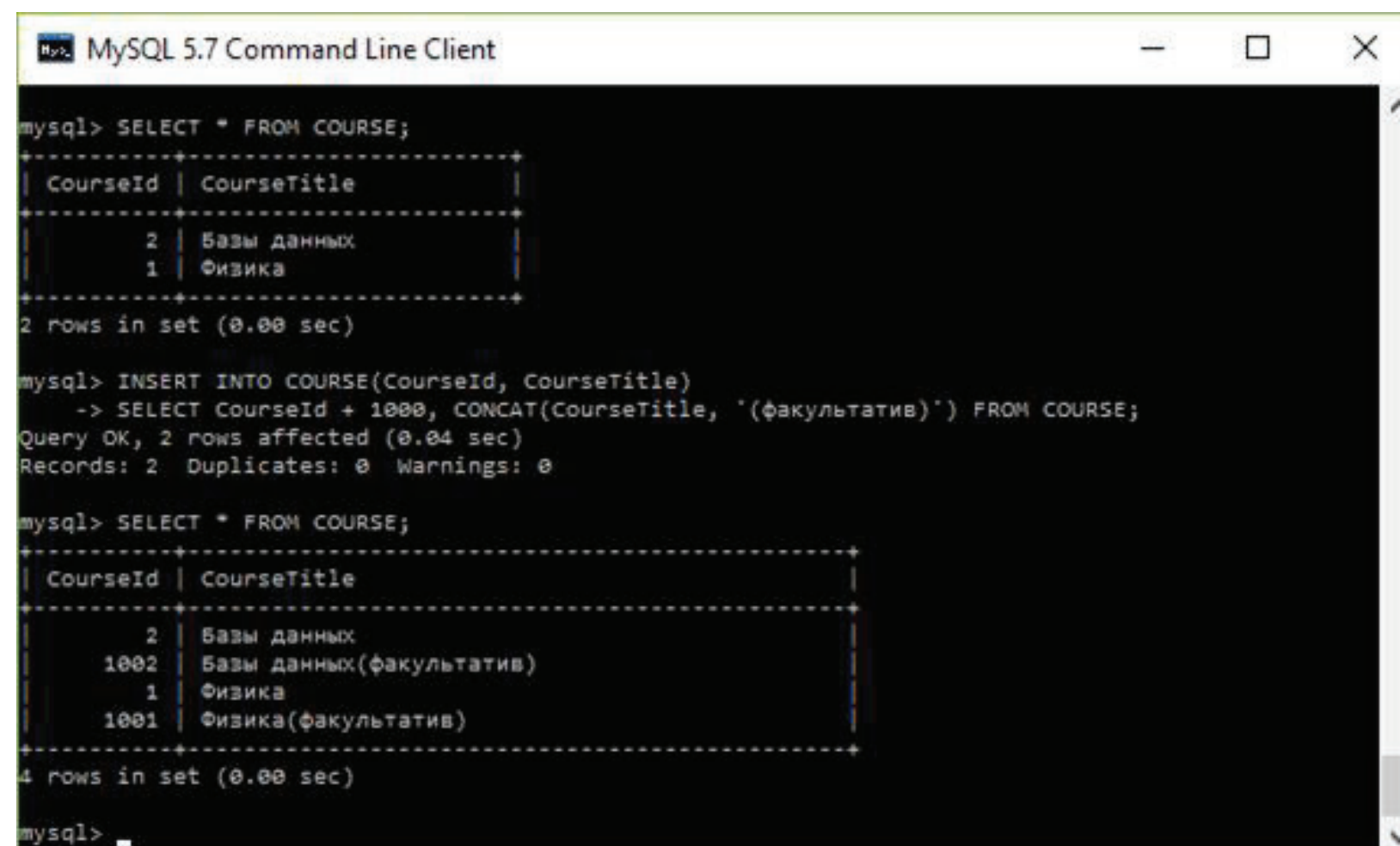


```
mysql> SELECT StudentName,  
->       (SELECT AVG(MARK)  
->       FROM EXAM_RESULT  
->       WHERE EXAM_RESULT.StudentId = STUDENT.StudentId) AVG_MARK  
-> FROM STUDENT;  
+-----+-----+  
| StudentName | AVG_MARK |  
+-----+-----+  
| Иванов Александр | 5.0000 |  
| Широков Федор | 3.5000 |  
| Антонова Даша | 3.5000 |  
| Антипенко Денис | 4.5000 |  
| Сидоров Александр | 2.5000 |  
| Фадеев Дмитрий | 5.0000 |  
+-----+-----+  
6 rows in set (0.00 sec)  
  
mysql>
```



# Пример: вложенный запрос в операторе INSERT

Добавление факультативных курсов в таблицу COURSE.



```
mysql> SELECT * FROM COURSE;
+----+-----+
| CourseId | CourseTitle |
+----+-----+
| 2 | Базы данных |
| 1 | Физика |
+----+-----+
2 rows in set (0.00 sec)

mysql> INSERT INTO COURSE(CourseId, CourseTitle)
-> SELECT CourseId + 1000, CONCAT(CourseTitle, '(факультатив)') FROM COURSE;
Query OK, 2 rows affected (0.04 sec)
Records: 2 Duplicates: 0 Warnings: 0

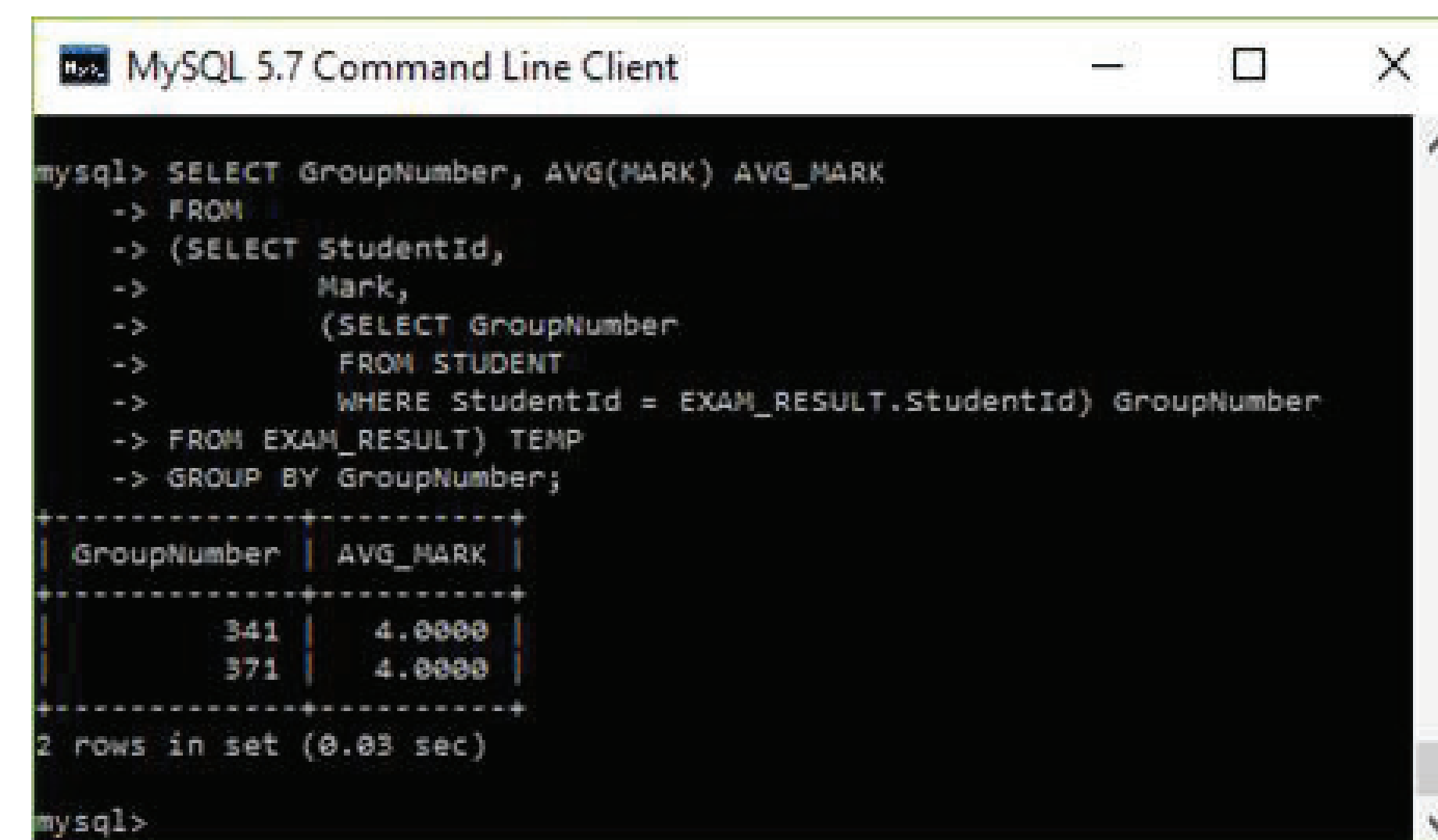
mysql> SELECT * FROM COURSE;
+----+-----+
| CourseId | CourseTitle |
+----+-----+
| 2 | Базы данных |
| 1002 | Базы данных(факультатив) |
| 1 | Физика |
| 1001 | Физика(факультатив) |
+----+-----+
4 rows in set (0.00 sec)

mysql>
```

```
INSERT INTO COURSE(CourseId, CourseTitle)
SELECT CourseId + 1000, CONCAT(CourseTitle,
' (факультатив) ')
FROM COURSE;
```

# Пример: агрегирование среднего балла для групп

```
SELECT GroupNumber, AVG(MARK) AVG_MARK
FROM
(SELECT StudentId, Mark,
        (SELECT GroupNumber FROM STUDENT
         WHERE StudentId = EXAM_RESULT.
StudentId) GroupNumber
FROM EXAM_RESULT) TEMP
GROUP BY GroupNumber;
```



```
mysql> SELECT GroupNumber, AVG(MARK) AVG_MARK
-> FROM
-> (SELECT StudentId,
->      Mark,
->      (SELECT GroupNumber
->      FROM STUDENT
->      WHERE StudentId = EXAM_RESULT.StudentId) GroupNumber
-> FROM EXAM_RESULT) TEMP
-> GROUP BY GroupNumber;
```

GroupNumber	AVG_MARK
341	4.0000
371	4.0000

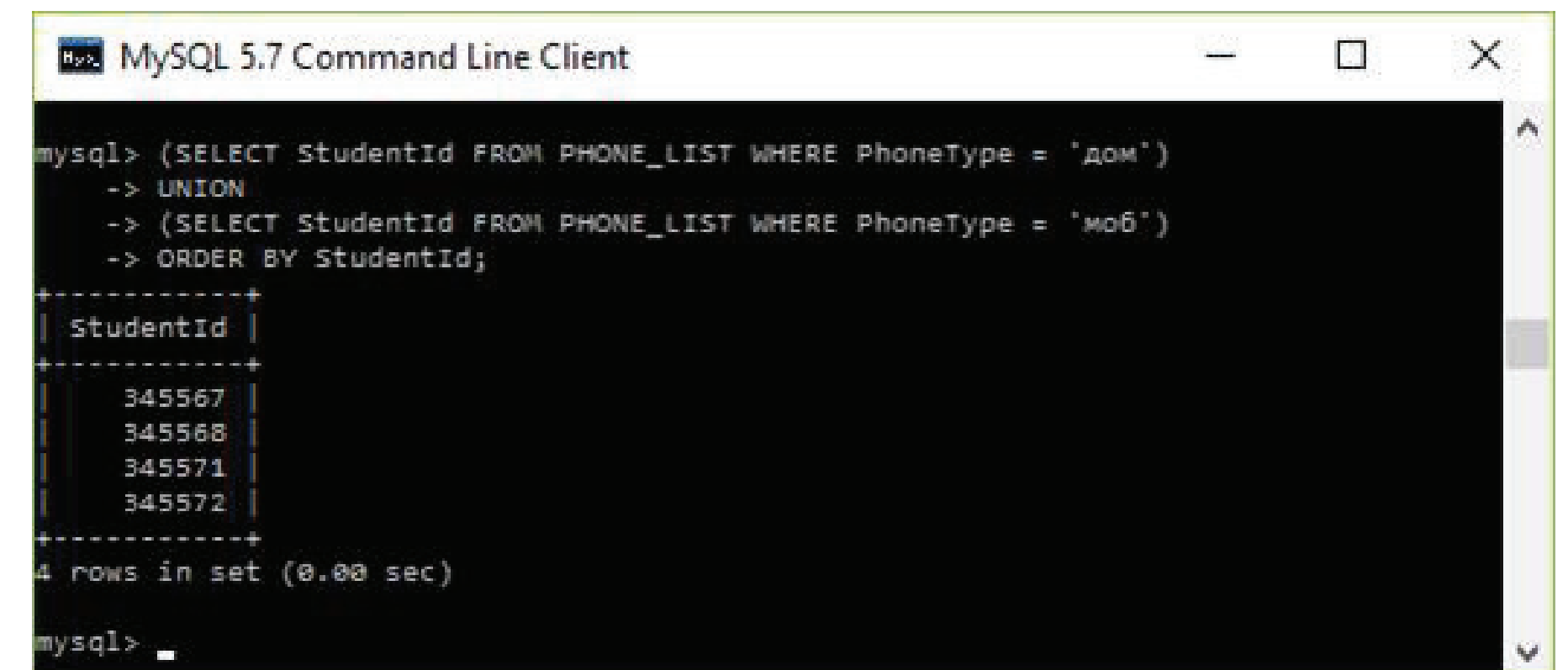
```
2 rows in set (0.03 sec)

mysql>
```

# Пример: использование подзапросов в операторе UNION

```
(SELECT StudentId FROM PHONE_LIST WHERE PhoneType = 'дом')  
UNION  
(SELECT StudentId FROM PHONE_LIST WHERE PhoneType = 'моб')  
ORDER BY StudentId;
```

Список номеров зачетов студентов, у которых  
в базе данных указан мобильный или домашний телефон



```
MySQL 5.7 Command Line Client  
mysql> (SELECT StudentId FROM PHONE_LIST WHERE PhoneType = 'дом')  
-> UNION  
-> (SELECT StudentId FROM PHONE_LIST WHERE PhoneType = 'моб')  
-> ORDER BY StudentId;  
+-----+  
| StudentId |  
+-----+  
| 345567 |  
| 345568 |  
| 345571 |  
| 345572 |  
+-----+  
4 rows in set (0.00 sec)  
  
mysql> _
```