MySQL

UNION
JOIN

UNION оператор объединения

Вывести в одной таблице фамилии студентов и преподавателей из города Находка:

```
SELECT 'Stud', SURNAME, STUDENT_ID
FROM STUDENT
WHERE CITY = 'Находка‘
UNION
SELECT 'Prep',SURNAME, LECTURER_ID
FROM LECTURER
WHERE CITY = 'Находка';
```

Столбцы объединяемых таблиц должны быть совместимы!! (тип, размер, NULL, …)

UNION дублирование

Выбираем города из двух таблиц. Дубликаты будут удалены:

```
SELECT CITY
FROM STUDENT
UNION
SELECT CITY
FROM LECTURER;
```

Если нужно оставить повторяющиеся строки:

```
SELECT CITY
FROM STUDENT
UNION ALL
SELECT CITY
FROM LECTURER;
```

Задание 1

Посчитайте, сколько городов получилось в каждом из двух предыдущих запросов

Семь раз подумай – один агрегируй!

Для каждой даты сдачи экзамена вывести информацию о каждом студенте, который получил максимальную и минимальную оценку:

```
SELECT 'макс_оц', A.STUDENT_ID, SURNAME, MARK, EXAM_DATE
FROM STUDENT A, EXAM_MARKS B
WHERE (
        A.STUDENT_ID = B.STUDENT_ID
        AND B.MARK =
                (SELECT MAX(MARK)
                FROM EXAM_MARKS C
                WHERE C.EXAM_DATE = B.EXAM_DATE)
        )
UNION ALL
SELECT 'мин_оц', A.STUDENT_ID, SURNAME, MARK, EXAM_DATE
FROM STUDENT A, EXAM_MARKS B
WHERE (
        A.STUDENT_ID = B.STUDENT_ID
        AND B.MARK =
                (SELECT MIN(MARK)
                FROM EXAM_MARKS C
                WHERE C.EXAM_DATE = B.EXAM_DATE)
        );
```

Почему так быстрее?

```
SELECT 'макс_оц', A.STUDENT_ID, SURNAME, E.MARK, E.EXAM_DATE
FROM STUDENT A,
(SELECT B.STUDENT_ID, B.MARK, B.EXAM_DATE
FROM EXAM_MARKS B,
(SELECT MAX(MARK) AS MAX_MARK, C.EXAM_DATE
FROM EXAM_MARKS C
GROUP BY C.EXAM_DATE) D
WHERE B.EXAM_DATE=D.EXAM_DATE AND B.MARK=MAX_MARK) E
WHERE A.STUDENT_ID=E.STUDENT_ID

UNION ALL

SELECT 'мин_оц', A.STUDENT_ID, SURNAME, E.MARK, E.EXAM_DATE
FROM STUDENT A,
(SELECT B.STUDENT_ID, B.MARK, B.EXAM_DATE
FROM EXAM_MARKS B,
(SELECT MIN(MARK) AS MIN_MARK, C.EXAM_DATE
FROM EXAM_MARKS C
GROUP BY C.EXAM_DATE) D
WHERE B.EXAM_DATE=D.EXAM_DATE AND B.MARK=MIN_MARK) E
WHERE A.STUDENT_ID=E.STUDENT_ID;
```

UNION + ORDER BY

```
SELECT 'макс_оц', A.STUDENT_ID, SURNAME, E.MARK, E.EXAM_DATE
FROM STUDENT A, (SELECT B.STUDENT_ID, B.MARK, B.EXAM_DATE
FROM EXAM_MARKS B,
(SELECT MAX(MARK) AS MAX_MARK, C.EXAM_DATE
FROM EXAM_MARKS C
GROUP BY C.EXAM_DATE) D
WHERE B.EXAM_DATE=D.EXAM_DATE AND B.MARK=MAX_MARK) E
WHERE A.STUDENT_ID=E.STUDENT_ID

UNION ALL

SELECT 'мин_оц', A.STUDENT_ID, SURNAME, E.MARK, E.EXAM_DATE
FROM STUDENT A, (SELECT B.STUDENT_ID, B.MARK, B.EXAM_DATE
FROM EXAM_MARKS B,
(SELECT MIN(MARK) AS MIN_MARK, C.EXAM_DATE
FROM EXAM_MARKS C
GROUP BY C.EXAM_DATE) D
WHERE B.EXAM_DATE=D.EXAM_DATE AND B.MARK=MIN_MARK) E
WHERE A.STUDENT_ID=E.STUDENT_ID

ORDER BY SURNAME, EXAM_DATE;
```

Внешнее объединение

Второй запрос выбирает строки, исключенные первым запросом.
В таблице STUDENT есть записи о студентах, для которых не указан идентификатор университета. Нужно составить список студентов с указанием названия университета так, чтобы при этом не потерялись студенты с неизвестным университетом.

```
SELECT SURNAME, NAME, UNIV_NAME
FROM STUDENT, UNIVERSITY
WHERE STUDENT.UNIV_ID = UNIVERSITY.UNIV_ID
UNION
SELECT SURNAME, NAME, 'неизвестен‘
FROM STUDENT
WHERE UNIV_ID IS NULL
ORDER BY 3;
```

Задание 2

Создайте объединение двух запросов, которые выдают значения полей UNIV_NAME, CITY, RATING для всех университетов. Те из них, у которых рейтинг равен или выше 300, должны иметь комментарий ‘Высокий’, все остальные – ‘Низкий’.

Полное объединение таблиц

SELECT * FROM STUDENT, UNIVERSITY;

Очень много строк!!! Комбинация каждой строки из первой таблицы с каждой строкой из второй таблицы.

Внутреннее (INNER) объединение таблиц

Нужно получить для каждого студента его фамилию и название университета, который находится в городе проживания студента, т.е. все сочетания записей с фамилиями студентов и названиях университетов, для которых совпадает город (из двух таблиц).

SELECT STUDENT.SURNAME, UNIVERSITY.UNIV_NAME, STUDENT.CITY
FROM STUDENT, UNIVERSITY
WHERE STUDENT.CITY = UNIVERSITY.CITY;

JOIN

SELECT STUDENT.SURNAME, UNIVERSITY.UNIV_NAME, STUDENT.CITY
FROM STUDENT **INNER JOIN** UNIVERSITY
ON STUDENT.CITY = UNIVERSITY.CITY; # INNER можно опустить

Еще раз полное соединение таблиц:

SELECT * **FROM** STUDENT **JOIN** UNIVERSITY;

=

SELECT * FROM STUDENT, UNIVERSITY;

Ключи

Первичный Ключ
Primary key (PK)

Внешний ключ
Foreign key (FK)

А

Ключ	ФИО	Должность	Город FK	Номер телефона	Семейное положение FK
1	Иванов И.И.	1	1	12345	2
2	Петров А.С.	2	2	234567	1
3	Сидоров А.В.	3	1	345678	2
4	Винтиков А.Е.	3	2	456789	1

PK

Д

Ключ	Должность
1	Начальник
2	Старший помощник младшего менеджера
3	Консультант

PK

Г

Ключ	Город
1	Москва
2	С.-Петербург

PK

С

Ключ	Семейное положение
1	Холост
2	Женат

Ключи

PK

STUDENT

STUDENT_ID	SURNAME	NAME	STIPEND	KURS	CITY	BIRTHDAY	UNIV_ID
1	Сергеев	Сергей	520.00	1	Одинцово	1983-03-21	39
2	Клюквина	Вера	140.00	3	Сатка	1987-03-15	6
3	Кийко	Ирина	NULL	2	Зубцов	1986-05-22	82
4	Водопьянова	Виктория	170.00	2	Костерево	1984-04-06	88
5	Тимашов	Дмитрий	90.00	2	Горячий Ключ	1984-06-19	27
6	Авраменко	Владислав	480.00	3	Павловское	1987-07-07	29
7	Величко	Алексей	250.00	1	Кедровый	1984-11-27	8
8	Филиппов	Артем	470.00	4	NULL	1984-08-04	35
9	Колдаева	Елена	360.00	1	Усолье-Сибирское	1989-08-07	86
10	Антипов	Андрей	520.00	2	Яхрома	1984-09-04	77
11	Грачев	Евгений	490.00	3	Боровое	1987-11-26	84
12	Попова	Дарья	120.00	2	Новокуйбышевск	1984-11-28	26

PK FK

EXAM_MARKS

EXAM_ID	STUDENT_ID	SUBJ_ID	MARK	EXAM_DATE
1	10000	3	1	2000-01-10
2	10000	2	2	2000-01-12
3	10000	14	4	2000-06-27
4	10000	2	4	2000-05-09
5	10000	3	2	2000-02-26
6	10000	8	1	2000-03-26
7	2	2	3	2000-03-20
8	2	14	3	2000-05-04
9	2	10	4	2000-03-09
10	3	10	1	2000-06-06

Ссылочная целостность: каждому значению поля STUDENT_ID в таблице EXAM_MARKS обязательно соответствует такое же значение поля STUDENT_ID в таблице STUDENT. В таблице EXAM_MARKS не может быть записей, имеющих такие идентификаторы студентов, которых нет в таблице STUDENT.

Соединение таблиц посредством ссылочной целостности

Получить список фамилий студентов с полученными ими оценками и ID предметов

Способ 1:

```
SELECT SURNAME, MARK, SUBJ_ID
FROM STUDENT, EXAM_MARKS
WHERE STUDENT.STUDENT_ID = EXAM_MARKS.STUDENT_ID;
```

Способ 2 (JOIN):

```
SELECT SURNAME, MARK, SUBJ_ID
FROM STUDENT JOIN EXAM_MARKS
ON STUDENT.STUDENT_ID = EXAM_MARKS.STUDENT_ID;
```

Соединение таблиц посредством ссылочной целостности

Получить список фамилий студентов, получивших «неуд», вместе с ID предметов

Способ 1:

```
SELECT SUBJ_NAME, SURNAME, MARK
FROM STUDENT, SUBJECT, EXAM_MARKS
WHERE STUDENT.STUDENT_ID = EXAM_MARKS.STUDENT_ID
AND SUBJECT.SUBJ_ID = EXAM_MARKS.SUBJ_ID
AND EXAM_MARKS.MARK = 2;
```

Способ 2 (JOIN):

```
SELECT SUBJ_NAME, SURNAME, MARK
FROM STUDENT JOIN SUBJECT JOIN EXAM_MARKS
ON STUDENT.STUDENT_ID = EXAM_MARKS.STUDENT_ID
AND SUBJECT.SUBJ_ID = EXAM_MARKS.SUBJ_ID
AND EXAM_MARKS.MARK = 2;
```

Внешнее соединение таблиц

Получить список фамилий студентов с полученными ими оценками и ID предметов

Если студент еще не сдавал экзамен, то записи о нем не будет:

```
SELECT SURNAME, MARK, SUBJ_ID
FROM STUDENT, EXAM_MARKS
WHERE STUDENT.STUDENT_ID = EXAM_MARKS.STUDENT_ID;
```

Хотим сохранить все строки из таблицы STUDENT:

Способ 1:

```
SELECT SURNAME, MARK
FROM STUDENT LEFT OUTER JOIN EXAM_MARKS
ON STUDENT.STUDENT_ID = EXAM_MARKS.STUDENT_ID;
```

Способ 2:

```
SELECT SURNAME, MARK
FROM EXAM_MARKS RIGHT OUTER JOIN STUDENT
ON EXAM_MARKS.STUDENT_ID = STUDENT.STUDENT_ID;
```

Задание 3

Выведите для каждого студента названия всех предметов, по которым от получил «хорошо» или «отлично».

Псевдонимы при соединении таблиц

Вывести пары фамилий студентов, носящих одно имя.

```
SELECT FIRST.SURNAME, SECOND.SURNAME
FROM STUDENT FIRST, STUDENT SECOND
WHERE FIRST.NAME = SECOND.NAME;
```

```
SELECT FIRST.SURNAME, SECOND.SURNAME
FROM STUDENT FIRST, STUDENT SECOND
WHERE FIRST.NAME = SECOND.NAME
AND FIRST.SURNAME < SECOND.SURNAME;
```

В чем разница?

Задание 3

Написать запрос, выполняющий вывод списка всех пар фамилий студентов, проживающих в одном городе. При этом не включать в список комбинации фамилий студентов самих с собой (то есть комбинацию типа “Иванов-Иванов”) и комбинации фамилий студентов, отличающиеся порядком следования (то есть включать одну из двух комбинаций типа “Иванов-Петров” и “Петров-Иванов”).