SQL. Вводный курс

Лекция №7: JOIN

Формирование связанных запросов

В **SQL** запросы к БД не ограничиваются одной таблицей в предложении **FROM**.

Использование нескольких таблиц внутри **FROM** реализует все возможности реляционной модели.

Обычно при указании нескольких таблиц одну из них называют ***Master***, остальные соответственно ***Detail*** таблицами.

Получить список студентов и учебных заведений в которых они учаться.

SELECT *ST.NAME, SURNAME, UN.NAME*
FROM *STUDENTS ST, UNIVERSITIES UN*

Результат "ДЕКАРТОВО" произведение = каждый из студентов (45) с каждым учебным заведением (15), в итоге 675 записей всех возможных вариантов.

Формирование связанных запросов

Для получения правильной выборки необходимо воспользоваться реляционной моделью и связать две таблицы: вторичный ключ ***Master***-таблицы с первичным ключом ***Detail***-таблицы.

SELECT *SURNAME, ST.NAME, UN.NAME UNIV_NAME*
FROM *STUDENTS ST, UNIVERSITIES UN*
WHERE *ST.UNIV_ID=UN.ID*

SURNAME	NAME	UNIV_NAME
Березовский	Роман	КНУ
Денисенко	Марк	КНУ
Кабанов	Виталий	ХАИ
Пименчук	Дмитрий	ХАИ
Цилюрик	Тимофей	ХСХА
Шуст	Марина	ХСХА
Ориненко	Анатолий	ЛГУ

Формирование подзапросов

Можно использовать подзапросы, связывающие таблицу со своей собственной копией.

Найти идентификаторы, фамилии и стипендии студентов, получающих стипендию выше средней на курсе, на котором они учатся.

```
SELECT STUD.ID, STUD.SURNAME, STUD.STIPEND
    FROM STUDENTS STUD WHERE STIPEND > (
    SELECT AVG(STIPEND)
    FROM STUDENTS WHERE STUD.COURSE=COURSE)
```

Эквивалентный запрос:

```
SELECT S.ID, S.SURNAME, S.STIPEND
FROM STUDENTS S JOIN
    (SELECT COURSE, AVG(STIPEND) STIPEND
     FROM STUDENTS GROUP BY COURSE) S_AVG
ON S.COURSE=S_AVG.COURSE AND S.STIPEND > S_AVG.STIPEND
```

Заметьте второй запрос будет выполнен гораздо быстрее: почему?

Соединение таблиц Оператор JOIN

Виды соединений нескольких таблиц:

- Полное или Декартово произведение

  (!) на практике применяется крайне редко

- Внутрение INNER (by default)

  объединяются строки таблиц, для которых истинно равенство связующих их полей

- Внешние OUTER
  - правое RIGHT
  - левое LEFT

  происходит выборка всех строк основной таблицы и дополнение имеющимися данными из связуемой таблицы, в случае их отсутствия - поля заполняются NULL-ами

Полное соединение CROSS JOIN

Самый простой пример

```
SELECT * FROM STUDENTS, UNIVERSITIES
```

Тоже самое с использованием JOIN

```
SELECT * FROM STUDENTS CROSS JOIN UNIVERSITIES
```

Выбрать все возможные комбинации строк из таблиц STUDENTS, ID которых не превышает 3 и UNIVERSITIES из города Киев?

```
SELECT *
FROM
    (SELECT * FROM STUDENTS WHERE ID < 3) TAB1
CROSS JOIN
    (SELECT * FROM UNIVERSITIES WHERE CITY = 'Киев') TAB2
```

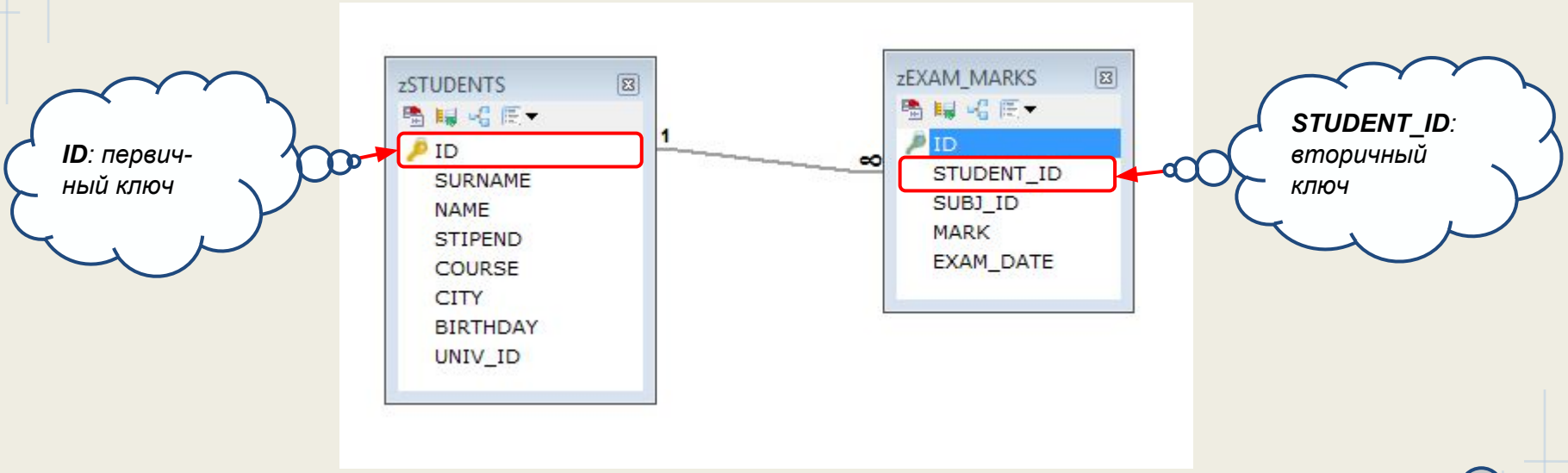
Полное соединение CROSS JOIN

ID	SURNAME	NAME	STIPEND	COURSE	CITY	BIRTHDAY	UNIV_ID	ID	NAME	RATING	CITY
1	Кабанов	Виталий	550	4	Харьков	1990-01-12	2	1	КНУ	608	Киев
2	Павленко	Игорь	600	1	Киев	1993-06-21	5	1	КНУ	608	Киев
1	Кабанов	Виталий	550	4	Харьков	1990-01-12	2	5	КНУСА	512	Киев
2	Павленко	Игорь	600	1	Киев	1993-06-21	5	5	КНУСА	512	Киев
1	Кабанов	Виталий	550	4	Харьков	1990-01-12	2	6	КПИ	568	Киев
2	Павленко	Игорь	600	1	Киев	1993-06-21	5	6	КПИ	568	Киев

WHERE TAB1.UNIV_ID=TAB2.ID

Оператор JOIN (продолжение)

Соединение таблиц посредством ссылочной целостности



каждому значению поля STUDENT_ID в таблице EXAM_MARKS обязательно соответствует такое же значение поля ID в таблице STUDENTS

Оператор JOIN (продолжение)

Классическое применение ссылочной целостности выглядит следующим образом

```
SELECT * FROM STUDENTS S, EXAM_MARKS E
WHERE S.ID = E.STUDENT_ID
```

... с использованием оператора JOIN

```
SELECT * FROM
      STUDENTS S
JOIN
      EXAM_MARKS E
ON S.ID = E.STUDENT_ID
```

Оператор INNER JOIN (продолжение)

Объединение может включать в себя более чем 2 таблицы

Отобрать фамилии всех студентов, получивших оценку "хорошо", вместе с названиями предметов, по которым получена оценка.

```
SELECT SUBJECTS.NAME, SURNAME, MARK
FROM EXAM_MARKS, STUDENTS, SUBJECTS
WHERE STUDENTS.ID = EXAM_MARKS.STUDENT_ID
    AND SUBJECTS.ID = EXAM_MARKS.SUBJ_ID
    AND EXAM_MARKS.MARK = 4
```

или используя JOIN

```
SELECT SUBJECTS.NAME, SURNAME, MARK
FROM EXAM_MARKS
JOIN STUDENTS ON STUDENTS.ID = EXAM_MARKS.STUDENT_ID
JOIN SUBJECTS ON SUBJECTS.ID = EXAM_MARKS.SUBJ_ID
    AND EXAM_MARKS.MARK = 4
WHERE EXAM_MARKS.MARK = 4
```

Оператор INNER JOIN (продолжение)

NAME	SURNAME	MARK
Информатика	Земляный	4
Физика	Федосеева	4
Математика	Чайка	4
Философия	Шуст	4
Философия	Чайка	4
Философия	Федосеева	4
Английский	Козьменко	4
Английский	Пименчук	4
Английский	Миланивская	4

Оператор OUTER JOIN

Иногда INNER JOIN возвращает выборку с нежелательной "потерей" данных.

Так в предыдущем примере в результат запроса не попадут студенты, которые еще не сдавали экзамены, и которые, следовательно, отсутствуют в таблице EXAM_MARKS.

Как отобрать полный список студентов?

Эту задачу можно обойти с помощью UNION.

```
SELECT SURNAME, NAME, MARK, SUBJ_ID
FROM STUDENTS, EXAM_MARKS
WHERE STUDENTS.ID = EXAM_MARKS.STUDENT_ID
UNION ALL
SELECT SURNAME, NAME, NULL, NULL
FROM STUDENTS WHERE NOT EXISTS
        (SELECT 1 FROM EXAM_MARKS
        WHERE STUDENTS.ID = EXAM_MARKS.STUDENT_ID)
```

Либо ... см.дальше

Оператор OUTER JOIN (LEFT)

Внешнее соединение OUTER JOIN выдающее аналогичный результат

```
SELECT SURNAME, MARK, SUBJ_ID
FROM
    STUDENTS
LEFT OUTER JOIN
    EXAM_MARKS
ON STUDENTS.ID = EXAM_MARKS.STUDENT_ID
```

SURNAME	MARK	SUBJ_ID
...		
Земляный	4	1
Осипуков	NULL	NULL
Зевцов	3	4
...		

Какие основные отличия обеих выборок?

Оператор OUTER JOIN (RIGHT)

В зависимости от положения основной таблицы в запросе различают правый и левый OUTER JOIN.

В предыдущем примере был рассмотрен LEFT OUTER JOIN.

Что будет если заменить его на RIGHT?

Мы получим аналог INNER JOIN - в выборке будут отсутствовать строки с NULL значениями, т.к. за основу будет взято подмножество значений ID студентов из таблицы EXAM_MARKS, в которой есть оценки возможно не всех студентов.

```
SELECT SURNAME, MARK, SUBJ_ID
FROM
    STUDENTS
RIGHT OUTER JOIN
    EXAM_MARKS
ON STUDENTS.ID = EXAM_MARKS.
STUDENT_ID
```

```
SELECT SURNAME, MARK, SUBJ_ID
FROM
    EXAM_MARKS
LEFT OUTER JOIN
    STUDENTS
ON STUDENTS.ID = EXAM_MARKS.
STUDENT_ID
```

Оператор объединения UNION & JOIN

Составить отчет, содержащий для каждой даты сдачи экзаменов сведения по каждому студенту, получившему максимальную или минимальную оценки.

```
SELECT 'Макс. оценка' MM, A.ID, SURNAME,
                E.MARK, E.EXAM_DATE
FROM STUDENTS A JOIN
   (SELECT B.STUDENT_ID, B.MARK, B.EXAM_DATE
    FROM EXAM_MARKS B JOIN
       (SELECT MAX(MARK) AS MAX_MARK, EXAM_DATE
        FROM EXAM_MARKS
        GROUP BY EXAM_DATE) D
   ON B.EXAM_DATE=D.EXAM_DATE AND B.MARK=MAX_MARK) E
ON A.ID=E.STUDENT_ID
UNION ALL
SELECT 'Мин. оценка', A.ID, SURNAME, E.MARK, E.EXAM_DATE
FROM STUDENTS A JOIN
   (SELECT B.STUDENT_ID, B.MARK, B.EXAM_DATE
    FROM EXAM_MARKS B JOIN
       (SELECT MIN(MARK) AS MIN_MARK, EXAM_DATE
        FROM EXAM_MARKS
        GROUP BY EXAM_DATE) D
   ON B.EXAM_DATE=D.EXAM_DATE AND B.MARK=MIN_MARK) E
ON A.ID=E.STUDENT_ID
```

3 шаг. Вывести результирующее отношение

2 шаг. Отобрать ID студентов, получивших макс. оценку

1 шаг. Отобрать все макс.оценки по датам

Оператор объединения UNION

MM	ID	SURNAME	MARK	EXAM_DATE
Макс. оценка	6	Березовский	5	2012-06-04
Макс. оценка	11	Корсунская	5	2012-06-04
Мин. оценка	5	Ориненко	3	2012-06-04
Мин. оценка	18	Зевцов	3	2012-06-04
Макс. оценка	19	Бородюк	5	2012-06-05
Мин. оценка	20	Миланивская	3	2012-06-05
Мин. оценка	4	Козь5менко	3	2012-06-05