

МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ
КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

Ордена Трудового Красного Знамени федеральное государственное
бюджетное образовательное учреждение
высшего образования
**«Московский технический университет связи и информатики»
(МТУСИ)**

Кафедра
«Сетевые информационные технологии и сервисы»

Лабораторная работа 5
По дисциплине «Интеллектуальные базы данных»
«Создание сложных запросов на выборку»

Выполнил студент:
М092301(75) Леонов Н.Н.
Вариант 17
Проверил:
Ст.пр. Фатхулин Т.Д.

Москва 2024

1. Цель работы

В данной лабораторной работе необходимо выполнить запросы на выборку с использованием объединения и вложенных запросов.

2. Задание

При выполнении лабораторной работы необходимо для заданной предметной области средствами MySQL:

- для заданной предметной области построить многотабличный запрос на выборку с использованием объединения;
- для заданной предметной области построить запрос на выборку, содержащий вложенный запрос;
- составить отчет по лабораторной работе.

3. Теоретические сведения

Рассмотрим следующие вопросы:

- использование объединений в запросах к нескольким таблицам;
- создание вложенных запросов.

В реальных приложениях часто требуется использовать сразу несколько таблиц БД. Запросы, которые обращаются одновременно к нескольким таблицам, называются многотабличными или сложными запросами.

Абсолютные ссылки на базы данных и таблицы. В запросе можно прямо указывать необходимую БД и таблицу. Например, можно представить ссылку на столбец *u_surname* из таблицы *users* в виде *users.u_surname*. Аналогично можно уточнить БД, таблица из которой упоминается в запросе. Если необходимо, то вместе с БД и таблицей можно указать и столбец, например:

```
mysql> SELECT book.users.u_surname FROM users;
```

u_surname
Иванов
Лосев
Симонов
Кузнецов
Петров
Корнеев

```
6 rows in set (0.00 sec)
```

При использовании сложных запросов это позволяет избежать двусмысленности при указании источника необходимой информации.

Использование объединений для запросов к нескольким таблицам. Хорошо спроектированная реляционная БД эффективна из-за связей между таблицами. При выборе информации из нескольких таблиц такие связи называют объединениями.

В качестве примера объединения двух таблиц рассмотрим запрос, извлекающий из БД *book* фамилии покупателей вместе с номерами сделанных ими заказов:

```
mysql> SELECT orders.order_ID, users.u_surname
-> FROM orders, users
-> WHERE orders.o_user_ID=users.user_ID
-> ORDER BY orders.order_ID;
```

order_ID	u_surname
1	Симонов
2	Корнеев
3	Иванов
4	Кузнецов
5	Симонов

5 rows in set (0.00 sec)

Выражение *WHERE* важно с точки зрения получения результата. Набор условий, используемых для объединения таблиц, называют условием объединения. В данном примере условие связывает таблицы *orders* и *users* по внешним ключам.

Объединение нескольких таблиц аналогично объединению двух таблиц. Например, необходимо выяснить, какому каталогу принадлежит товарная позиция из заказа, сделанного 10 февраля 2009 г. в 09:40:29:

```
mysql> SELECT catalogs.cat_name
-> FROM catalogs,books,orders
-> WHERE o_time='2009-02-10 09:40:29'
-> AND catalogs.cat_ID=books.b_cat_ID
-> AND orders.o_book_ID=books.book_ID;
```

cat_name
Интернет

1 row in set (0.09 sec)

Самообъединение таблиц. Можно объединить таблицу саму с собой (когда интересуют связи между строками одной и той же таблицы). Пусть нужно выяснить, какие книги есть в каталоге, содержащем книгу с названием «Компьютерные сети». Для этого необходимо найти в таблице *books* номер каталога (*b_cat_ID*) с этой книгой, а затем посмотреть в таблице *books* книги этого каталога.

```
mysql> SELECT b2.b_name
-> FROM books b1, books b2
-> WHERE b1.b_name='Компьютерные сети'
-> AND b1.b_cat_ID=b2.b_cat_ID;
```

b_name
Компьютерные сети
Сети. Поиск неисправностей
Безопасность сетей
Анализ и диагностика компьютерных сетей
Локальные вычислительные сети

5 rows in set (0.02 sec)

В этом запросе для таблицы *books* определены два разных псевдонима (две отдельных таблицы *b1* и *b2*, которые должны содержать одни и те же данные). После этого они объединяются, как любые другие таблицы. Сначала ищется строка в таблице *b1*, а затем в таблице *b2* – строки с тем же значением номера каталога.

Основное объединение. Набор таблиц, перечисленных в выражении *FROM* и разделенных запятыми, – это декартово произведение (полное или перекрестное объединение), которое возвращает полный набор комбинаций. Добавление к нему условного выражения *WHERE* превращает его в объединение по эквивалентности, ограничивающее число возвращаемых запросом строк.

Вместо запятой в выражении *FROM* можно использовать ключевое слово *JOIN*. В этом случае вместо *WHERE* лучше использовать ключевое слово *ON*:

```
mysql> SELECT orders.order_ID, users.u_surname
-> FROM orders JOIN users
-> ON orders.o_user_ID=users.user_ID
-> ORDER BY orders.order_ID;
```

order_ID	u_surname
1	Симонов
2	Корнеев
3	Иванов
4	Кузнецов
5	Симонов

5 rows in set (0.03 sec)

Вместо *JOIN* с тем же результатом можно использовать *CROSS JOIN* (перекрестное объединение) или *INNER JOIN* (внутреннее объединение). Пример запроса, выдающего число товарных позиций в каталогах:

```
mysql> SELECT catalogs.cat_name, COUNT(book_ID)
-> FROM catalogs JOIN books ON catalogs.cat_ID=books.b_cat_ID
-> GROUP BY books.b_cat_ID;
```

cat_name	COUNT(book_ID)
Программирование	9
Интернет	6
Базы данных	4
Сети	5
Мультимедиа	6

5 rows in set (0.05 sec)

Допустим, происходит расширение ассортимента и в списке каталогов появляется новый каталог «Компьютеры»:

```
mysql> INSERT INTO catalogs VALUES (NULL, 'Компьютеры');
Query OK, 1 row affected (0.05 sec)
```

```
mysql> SELECT * FROM catalogs;
```

cat_ID	cat_name
1	Программирование
2	Интернет
3	Базы данных
4	Сети
5	Мультимедиа
6	Компьютеры

```
6 rows in set (0.00 sec)
```

Предыдущий запрос не отразит наличие нового каталога (таблица *books* не содержит записей, относящихся к новому каталогу). Выходом является использование левого объединения (таблица *catalogs* должна быть левой таблицей):

```
mysql> SELECT catalogs.cat_name, COUNT(book_ID)
-> FROM catalogs LEFT JOIN books ON catalogs.cat_ID=books.b_cat_ID
-> GROUP BY catalogs.cat_ID;
```

cat_name	COUNT(book_ID)
Компьютеры	0
Программирование	9
Интернет	6
Базы данных	4
Сети	5
Мультимедиа	6

```
5 rows in set (0.00 sec)
```

Пусть нужно вывести список покупателей и число осуществленных ими покупок, причем покупателей необходимо отсортировать по убыванию числа заказов:

```
mysql> SELECT users.u_surname, users.u_name, users.u_patronymic,
-> COUNT(orders.order_ID) AS total
-> FROM users JOIN orders ON users.user_ID=orders.o_user_ID
-> GROUP BY users.user_ID
-> ORDER BY total DESC;
```

u_surname	u_name	u_patronymic	total
Симонов	Игорь	Николаевич	2
Иванов	Александр	Валерьевич	1
Кузнецов	Максим	Петрович	1
Корнеев	Александр	Александрович	1

```
4 rows in set (0.02 sec)
```

В список не входят покупатели, которые не сделали ни одной покупки. Чтобы вывести полный список покупателей, необходимо вместо перекрестного объединения таблиц *users* и *orders* использовать левое объединение (левой таблицей должна быть таблица *users*):

```
mysql> SELECT users.u_surname,users.u_name,users.u_patronymic,
-> COUNT(orders.order_ID) AS total
-> FROM users LEFT JOIN orders ON users.user_ID=orders.o_user_ID
-> GROUP BY users.user_ID
-> ORDER BY total DESC;
```

u_surname	u_name	u_patronymic	total
Симонов	Игорь	Николаевич	2
Иванов	Александр	Валерьевич	1
Корнеев	Александр	Александрович	1
Кузнецов	Максим	Петрович	1
Петров	Анатолий	Юрьевич	0
Лосев	Сергей	Иванович	0

6 rows in set (0.02 sec)

Вложенный запрос. Позволяет использовать результат, возвращаемый одним запросом, в другом запросе. Так как результат возвращает только оператор *SELECT*, то в качестве вложенного запроса всегда выступает *SELECT*-запрос. В качестве внешнего запроса может выступать запрос с участием любого SQL-оператора: *SELECT*, *INSERT*, *UPDATE*, *DELETE*, *CREATE TABLE* и др.

Пусть требуется вывести названия и цены товарных позиций из таблицы *books* для каталога «Базы данных» таблицы *catalogs*:

```
mysql> SELECT b_name,b_price FROM books
-> WHERE b_cat_ID=(SELECT cat_ID FROM catalogs
-> WHERE cat_name='Базы данных')
-> ORDER BY b_price;
```

b_name	b_price
Практикум по Access	87.00
Базы данных. Разработка приложений	189.00
Раскрытие тайн SQL	200.00
Базы данных	326.00

4 rows in set (0.03 sec)

Получить аналогичный результат можно при помощи многотабличного запроса, но имеется ряд задач, которые решаются только при помощи вложенных запросов. Вложенный запрос может применяться не только с условием *WHERE*, но и в конструкциях *DISTINCT*, *GROUP BY*, *ORDER BY*, *LIMIT* и т. д. Различают:

- вложенные запросы, возвращающие одно значение;
- вложенные запросы, возвращающие несколько строк.

В первом случае вложенный запрос возвращает скалярное значение или литерал, которое используется во внешнем запросе (подставляет результат на место своего выполнения). Например, необходимо определить название каталога, содержащего самую дорогую товарную позицию:

```
mysql> SELECT cat_name FROM catalogs
-> WHERE cat_ID=(SELECT b_cat_ID FROM books
-> WHERE b_price=(SELECT MAX(b_price) FROM books));
+-----+
| cat_name |
+-----+
| Программирование |
+-----+
1 row in set (0.00 sec)
```

Наиболее часто вложенные запросы используются в операциях сравнения в условиях, которые задаются ключевыми словами *WHERE*, *HAVING* или *ON*.

Однако следующий вложенный запрос вернет ошибку:

```
mysql> SELECT cat_name FROM catalogs
-> WHERE cat_ID=(SELECT b_cat_ID FROM books);
ERROR 1242 (21000): Subquery returns more than 1 row
```

Чтобы выбрать строки из таблицы *catalogs*, у которых первичный ключ совпадает с одним из значений, возвращаемых вложенным запросом, следует воспользоваться конструкцией *IN*:

```
mysql> SELECT cat_name FROM catalogs
-> WHERE cat_ID IN (SELECT b_cat_ID FROM books GROUP BY b_cat_ID);
+-----+
| cat_name |
+-----+
| Программирование |
| Интернет |
| Базы данных |
| Сети |
| Мультимедиа |
+-----+
5 rows in set (0.17 sec)
```

Ключевое слово *ANY* может применяться с использованием любого оператора сравнения. Используется логика *ИЛИ*, т. е. достаточно, чтобы срабатывало хотя бы одно из многих условий. Запрос вида *WHERE X > ANY (SELECT Y ...)* можно интерпретировать как «где *X* больше хотя бы одного выбранного *Y*». Соответственно, запрос вида *WHERE X < ANY (SELECT Y ...)* интерпретируется как «где *X* меньше хотя бы одного выбранного *Y*». Рассмотрим запрос, возвращающий имена и фамилии покупателей, совершивших хотя бы одну покупку:

```
mysql> SELECT u_name, u_surname FROM users
-> WHERE user_ID=ANY(SELECT o_user_ID FROM orders);
+-----+-----+
| u_name | u_surname |
+-----+-----+
| Александр | Иванов |
| Игорь | Симонов |
| Максим | Кузнецов |
| Александр | Корнеев |
+-----+-----+
4 rows in set (0.03 sec)
```

Ключевое слово *ALL* также может применяться с использованием любого оператора сравнения, но при этом используется логика *И*, то есть должны срабатывать все условия. Запрос вида *WHERE X > ALL (SELECT Y ...)* интерпретируется как «где *X* больше любого выбранного *Y*». Соответственно, запрос вида *WHERE X < ALL (SELECT Y ...)* интерпретируется как «где *X*

меньше, чем все выбранные Y ». Рассмотрим запрос, возвращающий все товарные позиции, цена которых превышает среднюю цену каждого из каталогов:

```
mysql> SELECT b_name, b_price FROM books
-> WHERE b_price > ALL (SELECT AVG(b_price) FROM books
-> GROUP BY b_cat_ID);
```

b_name	b_price
Visual FoxPro 9.0	660.00
Delphi. Полное руководство	500.00
Совершенный код	771.00
Принципы маршрутизации в Internet	428.00
Компьютерные сети	630.00
Сети. Поиск неисправностей	434.00
Безопасность сетей	462.00

7 rows in set (0.03 sec)

Результирующая таблица, возвращаемая вложенным запросом, может не содержать ни одной строки. Для проверки этого факта могут использоваться ключевые слова *EXISTS* и *NOT EXISTS*.

Запрос, формирующий список покупателей, совершивших хотя бы одну покупку, можно записать следующим образом:

```
mysql> SELECT u_name, u_surname FROM users
-> WHERE EXISTS (SELECT * FROM orders
-> WHERE orders.o_user_ID=users.user_ID);
```

u_name	u_surname
Александр	Иванов
Игорь	Симонов
Максим	Кузнецов
Александр	Корнеев

4 rows in set (0.00 sec)

4. Выполнение лабораторной работы

Выполним запрос с объединением двух таблиц. Необходимо получить данные о дате закупок в соответствии с названием отделов, отсортированных по их новизне (рис. 1). Для этого выполним команду

```
SELECT d_name AS 'Отдел', p_date AS 'Дата' FROM Department, Purchase WHERE
Purchase.p_d_ID = Department.d_ID ORDER BY p_date DESC
```


1 SELECT d_name AS 'Отдел', p_date AS 'Дата' FROM Department, Purchase WHERE Purchase.p_d_ID = Department.d_ID ORDER BY p_date DESC	
Grid view Form view	
Total rows loaded: 4	
Отдел	Дата
1 section_4	2024-01-10 10:01:35
2 ac_section	2024-01-10 10:01:35
3 m_section	2024-01-09 10:01:35
4 section_1	2024-01-09 10:00:35

Рисунок 1 – Результат выборки с объединением таблиц

Также выполним запрос с использованием вложенности. Необходимо вывести данные о затратах, которые достигают или превышают минимальный лимит из всех видов затрат, предоставляемых компанией (рис. 2). Для этого выполним команду

SELECT p_ID AS 'Номер заказа' FROM Purchase WHERE p_sum >= (SELECT MIN(exp_limit) FROM Expenses)

1 SELECT p_ID AS 'Номер заказа' FROM Purchase WHERE p_sum >= (SELECT MIN(exp_limit) FROM Expenses)	
Grid view Form view	
Total rows loaded: 1	
Номер заказа	
1	3

Рисунок 2 – Результат выборки с использованием вложенного запроса

Выводы

В ходе лабораторной работы были произведены сложные запросы на выборки с использованием объединения и вложенных запросов на основе заполненной базы данных «db_cost_accounting» в лабораторной работе №3.