# **Практическая работа № 7** Подзапросы

Объём учебного времени – 2ч

Методические рекомендации

1. Цель работы: научиться работать с подзапросами.
2. Перечень необходимых средств обучения:
   1. Программное обеспечение:

* ОС Windows 10
* СУБД Oracle

1. Основные теоретические положения:

* Внутренний (вложенный) запрос генерирует значение, которое проверяется в предикате внешнего запроса.
* Имеются два вида подзапросов: простой и соотнесенный. Соотнесенный подзапрос использует во внутреннем запросе таблицу, указанную в предложении внешнего запроса FROM , а простой подзапрос – нет.

Пример 1.

* SELECT \*
* FROM Продажи
* WHERE kod\_pr =
* (SELECT kod\_pr
* FROM Продавец
* WHERE name = 'Иванов И.И.');
* Для неключевых атрибутов можно применить DISTINCT, чтобы вынудить простой подзапрос генерировать одиночное значение.
* Простой подзапрос может вывести более одной строки, если в нем использован специальный оператор IN. В этом случае подзапрос выполняется многократно и формируется набор значений для оператора IN.
* Простой подзапрос можно поместить внутрь предложения HAVING, и он будет выполняться по одному разу для очередного значения из поля в предложении GROUP BY.
* Простой подзапрос может содержать агрегатные функции, производящие одиночное значение, или предложения GROUP BY и HAVING.

Соотнесенный подзапрос выполняется неоднократно, по одному разу для каждой строки таблицы основного запроса.

* Пример 2.
* SELECT \*
* FROM Студенты
* WHERE группа in
* ( SELECT группа
* FROM Группа
* WHERE специальность = ‘230115');
* **Запрос, содержащий соотнесенный запрос, выполняется следующим образом.**
* Шаг 1. Выбирается строка из таблицы, именованной во внешнем запросе. Эта строка называется текущей строкой-кандидатом.
* Шаг 2. Значение из этой строки-кандидата запоминается.
* Шаг 3. Выполняется подзапрос, причем во всех местах появления имени таблицы из внешнего запроса используется значение из текущей строки-кандидата.
* Шаг 4. Оценивается предикат внешнего запроса и в случае его истинности  выводится строка-кандидат.
* Процедура повторяется для всех строк таблицы.
* Для того чтобы ссылаться к одной и той же таблице во внутреннем и внешнем запросах, используют разные псевдонимы этой таблицы.

**Подзапросы с ANY (любой), SOME (некоторый), ALL (все)**

В случае использования оператора ALL предикат является истинным, если каждое из значений, выбранных подзапросом, удовлетворяет условию в предикате запроса.

Пример 3. выведем данные на тех детей, фамилии которых предшествуют по алфавиту фамилиям детей с 1995 годом рождения:

* SELECT \* FROM РЕБЕНОК
* WHERE ФИО\_Р < ALL (SELECT ФИО\_Р FROM РЕБЕНОК
* WHERE Год\_рождения = 1995 );

Отличие между ALL и SOME (ANY) – действие в ситуации, когда подзапрос не возвращает никаких значений: если множество результатов подзапроса пусто, ALL истинен, а SOME (ANY) ложен.

* Слово ALL, которое должно следовать за операцией сравнения, означает "возвратить TRUE, если сравнение дает TRUE для всех строк, возвращаемых подзапросом".

**Использование оператора EXISTS**

EXISTS - это оператор, который производит верное или неверное значение. Оператор EXISTS используется, чтобы оценить факт истинности вывода для соотнесенного подзапроса. Подзапрос рассматривается как аргумент и оценивается как истинный, если тот производит любой вывод или как неистинный – в противном случае. Оператор EXISTS отличается от других операторов тем, что он не может быть неизвестным.

Пример 4.

* SELECT \*
* FROM Студенты
* WHERE EXISTS
* (SELECT \*
* FROM Группа
* WHERE специальность = ‘230115');

Пример 5. Выведем данные на студентов, живущих по одному адресу:

* SELECT \* FROM СТУДЕНТ первый
* WHERE EXISTS ( SELECT \* FROM СТУДЕНТ второй
* WHERE первый.Адрес = второй .Адрес
* AND первый.ФИО\_НТ < > второй .ФИО\_НТ );

**Вынесенные подзапросы (именованные подзапросы), или разложение запроса на подзапросы с помощью фразы WITH**

Oracle допускает вынесение определений подзапросов из тела основного запроса с помощью особой фразы WITH.

Фраза WITH используется в двух целях:

* для придания запросу **формулировки**, более понятной программисту (просто *subquery* factoring)
* для записи **рекурсивных запросов** (recursive *subquery* factoring).

могут использоваться совместно

Первый вариант фразы WITH не отменяет описательного характера предложения SELECT и (помимо удобства формулировки) способен разве что дать ускоренное общее выполнение. Рекурсивный же вариант фразы WITH по сути откровенно процедурен и тем противоречит описательному характеру предложения SELECT, положенному в основу SQL.

Фраза WITH предшествует фразе SELECT и позволяет привести сразу несколько предварительных формулировок подзапросов для ссылки на них в нижеформулируемом основном запросе

Общая схема:

**WITH**

**x AS ( SELECT ... )**

**, y AS ( SELECT ... FROM x )**

**, z AS ( SELECT ... FROM x, y )**

**SELECT ... FROM x, y, z, w**

**;**

**Пример**

**WITH**

**commissioners AS ( SELECT \* FROM emp WHERE comm IS NOT NULL )**

**SELECT**

**ename**

**, deptno**

**, sal + comm AS earnings**

**FROM commissioners**

**;**

Фраза WITH может использоваться для формулирования рекурсивных запросов, в соответствии (неполном) со стандартом SQL:1999

Операция **UNION ALL** здесь используется символически, в рамках определенного контекста, для указания способа рекурсивного накопления результата

WITH

numbers ( n ) AS (

SELECT 1 AS n FROM dual -- исходное множество -- одна строка

UNION ALL -- символическое "объединение" строк

SELECT n + 1 AS n -- рекурсия: добавок к предыдущему результату

FROM numbers -- предыдущий результат в качестве источника данных

WHERE n < 5 -- если не ограничить, будет бесконечная рекурсия

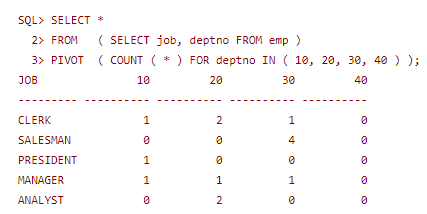
)

SELECT n FROM numbers -- основной запрос

;

**Соединение фраз SELECT и FROM фразами PIVOT/UNPIVOT**

Сочетание SELECT … FROM … *PIVOT* … позволяет развернуть данные одного столбца в отдельные столбцы конечного результата



**Аналог данного запроса без pivot**

SELECT

JOB

, COUNT ( CASE WHEN deptno = 10 THEN 1 END ) "10"

, COUNT ( CASE WHEN deptno = 20 THEN 1 END ) "20"

, COUNT ( CASE WHEN deptno = 30 THEN 1 END ) "30"

, COUNT ( CASE WHEN deptno = 40 THEN 1 END ) "40"

FROM ( SELECT job, deptno FROM emp )

GROUP BY job

;

**Сворачивание данных в столбец указанием UNPIVOT**

**Фраза UNPIVOT выполняет действие, содержательно противоположное фразе PIVOT**

**Создание таблицы по запросу:**

**CREATE TABLE total AS**

**SELECT \***

**FROM ( SELECT job, deptno FROM emp )**

**PIVOT ( COUNT ( \* ) FOR deptno IN ( 10, 20, 30, 40 ) );**

**Выдача данных со сворачиванием показателей в один столбец:**

**SQL> SELECT \***

**2 FROM total**

**3 UNPIVOT ( jobcount FOR deptno IN ( "10", "20", "30", "40" ) )**

**4 ;**

1. Практические указания:
   1. **Использование подзапросов, возвращающих одно значение**
      1. Выведите информацию об авторах книги под названием «*Ваше название книги*».
      2. Подзапрос с DISTINCT: Произведите вывод списка книг, которые были взяты пользователями библиотеки.
      3. Агрегатные функции в подзапросах: Выведите на экран все книги, которые имеют стоимость, большую средней стоимости всех книг.
      4. Найдите все книги библиотеки, которые имеет стоимость такую же, как книга «Война и мир».
      5. HAVING: Выведите список количество книг каждого автора, год рождения которых от 1900 до 1920.
   2. **Подзапросы с IN**
      1. Выведите список авторов книг, которые написали книги жанра «фантастика».
      2. Найти имена авторов, которые принимали участие в написании, по крайней мере, одной книги, выданной пользователям библиотеки категории «студент».
      3. Напишите запрос на изменение записей, который удвоит цену всех книг, написанные автором Пушкин.
      4. Напишите запрос на удаление всех данных о выдачи книг автора Чуковский К.И.
   3. **Подзапросы с ANY, SOME, ALL**
      1. ANY: Напишите запрос для определения списка книг, которые взяли хотя бы один пользователь.
      2. SOME: Напишите запрос для вывода списка пользователей библиотеки, год рождения которых по алфавиту больше года рождения читателей категории студент.
      3. ALL. найти книги, которые стоят больше, чем самая дорогая книга жанра Роман.
   4. **Подзапросы с EXISTS, NOT EXISTS**
      1. Напишите подзапрос для вывода фамилий авторов, которые пишут книги одного или более жанров.
      2. Напишите запрос, для нахождения списка книг, которые на взяли ни один пользователь.
      3. Напишите запрос для вывода списка книг, которые были взяты хотя бы одним пользователем.
      4. Напишите запрос для вывода фамилий авторов без повторений, которые написали книги каждого жанра.
   5. **Вынесенные (именованные) подзапросы**
      1. Распределите всех пользователей библиотеки по группам по возрасту
         * От 7 до 17 – школьники
         * От 18 до 24 – студенты
         * От 25 до 65 – служащие
         * От 66 – пенсионеры

По каждой группе найдите количество выданных книг

* 1. **Использование Pivot**
     1. По каждому разделу книги найдите среднюю цену

1. Содержание отчёта по практической работе
   1. Цель работы.
   2. Результаты выполнения работы;
   3. Выполненные работы в электронном виде представляются на сетевом диске H (X)
2. Контрольные вопросы
   1. Функции работы с датой и временем.
   2. Функции работы со строками.
3. Список рекомендуемой литературы:
   1. Агальцов В.П. Базы данных. Распределённые и удалённые базы данных. Учебник М.: ИД «Форум»: ИНФРА-М, 2011. - 272с.
   2. Дунаев В.В. Базы данных. Язык SQL. СПб.: БХВ-Петербург-2006, 288с.
   3. Дюбуа П. MySQL.: учеб.пособие.- М.-:Вильямс,2007.- 1168с.
   4. Кузнецов М.В., MySQL 5, СПб.: БХВ-Петербург, 2010, 1024с.
   5. Петров В.Н. Избачков Ю.С., Информационные системы. СПб.: Питер, 2006, 656с.
   6. Шелдон Р. MySQL.: учеб.пособие.- М.-:Вильямс,2007.- 880с.