



Базы данных и Информационные системы

9/15 Вложенные запросы

Кузиков Б.О.
Сумы, СумГУ
2013



Задачи занятия

- ▶ После завершения занятия вы должны уметь и знать следующее:
 - ▶ Типы задач, которые могут быть решены с помощью вложенных запросов
 - ▶ Определение вложенного запроса
 - ▶ Список типов вложенных запросов
 - ▶ Писать однострочные и многострочные вложенные запросы



Использование вложенного запроса для решения задачи

- ▶ “У кого зарплата больше, чем у Джона?”

Основной запрос



“Кто из работников имеет зарплату больше, чем Джон?”

Вложенный запрос



“А какая зарплата у Джона?”



Подзапросы

```
SELECT select_list  
FROM table  
WHERE expr operator
```


```
(SELECT select_list  
FROM table);
```

- ▶ Подзапросы (вложенные запросы) выполняются единожды перед выполнением основного запроса.
- ▶ Результат выполнения подзапроса используется в основном запросе (внешнем запросе).



Использование подзапросов

```
SQL> SELECT  ename
2  FROM      emp
3  WHERE     sal > 2975
4             (SELECT sal
5              FROM    emp
6              WHERE   empno=7566) ;
```



```
ENAME
```

```
-----
```

```
KING
```

```
FORD
```

```
SCOTT
```



Советы по использованию подзапросов

- ▶ Заклучайте подзапросы в скобки.
- ▶ Располагайте подзапросы в правой части операторов сравнения.
- ▶ Не используйте предложение **ORDER BY** в подзапросах.
- ▶ Используйте «однострочные» операторы для результатов подзапросов, возвращающих одну строку.
- ▶ Используйте «многострочные» операторы для результатов подзапросов, возвращающих набор строк.



Виды подзапросов

- ▶ Однострочный подзапрос



- ▶ Многострочный подзапрос



- ▶ Многостолбцовый подзапрос



Однострочные подзапросы

- ▶ Возвращает только одну строку
- ▶ Используйте «однострочные» операторы

Оператор	Значение
=	Равны
>	Больше
>=	Больше либо равен
<	Меньше
<=	Меньше либо равен
<>	Не равны



Выполнение однострочных подзапросов

```
SQL> SELECT      ename, job
  2  FROM          emp
  3  WHERE         job =
  4                (SELECT      job
  5                  FROM        emp
  6                  WHERE        empno = 7369)
  7  AND          sal >
  8                (SELECT      sal
  9                  FROM        emp
 10                  WHERE        empno = 7876) ;
```

The diagram illustrates the execution of a SQL query with two subqueries. The main query is a SELECT statement with two WHERE clauses. The first WHERE clause uses a subquery to filter for employees with the job 'CLERK'. The second WHERE clause uses a subquery to filter for employees with a salary greater than 1100. Red arrows indicate the flow of data from the subqueries to the main query conditions.

ENAME	JOB
-----	-----
MILLER	CLERK

Будет ли выполнено данное выражение?

```
SQL> SELECT ename, job
2 FROM emp
3 WHERE job =
4         (SELECT job
5          FROM emp
6          WHERE ename='SMYTHE' );
```

```
no rows selected
```

Подзапрос не возвращает значений



Многострочные подзапросы

- ▶ Возвращают более одной строки
- ▶ Используйте «многострочные» операторы сравнения

Оператор	Значение
IN	Совпадает с любым из списка
ANY <i>SOME</i>	Сравнивает значение со всеми значениями в смысле СУЩЕСТВУЕТ
ALL	Сравнивает значение со всеми значениями в смысле ДЛЯ ВСЕХ

- ▶ Запросы:
 - ▶ `SELECT * FROM a WHERE a.b IN (...)`
 - ▶ `SELECT * FROM a WHERE a.b = ANY (...)`

Равнозначны



Использование оператора ANY в многострочных подзапросах

```
SQL> SELECT empno, ename, job 1300
2 FROM emp 1100
3 WHERE sal < ANY 800
4 (SELECT sal 950
5 FROM emp
6 WHERE job = 'CLERK')
7 AND job <> 'CLERK';
```

EMPNO	ENAME	JOB
7654	MARTIN	SALESMAN
7521	WARD	SALESMAN

Выводы

- ▶ Подзапросы полезны, если запрос основывается на неизвестных данных

```
SELECT  select_list
FROM    table
WHERE   expr operator
        (SELECT select_list
         FROM   table);
```



Связанные подзапросы

- ▶ Коррелированные подзапросы выполняются для каждой строки входных данных



Связанные подзапросы

- ▶ The subquery references a column from a table in the parent query.

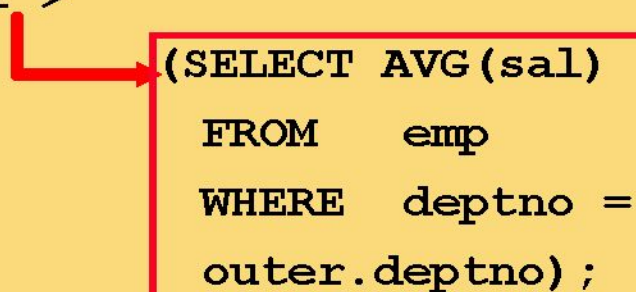
```
SELECT column1, column2, ...
FROM table1 outer
WHERE column1 operator
           (SELECT column1, column2
            FROM table2
            WHERE expr1 =
                  outer.expr2);
```



Использование связанных подзапросов

- ▶ Найдите сотрудников, которые получают больше, чем средняя зарплата по их отделу

```
SELECT ename, salary, deptno
FROM emp outer
WHERE sal >
      (SELECT AVG (sal)
       FROM emp
       WHERE deptno =
         outer.deptno);
```



Подзапрос выполнится для
каждого сотрудника



Найти сотрудников, которые руководят хотя-бы одним подчиненным

```
SELECT empno, job, deptno
FROM   emp1 tbl1
WHERE  EXISTS ( SELECT 'X'
                FROM   emp
                WHERE  mng =
                       tbl1.empno) ;
```



Найдите отделы, в которых нет сотрудников

```
SELECT deptno, dname
FROM dept d
WHERE NOT EXISTS (SELECT 'X'
                  FROM emp
                  WHERE deptno = d.deptno);
```



Связанный подзапрос в UPDATE

- ▶ Связанный подзапрос в UPDATE может быть использован для обновления одной таблицы данными другой таблицы

```
UPDATE table1 alias1
SET    column = (SELECT expression
                   FROM    table2 alias2
                   WHERE   alias1.column =
                           alias2.column);
```



Связанный подзапрос в UPDATE

- ▶ Денормализуем таблицу EMP16, добавив в нее столбец с названием отдела.
- ▶ Заполним таблицу необходимыми данными.

```
ALTER TABLE emp16  
ADD(department_name VARCHAR2(25));
```

```
UPDATE emp16 e  
SET    department_name =  
        (SELECT dname  
         FROM   dept d  
         WHERE  e.deptno = d.deptno);
```



Связанный подзапрос в DELETE

- ▶ Связанный подзапрос в может быть использован для удаления строк на основе строк других таблиц

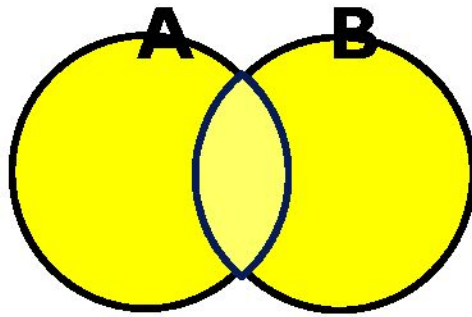
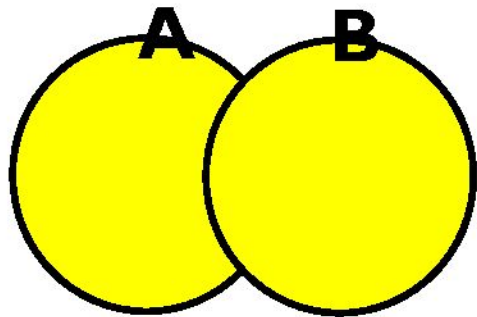
```
DELETE FROM table1 alias1
WHERE column operator
      (SELECT expression
       FROM table2 alias2
       WHERE alias1.column = alias2.column);
```



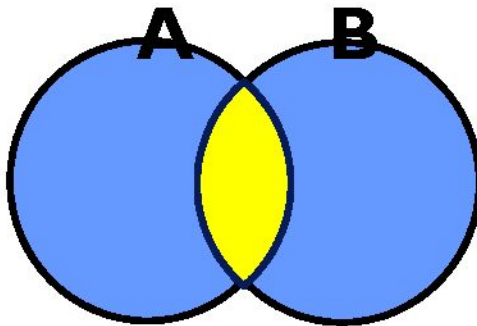
Операции над множествами



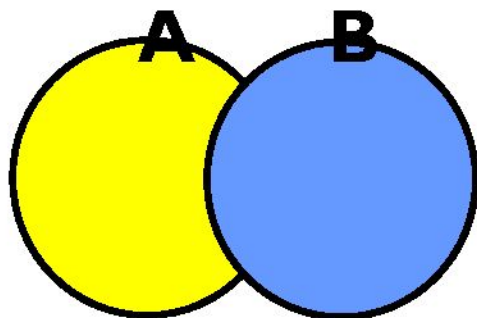
Операции над множествами



UNION/UNION ALL



INTERSECT



MINUS



Руководство по операциям на множествах

- ▶ В запросах `SELECT` должно возвращаться одинаковое количество столбцов
- ▶ Типы столбцов первого и второго запроса должны попарно совпадать.
- ▶ Для удобства можно использовать скобочки.
- ▶ `ORDER BY` может быть использован только в самом конце выражения



Особенности Oracle Server

- ▶ Повторяющиеся строки автоматически исключаются, за исключением операции `UNION ALL`.
- ▶ Имена столбцов первого запроса становятся именами столбцов результирующего запроса.
- ▶ По умолчанию результаты отсортированы по возрастанию, за исключением `UNION ALL`.



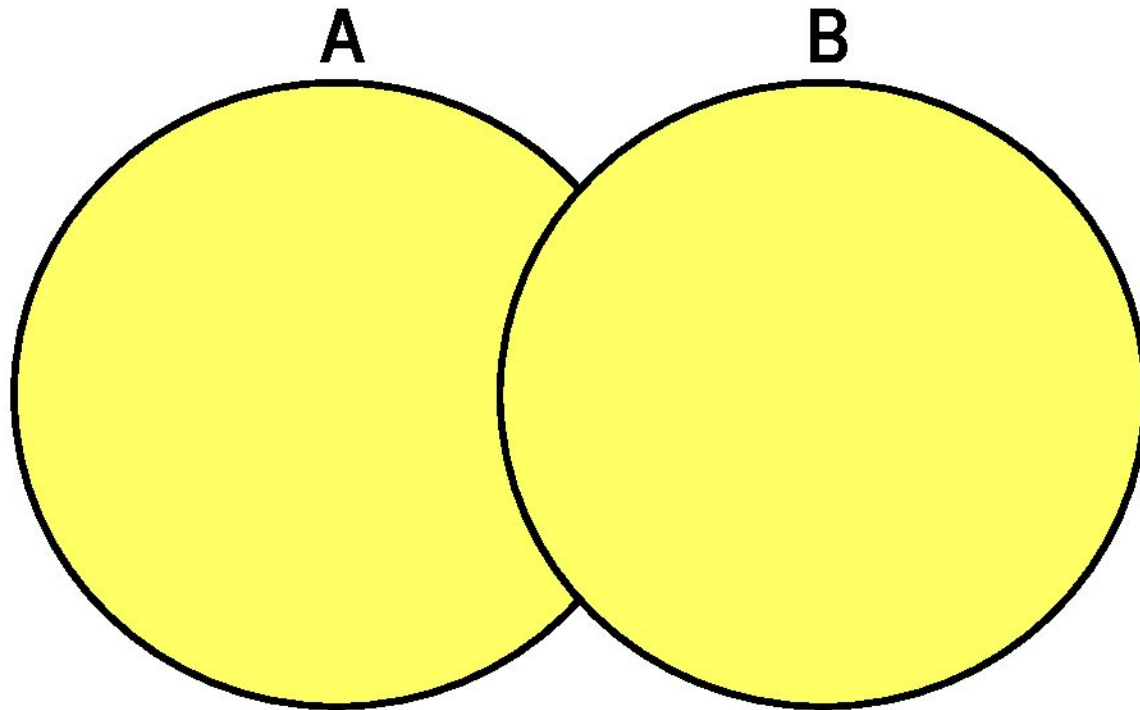
Таблицы в примерах

- ▶ EMP: Таблица хранит текущую должность рабочего
- ▶ JOB_HISTORY: Таблица хранит пару значений (рабочий-должность). В таблице сохранены все когда-либо работавшие служащие и все должности которые занимал служащий **за исключением предыдущей**.



UNION

- ▶ Оператор `UNION` вернет строки из обеих таблиц, за исключением повторяющихся.



UNION

- ▶ Запрос вернет историю всех должностей всех рабочих (включая уволенных).

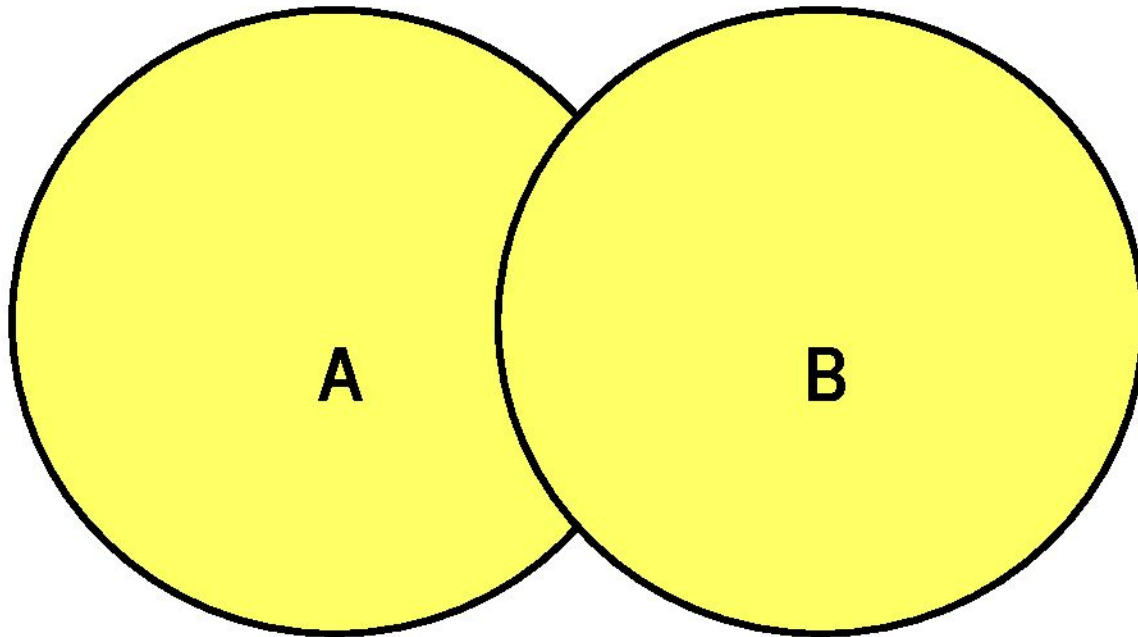
```
SELECT empno, job
FROM emp
UNION
SELECT empno, job
FROM job_history;
```

...



UNION ALL

- ▶ Оператор `UNION ALL` вернет строки из обеих таблиц, включая повторяющиеся.



UNION ALL

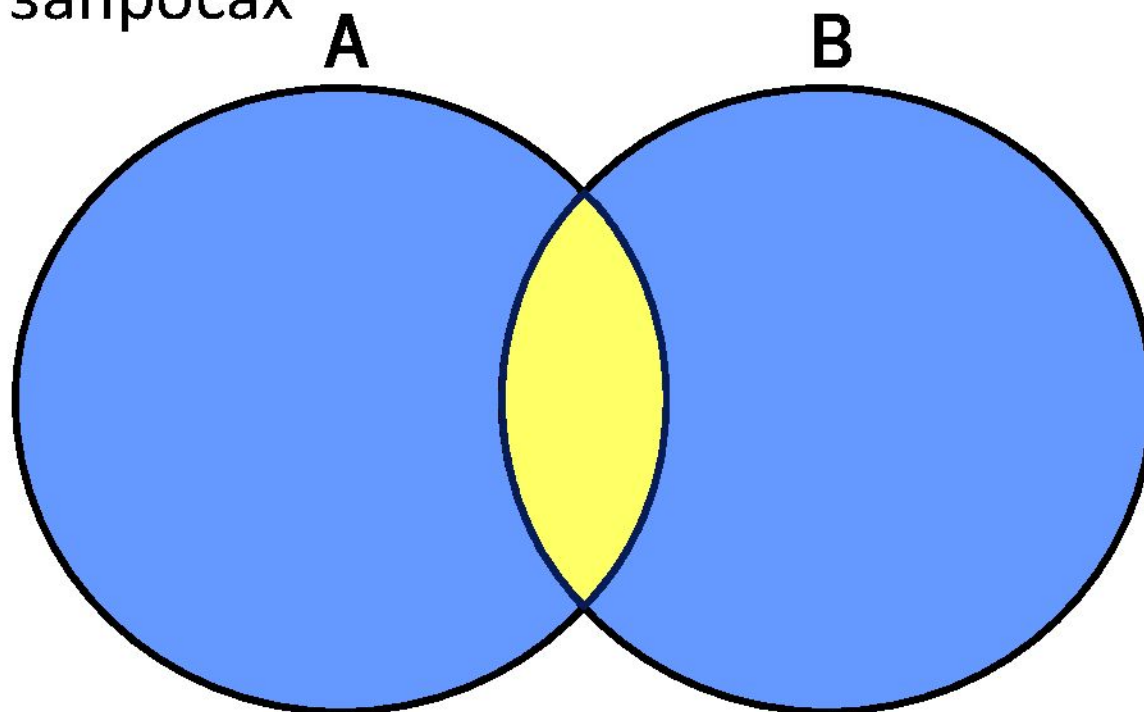
- ▶ Запрос отобразит всех служащих, их текущие и предыдущие должности.

```
SELECT empno, job, depno
FROM emp
UNION ALL
SELECT emp, job, depno
FROM job_history
ORDER BY empno;
```



INTERSECT (Пересечение)

- ▶ INTERSECT возвращает строки, которые есть в обоих запросах



Using the INTERSECT Operator

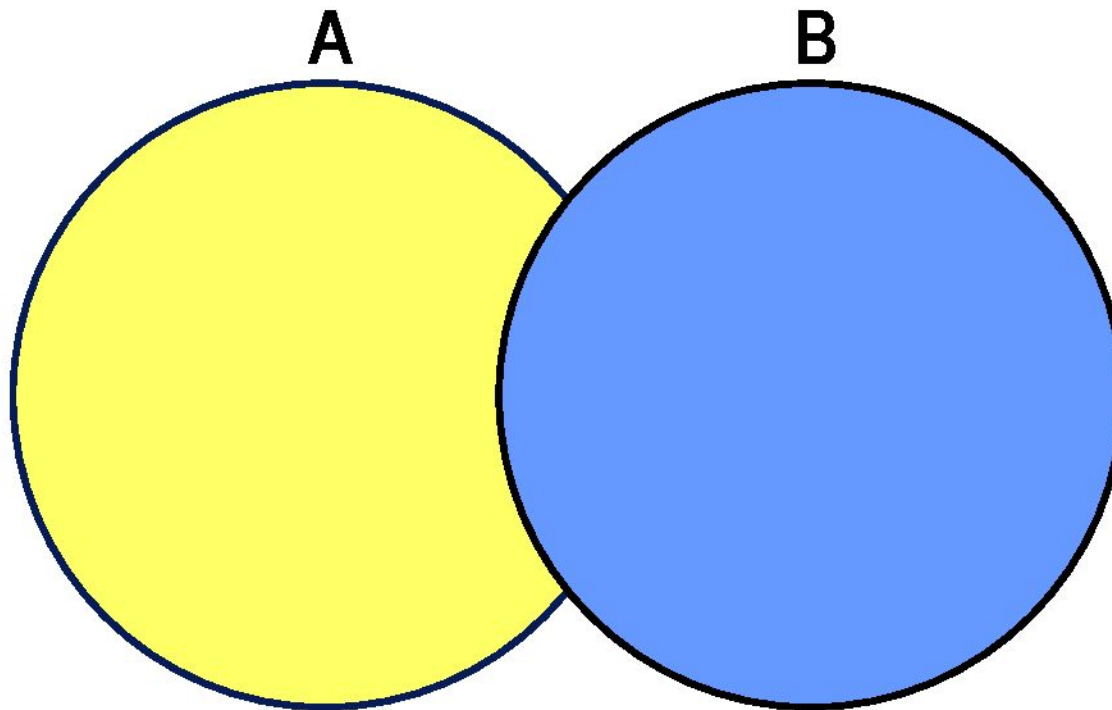
- ▶ Запрос вернет служащих, которые когда либо раньше работали на должностях, которые занимают сейчас.

```
SELECT empno, job
FROM emp
INTERSECT
SELECT empno, job
FROM job_history;
```



MINUS

- ▶ Оператор `MINUS` все неповторяющиеся строки из первого запроса, которых нет во втором запросе



MINUS

- ▶ Запрос вернет служащих, которые никогда не меняли должность (их нет в таблице job_history)

```
SELECT empno
FROM emp
MINUS
SELECT empno
FROM job_history;
```

...



Использование операторов

- ▶ Используя операторы действий над множествами
 - ▶ Убедитесь что все запросы возвращают одинаковое количество атрибутов
 - ▶ Типы атрибутов попарно совпадают

```
SELECT empno, job, salary
FROM emp
UNION
SELECT empno, job, 0
FROM job_history;
```



Итоги

- ▶ Подзапросы
- ▶ Операции над множествами
 - ▶ UNION возвращает все строки без повторений
 - ▶ UNION ALL возвращает все строки с повторениями
 - ▶ INTERSECT возвращает строки, которые есть в обоих запросах
 - ▶ MINUS возвращает строки, которые есть в первом запросе, но нет во втором
 - ▶ ORDER BY можно использовать только самом конце запроса

