Курс QA. Выборки из одной таблицы с помощью SQL

NetCracker®

Часть 1. Simple SELECT

NetCracker®

SELECT

```
SELECT [ALL | DISTINCT | TOP]
{*, column [alias], ...}
FROM table;
```


Ограничение выбираемых строк

Ограничение производится с использованием предложения WHERE.

```
SELECT      {*, column [alias], ...}
FROM        table
[WHERE      condition(s)];
```

Предложение WHERE следует за предложением FROM.

Использование предложения WHERE

```
SQL> SELECT ename, job, deptno
  2  FROM   emp
  3  WHERE  job='CLERK';
```

ENAME	JOB	DEPTNO
JAMES	CLERK	30
SMITH	CLERK	20
ADAMS	CLERK	20
MILLER	CLERK	10

Символьные строки и даты

- Символьные строки и даты заключаются в одинарные кавычки
- Символьные значения чувствительны к регистру, а даты чувствительны к формату
- Для дат формат по умолчанию имеет вид 'DD.MM.YY'

```
SQL> SELECT   ename, job, deptno
  2  FROM     emp
  3  WHERE    ename = 'JAMES';
```

Операторы сравнения

Обозначение	Назначение
=	Равны
>	Больше
>=	Больше или равны
<	Меньше
<=	Меньше или равны
<> != ^=	Не равны
IS Null	Является NULL
IS NOT NULL	Не является NULL

Использование операторов сравнения

```
SQL> SELECT ename, sal, comm
  2  FROM   emp
  3  WHERE  sal<=comm;
```

```
ENAME           SAL       COMM
---------- ---------- ----------
MARTIN         1250 <--> 1400
```


Прочие операторы сравнения

Обозначение	Назначение
BETWEEN … AND …	Находится между
LIKE	Похоже на шаблон
IN (…)	Принадлежит списку

Использование оператора BETWEEN

Используйте оператор BETWEEN для выбора значений, лежащих в диапазоне.

```
SQL> SELECT      ename, sal
  2  FROM  emp
  3  WHERE sal BETWEEN 1000 AND 1500;
```

Использование оператора IN

Используйте оператор IN для проверки вхождения значения в список.

```
SQL> SELECT empno, ename, sal, mgr
  2  FROM   emp
  3  WHERE  mgr IN (7902, 7566, 7788);
```

```
     EMPNO ENAME                SAL        MGR
---------- ---------- ---------- ----------
      7902 FORD                3000       7566
      7369 SMITH                800       7902
      7788 SCOTT               3000       7566
      7876 ADAMS               1100       7788
```

Использование оператора LIKE

- Используйте оператор LIKE для поиска по шаблону.
- Условие поиска может содержать как символы, так и числа.
  - % обозначает ноль или более символов
  - _ обозначает один символ

```
SQL> SELECT ename
  2  FROM   emp
  3  WHERE  ename LIKE 'S%';
```

Использование оператора LIKE

Символы шаблона можно совмещать.

```
SQL> SELECT ename
  2  FROM   emp
  3  WHERE  ename LIKE '_A%';
```

```
ENAME
----------
JAMES
WARD
```


Использование оператора LIKE

- Вы можете использовать идентификатор ESCAPE для поиска символа "%" и "_“.

```
… WHERE Columns LIKE ‘%/%’ ESCAPE ‘/’
```

Поиск имени «KING»

```
SELECT * FROM emp WHERE ename LIKE ‘KI%’
```

```
SELECT * FROM emp WHERE ename LIKE ‘KI%’  ESCAPE ‘/’
```

```
SELECT * FROM emp WHERE ename LIKE ‘KI%’  ESCAPE ‘I’
```

```
SELECT * FROM emp WHERE ename LIKE ‘KII%’ ESCAPE ‘I’
```


Использование оператора IS NULL

Оператор IS NULL используется для проверки, совпадает ли значение с null

```
SQL> SELECT   ename, mgr
  2  FROM     emp
  3  WHERE    mgr IS NULL;
```

```
ENAME          MGR
---------- ---------
KING
```


Логические операторы

Оператор	Значение
AND	Возвращает TRUE если *оба* операнда имеют значение TRUE
OR	Возвращает TRUE если *любой* из операндов имеет значение TRUE
NOT	Возвращает TRUE если операнд имеет значение FALSE

Использование оператора AND

- AND требует, чтобы оба операнда имели значение TRUE.

```
SQL> SELECT empno, ename, job, sal
  2  FROM   emp
  3  WHERE  sal>=1100
  4  AND    job='CLERK';
```

```
     EMPNO ENAME      JOB              SAL
---------- ---------- --------- ----------
      7876 ADAMS      CLERK           1100
      7934 MILLER     CLERK           1300
```


Использование оператора OR

- OR требует, чтобы хотя бы один из операндов имел значение TRUE.

```
SQL> SELECT empno, ename, job, sal
  2  FROM   emp
  3  WHERE  sal>=1100
  4  OR     job='CLERK';
```

```
     EMPNO ENAME      JOB              SAL
---------- ---------- --------- ----------

      7839 KING       PRESIDENT        5000
      7698 BLAKE      MANAGER          2850
      7782 CLARK      MANAGER          2450
      7566 JONES      MANAGER          2975
      7654 MARTIN     SALESMAN         1250
...
14 rows selected.
```

Использование оператора NOT

```
SQL> SELECT ename, job
  2  FROM   emp
  3  WHERE  job NOT IN ('CLERK','MANAGER','ANALYST');
```

```
ENAME      JOB
---------- ---------
KING       PRESIDENT
MARTIN     SALESMAN
ALLEN      SALESMAN
TURNER     SALESMAN
WARD       SALESMAN
```

AND, OR, NOT

AND	TRUE	FALSE	NULL
TRUE	TRUE	FALSE	NULL
FALSE	FALSE	FALSE	FALSE
NULL	NULL	FALSE	NULL

OR	TRUE	FALSE	NULL
TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	FALSE	NULL
NULL	TRUE	NULL	NULL

NOT	TRUE	FALSE	NULL
	FALSE	TRUE	NULL

Приоритеты

Порядок выполнения	
1	Арифметические операции
2	Конкатенация (||)
3	Сравнение (>,>=, =, …)
4	IS [NOT] NULL, LIKE, [NOT] IN
5	[NOT] BETWEEN
6	!=
7	NOT
8	AND
9	OR

Порядком вычисления можно управлять с помощью скобок.

Приоритеты

```
SQL> SELECT ename, job, sal
  2  FROM   emp
  3  WHERE  job='SALESMAN'
  4  OR     job='PRESIDENT'
  5  AND    sal>1500;
```

ENAME	JOB	SAL
KING	PRESIDENT	5000
MARTIN	SALESMAN	1250
ALLEN	SALESMAN	1600
TURNER	SALESMAN	1500
WARD	SALESMAN	1250

Приоритеты

- Используйте скобки для указания приоритета вычисления.

```
SQL> SELECT   ename, job, sal
  2  FROM     emp
  3  WHERE    (job='SALESMAN'
  4  OR       job='PRESIDENT')
  5  AND      sal>1500;
```

```
ENAME      JOB              SAL
---------- --------- ---------
KING       PRESIDENT       5000
ALLEN      SALESMAN        1600
```

Предложение ORDER BY

- Предложение ORDER BY сортирует данные.
  - ASC: по возрастанию, по умолчанию
  - DESC: по убыванию
- Предложение ORDER BY идет последним в выражении SELECT.

```
SELECT      [DISTINCT] {*, column [alias], ...}
FROM        table
[WHERE      condition(s)]
[ORDER BY   {column, expr, alias} [ASC|DESC]];
```

Сортировка по убыванию

```
SQL> SELECT        ename, job, deptno, hiredate
  2  FROM          emp
  3  ORDER BY hiredate DESC;
```

```
ENAME      JOB              DEPTNO HIREDATE
---------- --------- ---------- ---------
ADAMS      CLERK            20 12-JAN-83
SCOTT      ANALYST          20 09-DEC-82
MILLER     CLERK            10 23-JAN-82
JAMES      CLERK            30 03-DEC-81
FORD       ANALYST          20 03-DEC-81
KING       PRESIDENT        10 17-NOV-81
MARTIN     SALESMAN         30 28-SEP-81
...
14 rows selected.
```

Сортировка по псевдониму

```
SQL> SELECT   empno, ename, sal*12 annsal
  2  FROM     emp
  3  ORDER BY annsal;
```

```
     EMPNO ENAME          ANNSAL
---------- ---------- ---------
      7369 SMITH           9600
      7900 JAMES          11400
      7876 ADAMS          13200
      7654 MARTIN         15000
      7521 WARD           15000
      7934 MILLER         15600
      7844 TURNER         18000
...
14 rows selected.
```

Сортировка по нескольким столбцам

- Порядок сортировки задается списком в предложении ORDER BY.
- Вы можете сортировать по столбцу, не входящему в список выборки выражения SELECT.

```
SQL> SELECT  ename, deptno, sal
  2  FROM    emp
  3  ORDER BY      deptno, sal DESC;
```

```
ENAME          DEPTNO        SAL
---------- ---------- ----------
KING               10       5000
CLARK              10       2450
MILLER             10       1300
FORD               20       3000
...
14 rows selected.
```

Сортировка

1

```
SQL> SELECT  ename, deptno, sal
  2  FROM    emp
  3  ORDER BY deptno ASC, sal DESC;
```

2

```
SQL> SELECT  ename, deptno, sal
  2  FROM    emp
  3  ORDER BY 12*sal;
```

3

```
SQL> SELECT  ename, deptno, sal annual_sal
  2  FROM    emp
  3  ORDER BY annual_sal;
```

4

```
SQL> SELECT  ename, deptno, sal annual_sal
  2  FROM    emp
  3  ORDER BY 1+1;
```

5

```
SQL> SELECT  ename, deptno, sal annual_sal
  2  FROM    emp
  3  ORDER BY 2;
```

Часть 2. Вложенные запросы, объединения

NetCracker®

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).

Подзапросы

- Подзапросы можно использовать:
  - SELECT:
    - Предложение SELECT
    - Предложение FROM
    - Предложение WHERE
    - Предложение HAVING
  - CREATE VIEW
  - CREATE TABLE
  - ….

Использование подзапросов

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

2975

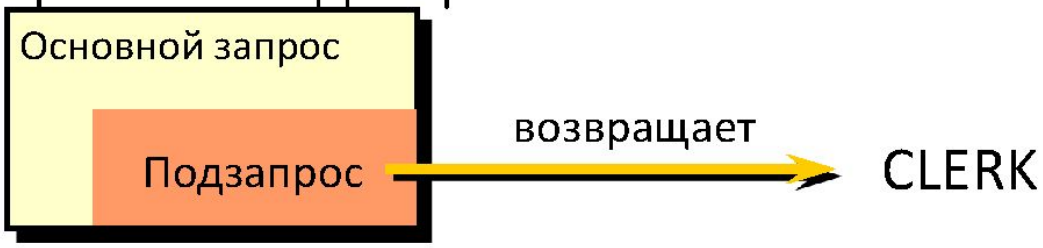
```
ENAME
----------
KING
FORD
SCOTT
```

Советы по использованию подзапросов

- Заключайте 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);
```

CLERK

1100

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

Многострочные подзапросы

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

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

Использование оператора ANY в многострочных подзапросах

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

1300
1100
800
950

```
     EMPNO ENAME      JOB
---------- ---------- ---------
      7654 MARTIN     SALESMAN
      7521 WARD       SALESMAN
```

Использование оператора ALL в многострочных подзапросах

```
SELECT empno, ename, job, sal
FROM   emp
WHERE  sal < ALL
                    (SELECT sal
                     FROM   empl
                     WHERE  job = 'CLERK')
AND    job <> 'CLERK';
```

9000, 6000, 4200

Значения null в подзапросах

```
SQL> SELECT employee.ename
  2  FROM   emp employee
  3  WHERE  employee.empno NOT IN
  4                        (SELECT manager.mgr
  5                         FROM   emp manager);
no rows selected.
```

NOT IN	!=ALL
IN	=ANY

Использование подзапросов в предложении FROM

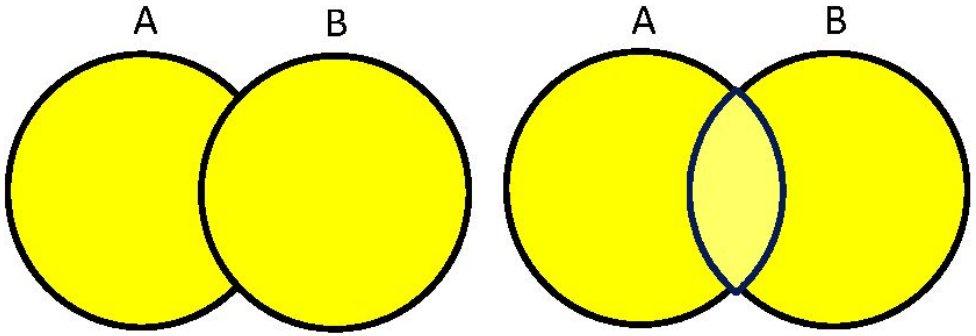
```
SQL> SELECT ename, sal
2  FROM (SELECT * FROM EMP ORDER BY sel)
3  WHERE rownum > 10;
```

```
SQL> SELECT ename, sal
2  FROM (SELECT * FROM EMP WHERE depno = 20)
3  WHRER sal > 1000
```

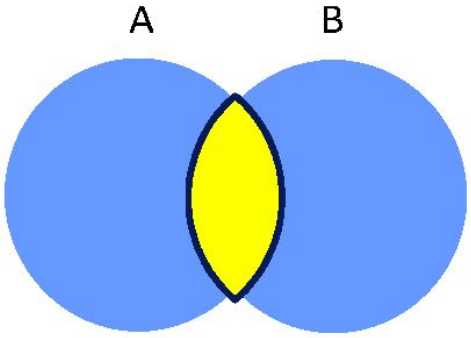
Использование подзапросов в предложении SELECT

```
SQL> SELECT   ename, (SELECT SYSDATE FROM DUAL)
2         FROM emp;
```

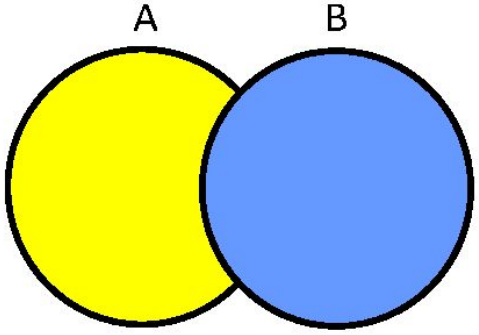

Операции над множествами



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 скобки.
- `ORDER BY` может быть использован только в самом конце выражения

Особенности Oracle Server

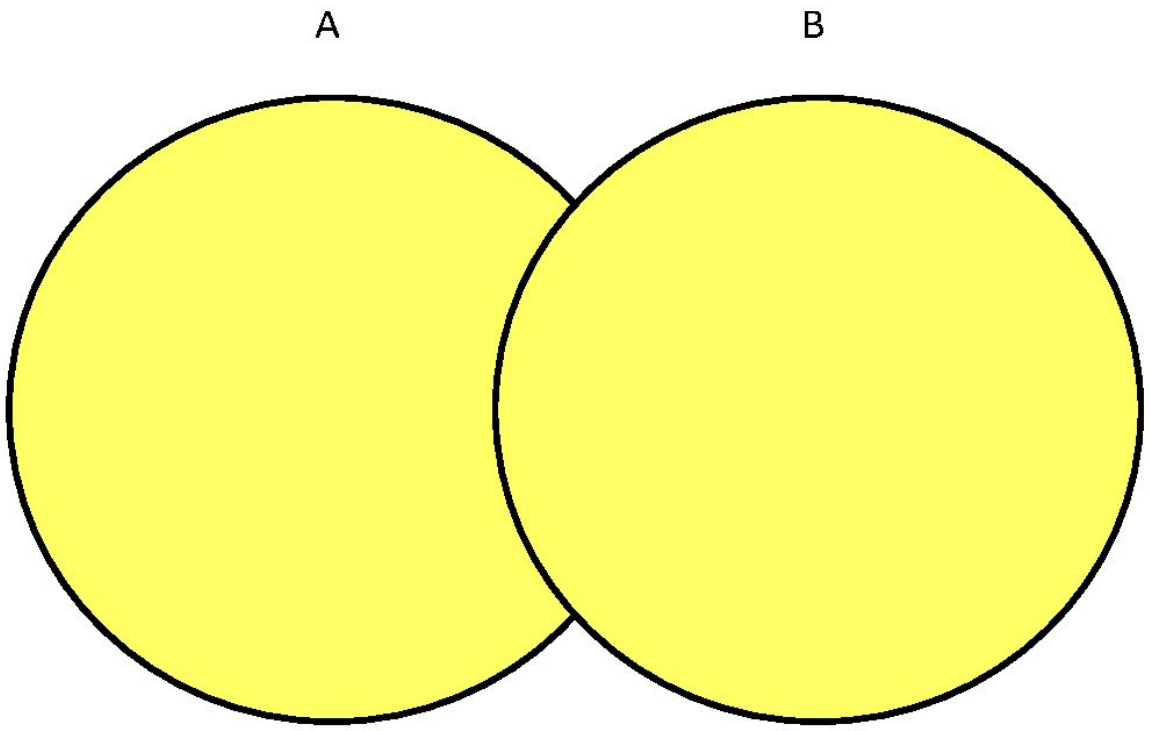
- Повторяющиеся строки автоматически исключаются, за исключением операции `UNION ALL`.
- Имена столбцов первого запроса становятся именами столбцов результирующего запроса.
- По умолчанию результаты отсортированы по возрастанию, за исключением `UNION ALL`.

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 текущей.

UNION

- Оператор UNION вернет строки из обоих таблиц, за исключением повторяющихся.



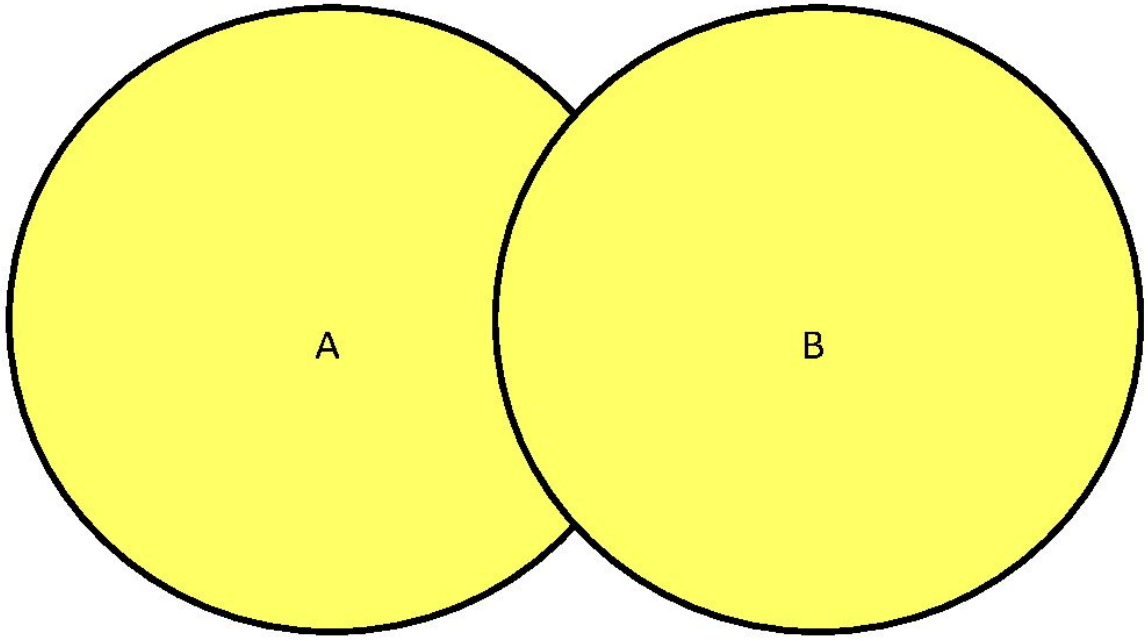
UNION

- 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 обоих таблиц, включая повторяющиеся.



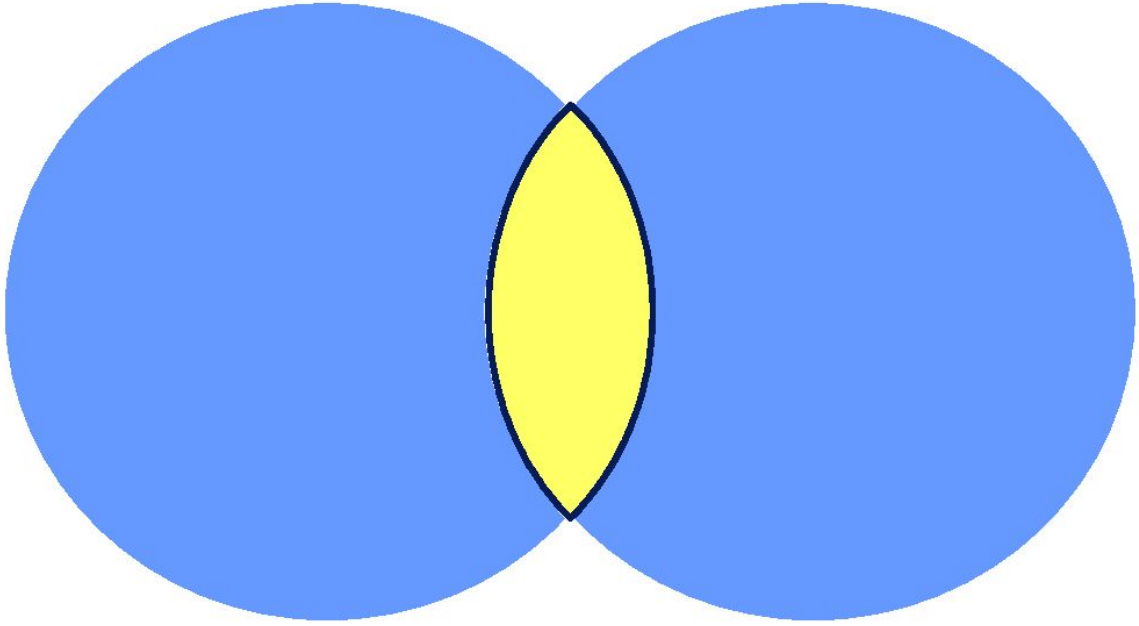
UNION ALL

- 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 A B



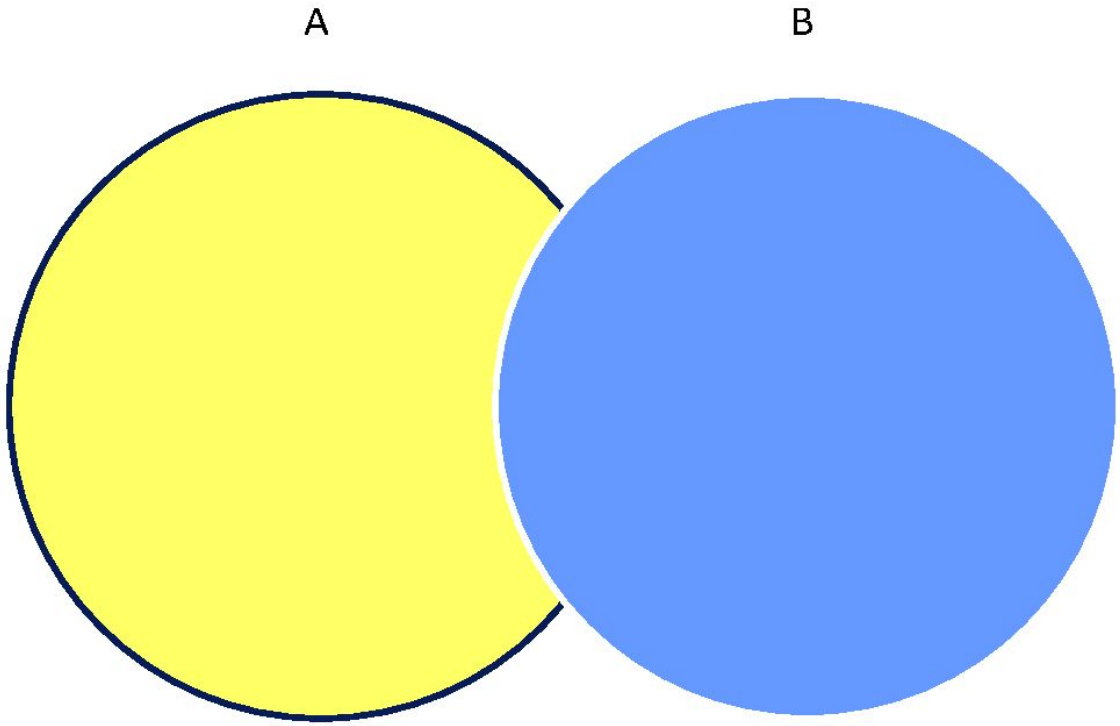
Using the INTERSECT Operator

- 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 можествами
  - 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;
```

Использование предложения ORDER BY в запросах с операциями на множествах

- Предложение ORDER BY может быть использовано только раз, в конце запроса.
- Отдельные подзапросы внутри запроса с операциями на множества не могут использовать предложение ORDER BY.
- В предложении ORDER BY используют колонки (или их псевдонимы) первого SELECTа
- По умолчанию результаты запроса отсортированы по первому столбцу по возрастанию.

Пример

```
SQL> SELECT a as aa,   b   bb
2   FROM tbl1
3   ORDER by a
4   UNION
5   SELECT c, d
6   FROM tbl2
7   UNION ALL
8   SELECT e, f
9   FROM tbl3
10  ORDER by          a DESC,   bb
```