UNION queries

The UNION statement is an algebraic operator that combines the results of two or more queries into a single table. The result-set will contain rows which bare a resemble to both the initial queries.

UNION – INTERSECT - EXCEPT

Select the state and city of each author, publisher and store. The state column of the results table should be in ascending order.

```
SELECT state, city
FROM authors
UNION
SELECT state, city
FROM publishers
UNION
SELECT state, city
FROM stores

ORDER BY state, city
```

1. Select each states where there are authors and publishers.
2. Select each states where there are authors but no publishers.
3. Select the names and ids of publishers who have published “business” books or “psychology” books.
4. Select the names and ids of publishers who have published “business” books and “psychology” books.
5. Select the names and ids of publishers who have published “business” books but not any “psychology” book.

Subqueries

A subquery (nested query) is a SELECT statement embedded in another SELECT, INSERT, UPDATE, or DELETE statement.

The parent statement requires the child's results before it can evaluate its own query; therefore, subqueries must be evaluated before their parent query.

Examples :

1. What are the most expensive books (title and price)?

```
SELECT title AS 'Most expensive book',
price
FROM titles
WHERE price= ( SELECT MAX(price)
               FROM titles     )
```

Solution using TOP :

```
SELECT TOP 1 title, price
FROM titles
ORDER BY price DESC
```

2. What are the least expensive books (title and price)?

```
SELECT title AS 'Least expensive book',
price
FROM titles
WHERE price= ( SELECT MIN(price)
               FROM titles     )
```

Solution using TOP :

```
SELECT TOP 1 title, price
FROM titles
ORDER BY price ASC
```

Note: The solutions using the TOP predicate **are not satisfactory for answering example questions 1 or 2.** TOP 1 only returns a single book, while we needed to find the set (it may be a tie) of books with the most / least cost.

3. Select the books that have a greater price than the average price of all books.

```
SELECT title AS 'Books', price
FROM titles
WHERE  price > (    SELECT AVG(price)
                    FROM titles )
```

4. Select the genre (type) of books whose average price is greater than the average price of all books.

```
SELECT type, AVG(price) AS 'Average price'
FROM titles
GROUP BY type
HAVING AVG(price) >   ( SELECT AVG(price)
                        FROM titles )
```

Using Subqueries with IN

Ex1.

Select the names of all publishers with an advance greater than $10,000.

```
SELECT pub_id, pub_name
FROM publishers
WHERE    pub_id IN
         (
         SELECT pub_id
         FROM titles
         WHERE advance >10000
         )
```

Select the same result using a join :

```
SELECT DISTINCT p.pub_id, p.pub_name
FROM publishers p JOIN titles t ON p.pub_id=t.pub_id
WHERE t.advance>10000
```

```
SELECT DISTINCT p.pub_id, p.pub_name  FROM
publishers p , titles t
WHERE p.pub_id=t.pub_id AND t.advance>10000
```

Using Subqueries with IN

Ex2.

Select the names of all publishers who have published a books in the "Business" or "psychology" genre (type).
Show 2 ways to express the query (subquery and join).

Using a subquery :

```
SELECT pub_id, pub_name
FROM publishers
WHERE   pub_id IN
        (
        SELECT pub_id  FROM titles
        WHERE type IN ('Business' , 'psychology')
        )
```

Using Subqueries with IN

Ex2.

Select the list of names of all publishers who published books in the "Business" or "psychology" genre (type).

Give 2 ways to formulate the query (subquery and join).

```
SELECT DISTINCT p.pub_id, p.pub_name

FROM publishers p, titles t

WHERE p.pub_id=t.pub_id AND t.type IN ('Business' , 'psychology')
```

```
SELECT DISTINCT p.pub_id, p.pub_name
FROM publishers p JOIN titles t ON p.pub_id=t.pub_id
WHERE t.type IN ('Business' , 'psychology')
```

Using Subqueries with IN

Ex3.

Select all advance payments made by the publisher "Binnet & Hardley", and the quantity of books each of these payments was made for.

Using a subquery :

```
SELECT advance, count(*) AS NbBooks
FROM titles
WHERE pub_id  IN
               (
               SELECT pub_id  FROM publishers
               WHERE pub_name='Binnet & Hardley'
               )
GROUP BY advance
```

Subqueries used with IN

Ex3.

Select all advance payments made by the publisher "Binnet & Hardley", and the quantity of books each of these payments was made for.

Using a join :

```
SELECT advance, COUNT(*) AS NbBooks
FROM titles t JOIN publishers p ON t.pub_id=p.pub_id
WHERE pub_name='Binnet & Hardley'
GROUP BY advance
```

```
SELECT advance, COUNT(*) AS NbBooks
FROM titles t , publishers p
WHERE t.pub_id=p.pub_id AND pub_name='Binnet & Hardley'
GROUP BY advance
```

Subqueries used with IN

Ex4.

Select the titles of books published by “Algodata infosystems”.

Using a subquery :

```
SELECT title
FROM titles
WHERE pub_id  IN
              (
              SELECT pub_id
              FROM publishers
              WHERE pub_name='Algodata infosystems'
              )
```

Subqueries used with IN

Ex4.

Select the titles of books published by “Algodata infosystems”.

Using a join :

```
SELECT title
FROM titles t JOIN publishers p ON t.pub_id=p.pub_id
WHERE pub_name='Algodata infosystems'
```

```
SELECT title
FROM titles t , publishers p
WHERE t.pub_id=p.pub_id  AND pub_name='Algodata infosystems'
```

Using ANY and ALL

Select the names and postal codes (zip) of authors who have a postal code greater than **at least one (1)** of the postal codes of authors by the last name of “White” or “Ringer”.

Using a subquery :

```
SELECT au_lname, zip
FROM authors
WHERE zip > ANY
                    (
                    SELECT DISTINCT zip
                    FROM authors
                    WHERE au_lname='White' OR au_lname='Ringer'
                    )
```

Using ANY and ALL

Select the zip codes of authors who have a postal code greater than **all** postal codes of authors with the name "White" or "Ringer".

Using a subquery :

```
SELECT au_lname, zip
FROM authors
WHERE zip > ALL
                    (
                    SELECT DISTINCT zip
                    FROM authors
                    WHERE au_lname='White' OR au_lname='Ringer'
                    )
```

Using EXISTS

Select the names of authors who live in the same city as a publishers.

```
SELECT au_lname, au_fname, city, state
FROM authors a
WHERE  EXISTS (
              SELECT *
              FROM publishers p
              WHERE a.city=p.city
              )
```

Using EXISTS

Select the names and publisher numbers of all publishers who have published psychology books.

```
SELECT pub_id, pub_name
FROM publishers p
WHERE EXISTS    (
                SELECT *
                FROM titles t
                WHERE p.pub_id=t.pub_id AND t.type='psychology'
                )
```

Using EXISTS

Select the names and publisher numbers of publishers who have published traditional or modern cuisine books.

```
SELECT pub_id,pub_name
FROM publishers p
WHERE EXISTS  (
               SELECT *
               FROM titles t
               WHERE p.pub_id=t.pub_id
                    AND (t.type='mod_cook' OR t.type='trad_cook')
)
```

Exercise:

Select the titles of all books which sold no copies in 1993.

```
SELECT *
FROM titles t
WHERE NOT EXISTS (
                    SELECT *
                    FROM sales s
                    WHERE s.title_id=t.title_id
                          AND YEAR(s.ord_date)=1993
                   )
```

Exercises:

1. Select the authors who have not had any books published by "Algodata Infosystems". In addition to the full name of the author, get the amount of each advance and the advance's corresponding book number and publisher.
2. Select the authors who have had all their books published by "Algodata Infosystems". In addition to the full name of the author, get the amount of each advance and their corresponding book number.
3. Select the names of authors who received an advance greater than all advances paid by the publisher "Algodata Infosystems". In addition to the full name of the author, get the amount of each advance, the corresponding book number, and publisher.