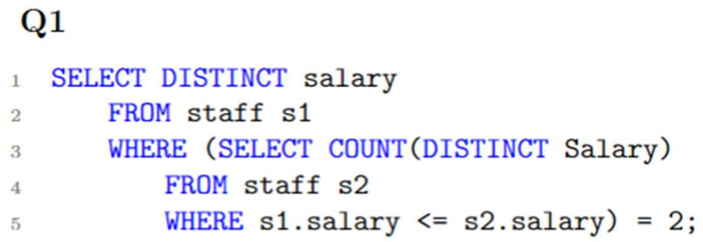
Code to Query

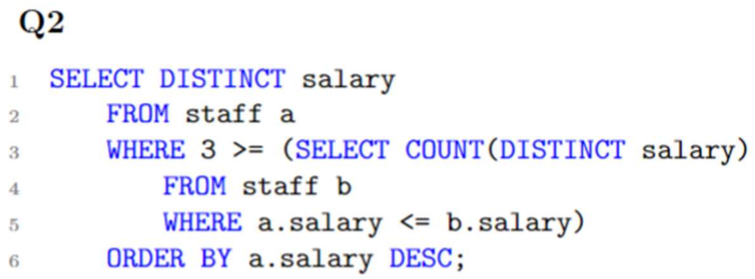
By Little Butterflies



Q1

```sql
SELECT DISTINCT salary
    FROM staff s1
    WHERE (SELECT COUNT(DISTINCT Salary)
        FROM staff s2
        WHERE s1.salary <= s2.salary) = 2;
```

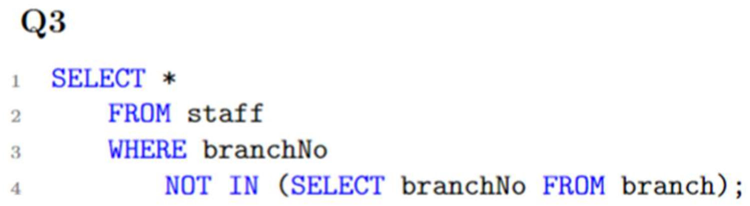
Shows all different salaries in the staff table that has exactly 2 different salaries greater than or equal to it. In other words, it shows the second biggest salary.



Q2

```sql
SELECT DISTINCT salary
    FROM staff a
    WHERE 3 >= (SELECT COUNT(DISTINCT salary)
        FROM staff b
        WHERE a.salary <= b.salary)
    ORDER BY a.salary DESC;
```

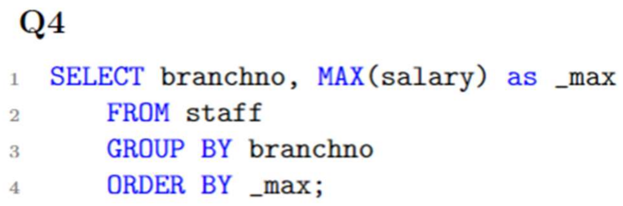
Shows all different salaries in the staff table that has 3 or fewer different salaries greater than or equal to it in a descending order. In other words, it shows the 3 biggest salaries.



Q3

```sql
SELECT *
    FROM staff
    WHERE branchNo
        NOT IN (SELECT branchNo FROM branch);
```

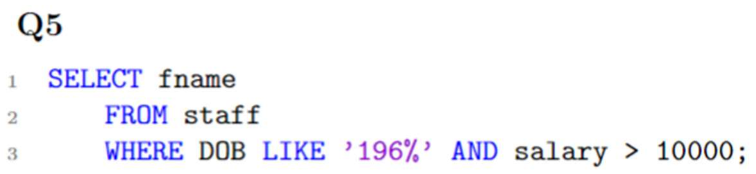
Shows all rows in the staff table where the branchNo is not connected to any of the branches in the branch table. In other words, it shows everything about every staff personnel who doesn't have a valid branchNo. (valid = connected to a branch)



Q4

```sql
SELECT branchno, MAX(salary) as _max
    FROM staff
    GROUP BY branchno
    ORDER BY _max;
```

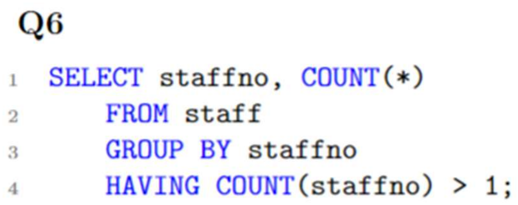
Shows all branchNos and the biggest salary at that branch as _max in the staff table. Every branch is written exactly once, and they are ordered in an ascending order by the biggest salaries.



Q5

```sql
SELECT fname
    FROM staff
    WHERE DOB LIKE ’196%’ AND salary > 10000;
```

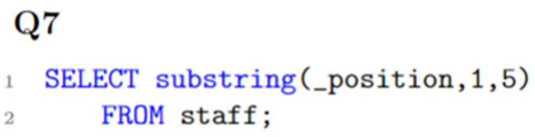
Shows all fnames in the staff table, where the person was born in the 1960s and their salary is greater than 10000.



Q6

```sql
SELECT staffno, COUNT(*)
    FROM staff
    GROUP BY staffno
    HAVING COUNT(staffno) > 1;
```

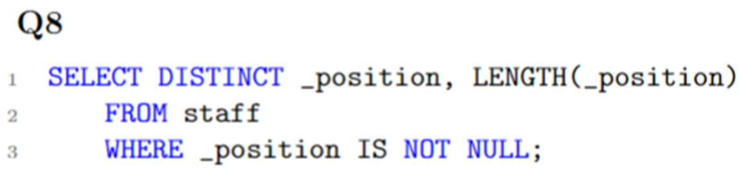
Shows all different staffno values in the staff table, and the count of staff personnel with that staffno, if the count of them is greater than 1.



Q7

```sql
SELECT substring(_position,1,5)
    FROM staff;
```

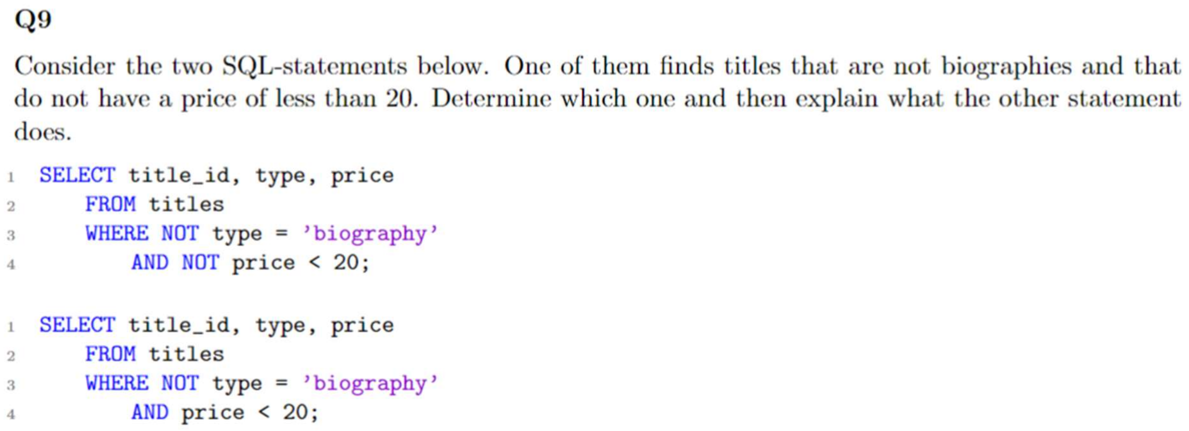
Shows the first 5 characters of the _position column for every row in the staff table.



Q8

```sql
SELECT DISTINCT _position, LENGTH(_position)
    FROM staff
    WHERE _position IS NOT NULL;
```

Shows all different _positions and their length in the staff table, where their value is not null.



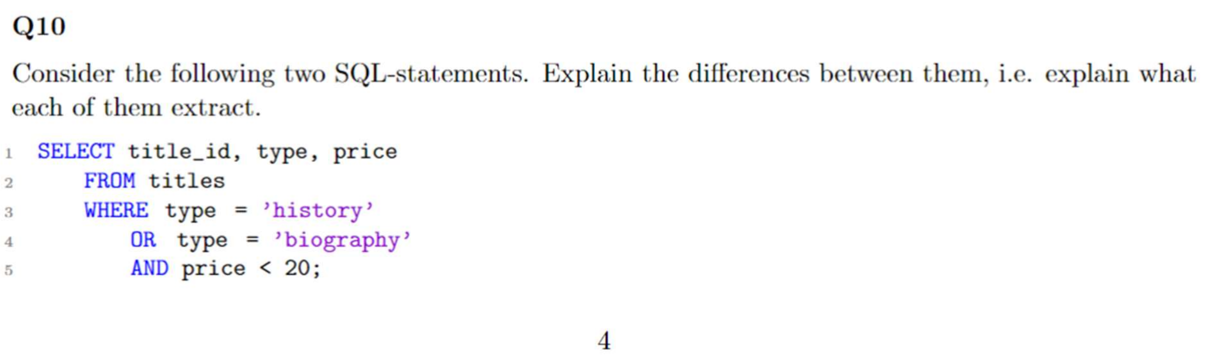
Q9

Consider the two SQL-statements below. One of them finds titles that are not biographies and that do not have a price of less than 20. Determine which one and then explain what the other statement does.

```sql
SELECT title_id, type, price
    FROM titles
    WHERE NOT type = ’biography’
        AND NOT price < 20;
```

```sql
SELECT title_id, type, price
    FROM titles
    WHERE NOT type = ’biography’
        AND price < 20;
```

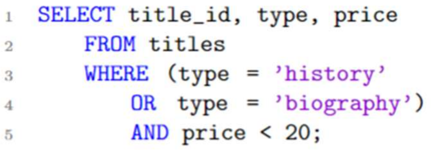
The first one. The second one shows titles that are not biographies and that have a price of less than 20.



Q10

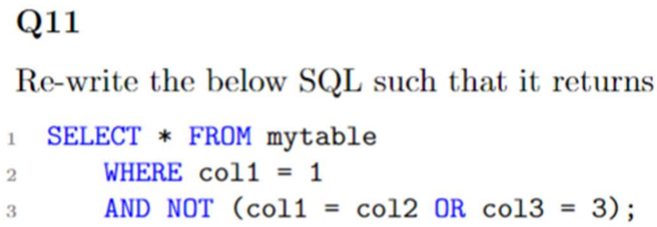
Consider the following two SQL-statements. Explain the differences between them, i.e. explain what each of them extract.

```
SELECT title_id, type, price
   FROM titles
   WHERE type = ’history’
      OR type = ’biography’
      AND price < 20;
```



```
SELECT title_id, type, price
   FROM titles
   WHERE (type = ’history’
      OR type = ’biography’)
      AND price < 20;
```

The first one shows title_id, type and price for every title that is type of history and their price doesn’t matter or type of biography and their price is less than 20. The second one shows title_id, type and price for every title that is type of history or biography, and their price is less than 20. For example, a history for 22 would return in the first one, but not in the second.



Q11

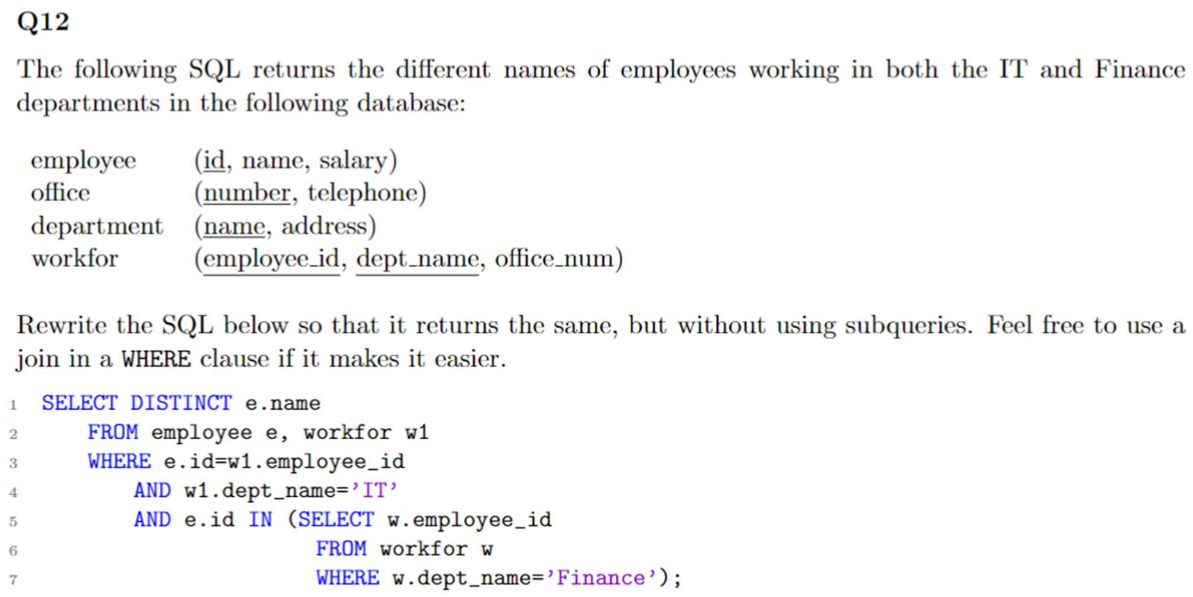
Re-write the below SQL such that it returns the same but without using NOT.

```
SELECT * FROM mytable
   WHERE col1 = 1
   AND NOT (col1 = col2 OR col3 = 3);
```

```
SELECT * FROM mytable

        WHERE col = 1

        AND (col1 != col 2 AND col3 != 3);
```



Q12

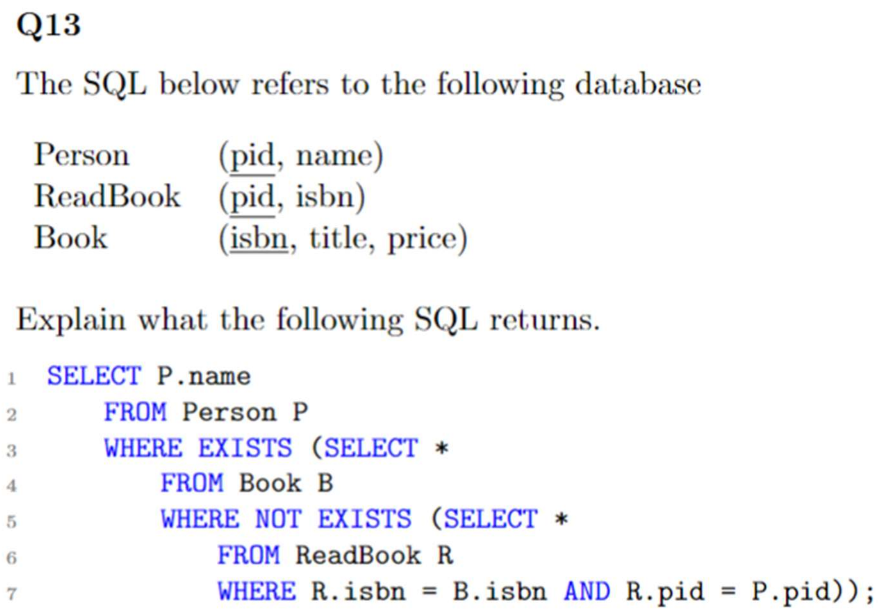
The following SQL returns the different names of employees working in both the IT and Finance departments in the following database:

employee	(id, name, salary)
office	(number, telephone)
department	(name, address)
workfor	(employee_id, dept_name, office_num)

Rewrite the SQL below so that it returns the same, but without using subqueries. Feel free to use a join in a WHERE clause if it makes it easier.

```
SELECT DISTINCT e.name
   FROM employee e, workfor w1
   WHERE e.id=w1.employee_id
      AND w1.dept_name='IT'
      AND e.id IN (SELECT w.employee_id
                      FROM workfor w
                      WHERE w.dept_name='Finance');
```

I honestly don't know how ☹



Q13

The SQL below refers to the following database

Person	(pid, name)
ReadBook	(pid, isbn)
Book	(isbn, title, price)

Explain what the following SQL returns.

```
SELECT P.name
   FROM Person P
   WHERE EXISTS (SELECT *
      FROM Book B
      WHERE NOT EXISTS (SELECT *
         FROM ReadBook R
         WHERE R.isbn = B.isbn AND R.pid = P.pid));
```

Shows every Person for there is a book from the Book table that is not present in the ReadBook table for that person's own id. The ReadBook and Book are connected by isbn. In other words, shows every person that had not read every book.