

Υποερωτήματα SQL

Παραδείγματα και εφαρμογές από τη βάση δεδομένων company

Αθανάσιος Σταυρακούδης

<http://stavrakoudis.econ.uoi.gr>

Άνοιξη 2014



Περιεχόμενα

- 1 Απλά υποερωτήματα
- 2 Τεστ: κάθε, τουλάχιστον, ύπαρξης
- 3 Επιπλέον Υποερωτήματα
- 4 Ένωση
- 5 Ασκήσεις επανάληψης



Η ανάγκη για υποερώτημα

Ποιος υπάλληλος παίρνει το μεγαλύτερο μισθό;

Αν ξέραμε το μεγαλύτερο μισθό, πχ 2787.69, θα γράφαμε:

```
1 SELECT empid, lastname
2     FROM employees
3     WHERE salary = 2787.69;
```

Ο μεγαλύτερος μισθός μπορεί να βρεθεί με:

```
1 SELECT MAX(salary)
2     FROM employees;
```



Το πρόβλημα με τα δύο ερωτήματα

- Στην ερώτηση ποιος υπάλληλος παίρνει το μεγαλύτερο μισθό, πρέπει πρώτα να βρούμε το μεγαλύτερο μισθό
- Δηλαδή να απαντήσουμε σε δύο ερωτήματα και μάλιστα με σωστή σειρά:
 - 1 Ποιος είναι ο μεγαλύτερος μισθός: **MAX(salary)**
 - 2 Ποιος υπάλληλος έχει τόσο μισθό; **WHERE**
- Η τιμή που προκύπτει από το πρώτο ερώτημα δεν μπορεί να εισαχθεί αυτόματα στο δεύτερο
- Στα λογιστικά φύλλα, μπορούμε να απαντήσουμε τέτοια ερωτήματα με συνδυασμό **INDEX MATCH MAX**
- Στην **SQL** το πρόβλημα λύνεται εύκολα με **υποερώτημα** εμφώλευση **SELECT**



Ένα απλό υποερώτημα

Να βρεθεί ο υπάλληλος με το μεγαλύτερο μισθό:

```
1  SELECT empid, salary
2      FROM employees
3      WHERE salary = (SELECT MAX(salary)
4                      FROM employees);
```

- Η λύση λοιπόν είναι η εμφώλευση **SELECT**
- Σε αντίθεση με τα λογιστικά φύλλα, το ερώτημα θα δώσει ορθή απάντηση στην περίπτωση που δύο ή περισσότεροι υπάλληλοι έχουν τον ίδιο μισθό
- Το υποερώτημα **χρειάζεται παρενθέσεις**
- Αυτό είναι ένα παράδειγμα υποερωτήματος ως βαθμωτής (**scalar**) μεταβλητής



Παρατηρήσεις για τα υποερωτήματα

- Το υποερώτημα πρέπει να είναι ένα **αυτοτελές** ερώτημα
- Υποερωτήματα μπορούν να χρησιμοποιηθούν τόσο στη φράση **WHERE** όσο και στη φράση **HAVING**
- Στις περισσότερες των περιπτώσεων, η φράση **SELECT** στο υποερώτημα ακολουθείται ένα μόνο πεδίο και μάλιστα **κλειδί**
- Το υποερώτημα **δεν μπορεί** (και δεν έχει νόημα) να περιέχει τη φράση **ORDER BY**
- Το υποερώτημα μπορεί να **αναφέρεται** σε πεδία που υπάρχουν στον πίνακα του **εξωτερικού ερωτήματος**, αλλά γενικά πρέπει να έχουν τη δική τους φράση **FROM**



Υποερώτημα ή αυτοσύζευξη;

Να βρεθεί ο κωδικός και το επώνυμο των υπαλλήλων που εργάζονται στο ίδιο τμήμα με αυτό που εργάζεται ο υπάλληλος με κωδικό 102.

Με αυτοσύζευξη

```
1 SELECT e1.empid, e1.salary
2   FROM employees e1, employees e2
3  WHERE e1.depid = e2.depid
4        AND e2.empid = 102;
```

Με υποερώτημα

```
1 SELECT empid, salary
2   FROM employees
3  WHERE depid = (SELECT depid
4                  FROM employees
5                  WHERE empid = 102);
```



Υποερώτημα μετά το HAVING

Να βρεθεί το πλήθος των υπαλλήλων ανά όνομα τμήματος, για εκείνα τα τμήματα στα οποία εργάζονται περισσότεροι από 3/2 των υπαλλήλων που απασχολούνται στο έργο 43

```
1  SELECT d.depname, COUNT(*)  
2      FROM employees e INNER JOIN departments d  
3          ON e.depid = d.depid  
4  GROUP BY d.depname  
5  HAVING COUNT(*) >
```



Υποερώτημα μετά το HAVING

Να βρεθεί το πλήθος των υπαλλήλων ανά όνομα τμήματος, για εκείνα τα τμήματα στα οποία εργάζονται περισσότεροι από $3/2$ των υπαλλήλων που απασχολούνται στο έργο 43

```
1  SELECT d.depname, COUNT(*)
2      FROM employees e INNER JOIN departments d
3          ON e.depid = d.depid
4  GROUP BY d.depname
5      HAVING COUNT(*) > 1.5*(SELECT COUNT(empid)
6          FROM workson
7          WHERE proid = 43);
```



Υποερώτημα μετά το HAVING

Να βρεθεί το πλήθος των υπαλλήλων ανά όνομα τμήματος, για εκείνα τα τμήματα στα οποία εργάζονται περισσότεροι από 3/2 των υπαλλήλων που απασχολούνται στο έργο 43

```
1 SELECT d.depname, COUNT(*)
2 FROM employees e INNER JOIN departments d
3 ON e.depid = d.depid
4 GROUP BY d.depname
5 HAVING COUNT(*) > 1.5*(SELECT COUNT(empid)
6 FROM workson
7 WHERE proid = 43);
```

| depname | COUNT(*) |
|-----------------------|----------|
| ----- | |
| Επιστημόνων/Μηχανικών | 9 |
| Μάνατζμεντ/Πωλήσεων | 8 |

Στο έργο 43 απασχολούνται 5 υπάλληλοι.



Δύο υποερωτήματα

Να βρεθούν όλες οι λεπτομέρειες των υπαλλήλων που προσλήφθηκαν ανάμεσα στην πιο παλιά και στην πρόσφατη ημερομηνία έναρξης των έργων, και δεν εργάζονται στο τμήμα 6

```
1  SELECT *  
2      FROM employees  
3      WHERE hiredate BETWEEN ( SELECT MIN(startdate)  
4                               FROM projects)  
5                               AND ( SELECT MAX(startdate)  
6                                     FROM projects)  
7      AND depid != 6;
```



Υποερώτημα με τελεστή συνόλου IN

Να βρεθεί το όνομα όλων των τμημάτων στα οποία εργάζονται υπάλληλοι, οι οποίοι αν πάρουν αύξηση κατά 5% ο μισθός τους θα ανέβει περισσότερο από 100 €

```
1  SELECT depname
2      FROM departments
3      WHERE depid IN (SELECT DISTINCT depid
4                      FROM employees
5                      WHERE salary*0.05 > 100);
```

- Η φράση **FROM** στο **υποερώτημα** μπορεί να έχει έχει διαφορετικό πίνακα από τη φράση **FROM** του **εξωτερικού** υποερωτήματος
- Η φράση **SELECT** του **υποερωτήματος** ακολουθείται από **ένα μόνο** πεδίο
- Στο υποερώτημα, η φράση **DISTINCT** μπορεί να παραληφθεί (με ποινή στην ταχύτητα)



Ισοδυναμία με σύζευξη

Να βρεθεί το όνομα όλων των τμημάτων στα οποία εργάζονται υπάλληλοι, οι οποίοι αν πάρουν αύξηση κατά 5% ο μισθός τους θα ανέβει περισσότερο από 100 €

```
1 SELECT DISTINCT d.depname
2   FROM employees e INNER JOIN departments d
3     ON e.depid = d.depid
4  WHERE e.salary*0.05 > 100;
```

- Σε μερικές περιπτώσεις, όπως αυτή, η απάντηση μπορεί να δοθεί τόσο με υποερώτημα όσο και με σύζευξη
- Δεν έχει σημασία ποιο τρόπο θα επιλέξετε
- Να ξέρετε και τους δύο τρόπους! Υπάρχουν περιπτώσεις που ένας μόνο είναι σωστός!



Περιεχόμενα

- 1 Απλά υποερωτήματα
- 2 Τεστ: κάθε, τουλάχιστον, ύπαρξης
- 3 Επιπλέον Υποερωτήματα
- 4 Ένωση
- 5 Ασκήσεις επανάληψης



Τεστ σύγκρισης με ALL

Να βρεθούν οι υπάλληλοι που προσλήφθηκαν μετά την έναρξη όλων των έργων

```
1  SELECT *  
2      FROM employees  
3      WHERE hiredate > ALL (SELECT DISTINCT startdate  
4                             FROM projects);
```

- Οι συγκρίσεις με **ALL** αποδίδουν **TRUE**, αν ο τελεσταίος της σύγκρισης αποδίδει **TRUE** με όλες τις εγγραφές που επιστρέφει το υποερώτημα
- Το **ALL** τοποθετείται μετά τον τελεστή σύγκρισης και πριν το υποερώτημα



Ισοδυναμία λόγω βέλους του χρόνου

Να βρεθούν οι υπάλληλοι που προσλήφθηκαν μετά την έναρξη όλων των έργων

```
1  SELECT *  
2      FROM employees  
3      WHERE hiredate > (SELECT MAX(startdate)  
4                          FROM projects);
```

- Τα δύο υποερωτήματα είναι ισοδύναμα
- Αν μια ημερομηνία είναι πιο πρόσφατη από την πιο πρόσφατη έναρξη έργου τότε θα είναι και πιο πρόσφατη από όλες τις άλλες
- Αυτό εφαρμόζεται μόνο σε χρονικά δεδομένα, σε άλλες περιπτώσεις είναι πιθανό η χρήση του τελεστή **ALL** να είναι αναπόφευκτη



Παρατηρήσεις για τα υποερωτήματα με ALL

Για τη χρήση του **ALL**, σε παραστάσεις

WHERE X θ ALL Y

όπου Y υποερώτημα, πρέπει να ξέρουμε:

- Αν το υποερώτημα επιστρέψει το κενό σύνολο τότε η σύγκριση με το **ALL** αποδίδει **TRUE**
- Αν η σύγκριση επιστρέψει **true** για όλες τις όλες τις εγγραφές που επιστρέφει το υποερώτημα, τότε και μόνο τότε, η παράσταση επιστρέφει **TRUE**
- Αν η σύγκριση επιστρέψει **FALSE** για τουλάχιστον μία φορά, τότε η παράσταση επιστρέφει **FALSE**
- Αν για κάποια σύγκριση, επιστραφεί η τιμή **NULL**, τότε η παράσταση επιστρέφει **NULL**



Τεστ σύγκρισης με ANY

Να βρεθούν οι υπάλληλοι που προσλήφθηκαν μέχρι την 31/12/2003 και παίρνουν μισθό μικρότερο από το μισθό οποιουδήποτε υπαλλήλου που προσλήφθηκε μετά την 1/1/2004

```
1  SELECT *
2      FROM employees
3  WHERE hiredate <= '2003-12-31'
4      AND salary < ANY (SELECT DISTINCT salary
5                          FROM employees
6                          WHERE hiredate > '2004-01-01'
7                          AND salary IS NOT NULL);
```

- Το πεδίο *salary* υπάρχει στη φράση **WHERE** του **εξωτερικού** ερωτήματος και στη φράση **SELECT** του **υποερωτήματος**
- Εδώ μας ενδιαφέρει η σύγκριση να αποδώσει τουλάχιστον μία φορά **TRUE**



Παρατηρήσεις για τα υποερωτήματα με ANY

Για τη χρήση του **ANY**, σε παραστάσεις

WHERE X θ ANY Y

όπου Y υποερώτημα, πρέπει να ξέρουμε:

- Αν το υποερώτημα επιστρέψει το κενό σύνολο τότε η σύγκριση με το **ANY** αποδίδει **FALSE**
- Αν η σύγκριση επιστρέψει **true** τουλάχιστον μία φορά, τότε, η παράσταση επιστρέφει **TRUE**
- Αν η σύγκριση επιστρέψει **FALSE** για όλες τις όλες τις εγγραφές που επιστρέφει το υποερώτημα, τότε και μόνο τότε, η παράσταση επιστρέφει **FALSE**
- Αν για κάποια σύγκριση, επιστραφεί η τιμή **NULL**, τότε η παράσταση επιστρέφει **NULL**



Τεστ σύγκρισης με EXISTS

Να βρεθεί ο κωδικός και ονοματεπώνυμο των υπαλλήλων του τμήματος 4 που απασχολούνται σε τουλάχιστον ένα έργο

```
1  SELECT e.empid, e.firstname, e.lastname
2      FROM employees e
3      WHERE depid = 4
4          AND EXISTS (SELECT empid
5                       FROM workson w
6                       WHERE e.empid = w.empid);
```

- Ο τελεστής **EXISTS** λέγεται τελεστής **ύπαρξης**
- Οι εγγραφές του υποερωτήματος **αναφέρονται** στις εγγραφές του εξωτερικού ερωτήματος, τέτοια υποερωτήματα λέγονται **συσχετιζόμενα**
- Το τεστ **EXISTS** δεν επιστρέφει ποτέ **NULL**, μόνο **TRUE** ή **FALSE**



Ανάλυση παραδείγματος τελεστή ύπαρξης

- 1 Αρχικά πρέπει να γνωρίζουμε ότι αν απασχολείται κάποιος υπάλληλος σε κάποιο έργο, τότε θα υπάρχει μια αντίστοιχη εγγραφή στον πίνακα *workson*
- 2 Επομένως, αυτό που ψάχνουμε μπορεί να θεωρηθεί ισοδύναμο με την έκφραση: να βρεθούν οι υπάλληλοι, ο κωδικός των οποίων εμφανίζεται στον πίνακα *workson*
- 3 Η **SQL** εκτελεί το εξωτερικό ερώτημα και βρίσκει τον κωδικό όλων των υπαλλήλων
- 4 Στη συνέχεια, για κάθε εγγραφή του κυρίου ερωτήματος, εκτελεί το υποερώτημα, σύμφωνα με τη σύγκριση:
WHERE e.empid = w.empid
- 5 Αν το υποερώτημα επιστρέψει εγγραφές (αδιάφορο πόσες) τότε το τεστ **EXISTS** επιστρέφει **TRUE**
- 6 Διαφορετικά, αν το υποερώτημα δεν επιστρέψει εγγραφές, το τεστ **EXISTS** επιστρέφει **FALSE**



Ισοδυναμία ύπαρξης με σύζευξη

Να βρεθεί ο κωδικός και ονοματεπώνυμο των υπαλλήλων του τμήματος 4 που απασχολούνται σε τουλάχιστον ένα έργο

```
1 SELECT DISTINCT e.empid, e.firstname, e.lastname
2 FROM employees e INNER JOIN workson w
3 ON e.empid = w.empid
4 WHERE depid = 4;
```

- Μερικές φορές ο τελεστής **EXISTS** ισοδυναμεί με **σύζευξη**.
- Αυτό γίνεται εύκολα αντιληπτό από την έκφραση **WHERE e.empid = w.empid** που χρησιμοποιήσαμε στο υποερώτημα.
- Το **EXISTS** έχει άρνηση το **NOT EXISTS**
- Για παράδειγμα, πως μπορούν να βρεθούν οι υπάλληλοι που δεν απασχολούνται σε κάποιο έργο;



Περιεχόμενα

- 1 Απλά υποερωτήματα
- 2 Τεστ: κάθε, τουλάχιστον, ύπαρξης
- 3 Επιπλέον Υποερωτήματα
- 4 Ένωση
- 5 Ασκήσεις επανάληψης



Υποερωτήματα στη φράση SELECT

Να υπολογιστεί το ποσοστό των υπαλλήλων ανά τμήμα, και το συνολικό ποσοστό (100%)

```
1  SELECT depid, 100*count(empid) /  
2      (SELECT count(empid) FROM employees)  
3      AS perEmp  
4  FROM employees  
5  GROUP BY depid WITH ROLLUP;
```

| <i>depid</i> | <i>perEmp</i> |
|--------------|---------------|
| 1 | 10.0000 |
| 2 | 13.3333 |
| 3 | 30.0000 |
| 4 | 16.6667 |
| 5 | 6.6667 |
| 6 | 23.3333 |
| NULL | 100.0000 |



Υποερωτήματα στη φράση FROM

Ένα οποιοδήποτε ερώτημα

```
1 SELECT depid, sum(salary)
2   FROM employees
3  GROUP BY depid
```

Μπορεί να “μεταφερθεί” στη φράση FROM αν του δώσουμε όνομα:

```
1 SELECT *
2   FROM ( SELECT depid, sum(salary)
3           FROM employees
4           GROUP BY depid) es;
```

- Η ονοματοδοσία του υποερωτήματος στη φράση **FROM** είναι υποχρεωτική
- Το επώνυμο αποτέλεσμα του ερωτήματος μέσα στις παρενθέσεις μπορεί να χρησιμοποιηθεί στη συνέχεια ως πίνακας ή όψη, πχ σε νέα σύζευξη



Να βρεθεί το ποσό της μισθοδοσίας ανά όνομα τμήματος

Με υποερώτημα στο FROM

```
1 SELECT depname, sumSal
2 FROM (
3     SELECT depid, SUM(salary) AS sumSal
4     FROM employees
5     GROUP BY depid
6 ) es
7 INNER JOIN departments d
8     ON es.depid = d.depid
```

Με φυσική σύζευξη

```
1 SELECT depname, SUM(salary) AS sumSal
2 FROM employees NATURAL JOIN departments
3 GROUP BY depname;
```



Περιεχόμενα

- 1 Απλά υποερωτήματα
- 2 Τεστ: κάθε, τουλάχιστον, ύπαρξης
- 3 Επιπλέον Υποερωτήματα
- 4 Ένωση
- 5 Ασκήσεις επανάληψης



Γενική μορφή ερωτημάτων ένωσης

```
1  SELECT ...  
2  UNION [ALL | DISTINCT]  
3  SELECT ...
```

- Τα αποτελέσματα των δύο ερωτημάτων πρέπει να έχουν “συμβατότητα τύπου”: ίδιο πλήθος πεδίων με αντίστοιχα κοινό πεδίο ορισμού
- Η επιλογή **ALL** θα επιστρέψει την ένωση κρατώντας τις διπλότιμες τιμές
- Η επιλογή **DISTINCT** θα επιστρέψει την ένωση χωρίς τις διπλότιμες τιμές, κάτι που είναι περισσότερο κοντά στον ορισμό της σχεσιακής πράξης της ένωσης



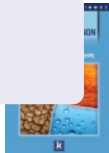
Παράδειγμα ένωσης - 1

Να βρεθούν οι υπάλληλοι που εργάζονται στα τμήματα 3 και 4

```
1 SELECT *  
2   FROM employees  
3   WHERE depid = 3  
4   UNION  
5 SELECT *  
6   FROM employees  
7   WHERE depid = 4;
```

Ισοδύναμο με:

```
1 SELECT *  
2   FROM employees  
3   WHERE depid IN (3,4);
```



Παράδειγμα ένωσης - 2

Να βρεθεί ο κωδικός και το όνομα των υπαλλήλων που είτε ανήκουν στο τμήμα με κωδικό 1, είτε δεν απασχολούνται σε κανένα έργο

```
1 SELECT empid, firstname, lastname
2     FROM employees e1
3     WHERE depid = 1
4     UNION
5 SELECT empid, firstname, lastname
6     FROM employees e2
7     WHERE NOT EXISTS (SELECT empid
8                        FROM workson w
9                        WHERE e2.empid = w.empid);
```

Ισοδύναμο με:

```
1 SELECT e.empid, e.firstname, e.lastname
2     FROM employees e LEFT JOIN workson w ON e.empid = w.empid
3     WHERE e.depid = 1
4           OR w.empid IS NULL;
```



Παράδειγμα ένωσης - 3

Όπως το προηγούμενο, αλλά με ένα επιπλέον πεδίο που να περιγράφει αν ο υπάλληλος είναι στη μία ή άλλη περίπτωση, πχ με το προσδιοριστικό «ΤΜΗΜΑ 1» ή «KANENA EPΓO»

```
1  SELECT empid, firstname, lastname, 'Department 1'
2                                     AS status
3
4  FROM employees e1
5  WHERE depid = 1
6  UNION
7  SELECT empid, firstname, lastname, 'Not in project'
8                                     AS status
9
10 FROM employees e2
11 WHERE NOT EXISTS (SELECT empid
                     FROM workson w
                     WHERE e2.empid = w.empid);
```

| <i>empid</i> | <i>firstname</i> | <i>lastname</i> | <i>status</i> |
|--------------|------------------|-----------------|----------------|
| 109 | Μαρία | Αθανασίου | Department 1 |
| 502 | Κρινιώ | Μαροπούλου | Department 1 |
| 901 | Κυριάκος | Ρούσσης | Department 1 |
| 311 | Νίκος | Στεργιόπουλος | Not in project |



Περιεχόμενα

- 1 Απλά υποερωτήματα
- 2 Τεστ: κάθε, τουλάχιστον, ύπαρξης
- 3 Επιπλέον Υποερωτήματα
- 4 Ένωση
- 5 Ασκήσεις επανάληψης



Να βρεθεί όνομα του τμήματος με το μεγαλύτερο ποσό μισθοδοσίας

```
1  SELECT depname
2      FROM departments NATURAL JOIN employees
3  GROUP BY depname
4      HAVING SUM(salary)=
5  (
6      SELECT sumSal
7          FROM (
8              SELECT depid, SUM(salary) AS sumSal
9                  FROM employees
10                 GROUP BY depid
11             ) es
12  WHERE sumSal >= ALL (
13              SELECT SUM(salary)
14                  FROM employees
15                 GROUP BY depid)
16  )
```



Να βρεθεί όνομα που τμήματος με το μεγαλύτερο ποσό μισθοδοσίας

Επειδή πάντα υπάρχει ο πιο εύκολος τρόπος

```
1 SELECT depname
2     FROM departments NATURAL JOIN employees
3 GROUP BY depname
4 ORDER BY SUM(salary) DESC
5 LIMIT 0, 1
```

Η λύση αυτή δεν είναι “αλγεβρική”,
βασίζεται στο τρυκ με τη χρήση του **LIMIT 0, 1**.



Να βρεθεί το όνομα του τμήματος και ο μέσος μισθός των υπαλλήλων του, για το τμήμα με το μικρότερη μέση τιμή στους μισθούς

Διπλή χρήση του υποερωτήματος στη φράση FROM και WHERE

```
1  SELECT d.depname, e1.msal
2      FROM ( SELECT depid, AVG(salary) AS msal
3              FROM employees
4              GROUP BY depid
5            ) e1
6      INNER JOIN departments d ON e1.depid=d.depid
7  WHERE e1.msal =
8      ( SELECT MIN(e2.msal)
9        FROM ( SELECT depid, AVG(salary)
10                AS msal
11              FROM employees
12              GROUP BY depid
13            ) e2
14      );
```



Να βρεθούν οι κωδικοί των έργων με τη μικρότερη απασχόληση εργαζομένων

Διπλή εμφώλευση υποερωτήματος στη φράση HAVING

```
1  SELECT proid
2  FROM workson
3  GROUP BY proid
4  HAVING count(empid) <= ALL
5      ( SELECT cnt
6        FROM
7          (
8            SELECT proid, COUNT(empid) AS cnt
9              FROM workson
10             GROUP BY proid
11             ) w2
12      );
```

Να βρεθούν οι κωδικοί των έργων με τη μικρότερη απασχόληση εργαζομένων, άλλος τρόπος με χρήση της συνάρτησης **min**

Διπλή εμφώλευση υποερωτήματος στη φράση HAVING

```
1  SELECT proid
2      FROM workson
3  GROUP BY proid
4      HAVING COUNT(empid) =
5              ( SELECT MIN(cnt)
6                  FROM
7                  (
8                      SELECT proid, COUNT(empid) AS cnt
9                      FROM workson
10                     GROUP BY proid
11                     ) w2
12              );
```



Σας ευχαριστώ
για την προσοχή σας

Είμαι στη διάθεσή σας για σχόλια, απορίες και ερωτήσεις

