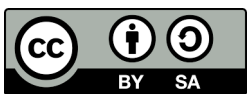
ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ
ΑΝΟΙΚΤΑ ΑΚΑΔΗΜΑΪΚΑ ΜΑΘΗΜΑΤΑ

Ηλεκτρονικοί Υπολογιστές ΙΙ

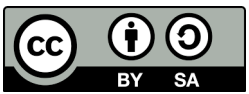
Υποερωτήματα SQL

Διδάσκων: Επίκουρος Καθηγητής
Αθανάσιος Σταυρακούδης



Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.



Υποερωτήματα στην SQL

Αθανάσιος Σταυρακούδης

http://stavrakoudis.econ.uoi.gr

Η ανάγκη για υποερώτημα

Ποιος υπάλληλος παίρνει το μεγαλύτερο μισθό;

Αν ξέραμε το μεγαλύτερο μισθό, πχ 2000, θα γράφαμε:

```
SELECT *
  FROM employees
 WHERE salary = 2000;
```

Ο μεγαλύτερος μισθός μπορεί να βρεθεί με:

```
SELECT MAX(salary)
  FROM empolyees;
```

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 τρώτο 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:

```
SELECT *
  FROM employees
 WHERE salary = (SELECT MAX(salary)
                   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;

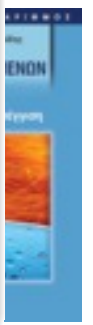
Να βρεθούν οι υπάλληλοι που εργάζονται στο ίδιο τμήμα με αυτό που εργάζεται ο υπάλληλος με κωδικό 102

Με αυτοσύζευξη

```
SELECT e1.*
  FROM employees e1, employees e2
 WHERE e1.depid = e2.depid
   AND e2.empid =102;
```

Με υποερώτημα

```
SELECT *
  FROM employees
 WHERE depid = (SELECT depid
                  FROM employees
                 WHERE empid = 102);
```



Υποερώτημα μετά το HAVING

Να βρεθεί το πλήθος των υπαλλήλων ανά όνομα τμήματος, για εκείνα τα τμήματα που απασχολούν λιγότερους από τους μισούς υπαλλήλους που απασχολούνται στο έργο 43

```
SELECT d.depname, COUNT(*)
  FROM employees e INNER JOIN departments d
                          ON e.depid = d.depid
GROUP BY d.depname
  HAVING COUNT(*) < 0.5*(SELECT COUNT(empid)
                           FROM workson
                          WHERE proid = 43);
```

depname	*COUNT(*)*
Γραμματείας	2

Στο έργο 43 απασχολούνται 5 υπάλληλοι και στο τμήμα Γραμματείας εργάζονται 2, δηλαδή λιγότερο από το μισό του 5

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

```
SELECT *
  FROM employees
 WHERE hiredate BETWEEN ( SELECT MIN(startdate)
                            FROM projects)
                    AND ( SELECT MAX(startdate)
                            FROM projects)
   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 €

```
SELECT depname
  FROM departments
 WHERE depid IN (SELECT DISTINCT depid
                   FROM employees
                  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 €

```
SELECT DISTINCT d.depname
  FROM employees e INNER JOIN departments d
                           ON e.depid = d.depid
 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!

Τεστ σύγκρισης με ALL

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

```
SELECT *
  FROM employees
 WHERE hiredate > ALL (SELECT DISTINCT startdate
                       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

```
SELECT *
  FROM employees
 WHERE hiredate > (SELECT MAX(startdate)
                   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

$$WHERE\ X\ \theta\ ALL\ Y$$

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

```
SELECT *
  FROM employees
 WHERE hiredate <= '2003-12-31'
   AND salary < ANY (SELECT DISTINCT salary
                       FROM employees
                      WHERE hiredate>'2004-01-01');
```

- 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

$$WHERE\ X\ \theta\ ANY\ Y$$

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 βρεθούν όλες οι λεπτομέρειες των υπαλλήλων που απασχολούνται σε τουλάχιστον ένα έργο

```
SELECT *
  FROM employees e
 WHERE EXISTS (SELECT empid
                 FROM workson w
                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 του παραδείγματος με τον 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 βρεθούν όλες οι λεπτομέρειες των υπαλλήλων που απασχολούνται σε τουλάχιστον ένα έργο

```
SELECT DISTINCT e.*
  FROM employees e INNER JOIN workson w
                   ON e.empid = w.empid
```

- 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;

Υποερωτήματα στη φράση SELECT

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

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

depid	*perEmp*
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

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

Μπορεί να ‘‘μεταφερθεί’’ στη φράση FROM αν του δώσουμε όνομα:

```
SELECT *
  FROM (  SELECT depid, sum(salary)
            FROM employees
        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

```
SELECT depname, sumSal
  FROM (
          SELECT depid, SUM(salary) AS sumSal
            FROM employees
         GROUP BY depid
        ) es
        INNER JOIN departments d
          ON es.depid = d.depid
```

Με φυσική σύζευξη

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

Να βρεθεί όνομα του τμήματος με το μεγαλύτερο ποσό μισθοδοσίας

```
   SELECT depname
     FROM departments NATURAL JOIN employees
GROUP BY depname
   HAVING sum(salary)=
(
   SELECT sumSal
     FROM (
             SELECT depid, SUM(salary) AS sumSal
               FROM employees
           GROUP BY depid
          ) es
    WHERE sumSal >= ALL (
                  SELECT SUM(salary)
                    FROM employees
                GROUP BY depid)
)
```

Να βρεθεί όνομα που τμήματος με το μεγαλύτερο ποσό μισθοδοσίας

Επειδή πάντα υπάρχει ο πιο εύκολος τρόπος

```
   SELECT depname
     FROM departments NATURAL JOIN employees
GROUP BY depname
ORDER BY SUM(salary) DESC
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

```
SELECT d.depname, e1.msal
  FROM (  SELECT depid, AVG(salary) AS msal
            FROM employees
         GROUP BY depid
        ) e1
        INNER JOIN departments d ON e1.depid=d.depid
 WHERE e1.msal =
        ( SELECT MIN(e2.msal)
            FROM (   SELECT depid, AVG(salary)
                                     AS msal
                       FROM employees
                   GROUP BY depid
                  ) e2
        );
```

Να βρεθούν οι κωδικοί των έργων με τη μικρότερη απασχόληση εργαζομένων

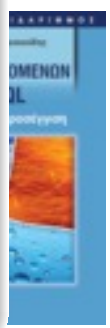
Διπλή εμφώλευση υποερωτήματος στη φράση HAVING

```
   SELECT proid
     FROM workson
GROUP BY proid
   HAVING count(empid) <= ALL
          ( SELECT cnt
              FROM
              (
                SELECT proid, count(empid) AS cnt
                  FROM workson
             GROUP BY proid
              ) w2
          );
```

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

Διπλή εμφώλευση υποερωτήματος στη φράση HAVING

```
   SELECT proid
     FROM workson
GROUP BY proid
   HAVING count(empid) =
          ( SELECT min(cnt)
              FROM
              (
                SELECT proid, count(empid) AS cnt
                  FROM workson
             GROUP BY proid
              ) w2
          );
```



Ένωση

Γενική μορφή ερωτημάτων ένωσης

SELECT ...
UNION [**ALL** | **DISTINCT**]
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

```
SELECT *
  FROM employees
 WHERE depid = 3
 UNION
SELECT *
  FROM employees
 WHERE depid = 4;
```

Ισοδύναμο με:

```
SELECT *
  FROM employees
 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

```
SELECT empid, firstname, lastname
  FROM employees e1
 WHERE depid = 1
 UNION
SELECT empid, firstname, lastname
  FROM employees e2
 WHERE NOT EXISTS (SELECT empid
                     FROM workson w
                    WHERE e2.empid = w.empid);
```

Ισοδύναμο με:

```
SELECT e.empid, e.firstname, e.lastname
  FROM employees e LEFT JOIN workson w ON e.empid = w.empid
 WHERE e.depid = 1
    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 «ΚΑΝΕΝΑ ΕΡΓΟ»

```
SELECT empid, firstname, lastname, 'Department 1'
                                    AS status
  FROM employees e1
 WHERE depid = 1
 UNION
SELECT empid, firstname, lastname, 'Not in project'
                                    AS status
  FROM employees e2
 WHERE NOT EXISTS (SELECT empid
                     FROM workson w
                    WHERE e2.empid = w.empid);
```

empid	*firstname*	*lastname*	*status*
109	Μαρία	Αθανασίου	Department 1
502	Κρινιώ	Μαροπούλου	Department 1
901	Κυριάκος	Ρούσσης	Department 1
311	Νίκος	Στεργιόπουλος	Not in project

Σας ευχαριστώ για την προσοχή σας

Είμαι στη διάθεσή σας για σχόλια, απορίες και ερωτήσεις

Τέλος Ενότητας

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Ιωαννίνων**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Σημειώματα

Σημείωμα Ιστορικού Εκδόσεων Έργου

Το παρόν έργο αποτελεί την έκδοση 1.0.

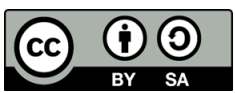
Έχουν προηγηθεί οι κάτωθι εκδόσεις:

- Έκδοση 1.0 διαθέσιμη εδώ. http://ecourse.uoi.gr/course/view.php?id=1065.

Σημείωμα Αναφοράς

Σημείωμα Αδειοδότησης

- Το παρόν υλικό διατίθεται με τους όρους της άδειας χρήσης Creative Commons Αναφορά Δημιουργού - Παρόμοια Διανομή, Διεθνής Έκδοση 4.0 [1] ή μεταγενέστερη.



- [1] https://creativecommons.org/licenses/by-sa/4.0/.