ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Συστήματα Διαχείρισης και Ανάλυσης Δεδομένων

Διδάσκων Καθηγητής
Ι. Κωτίδης

Φροντιστήριο 1

Καπέτης Χρυσόστομος

Εργαστηριακό Διδακτικό Προσωπικό

mkap@aueb.gr

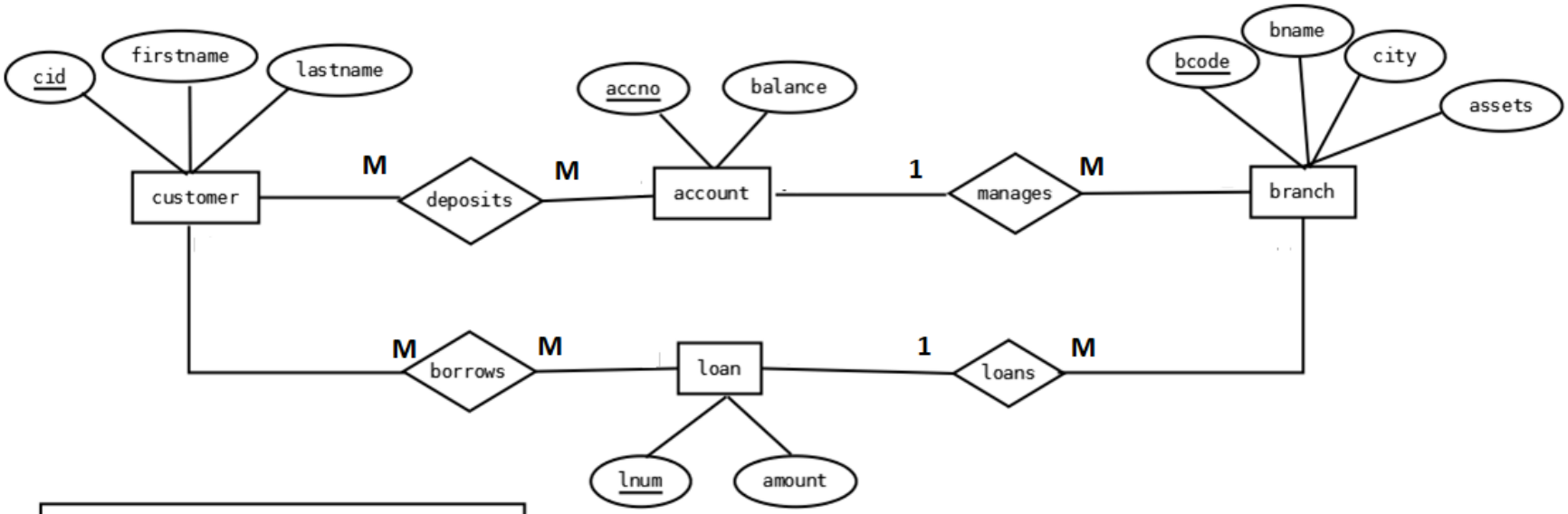
ΑΣΚΗΣΗ

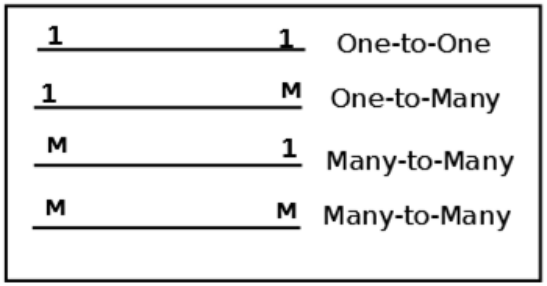
Μία νέα τράπεζα η οποία διαθέτει υποκαταστήματα σε διάφορες ελληνικές πόλεις, ενδιαφέρεται να εγκαταστήσει ένα πληροφοριακό σύστημα για την διαχείριση των βασικών συναλλαγών με τους πελάτες της. Οι πελάτες της τράπεζας έχουν τη δυνατότητα να τηρούν λογαριασμούς στα διάφορα υποκαταστήματά της, καθώς επίσης και να δανείζονται χρήματα υπό την μορφή δανείων. Οι πελάτες της τράπεζας μπορούν αν το επιθυμούν να τηρούν έναν ή περισσότερους λογαριασμούς σε ένα ή περισσότερα υποκαταστήματα της τράπεζας, καθώς επίσης και να δανείζονται χρήματα από οποιοδήποτε υποκατάστημα. Η τράπεζα δίνει τη δυνατότητα στους πελάτες της να τηρούν κοινούς λογαριασμούς καθώς επίσης και να συνάπτουν κοινά δάνεια.

Ζητείται να σχεδιάσετε τη βάση δεδομένων για το πληροφοριακό σύστημα της τράπεζας. Συγκεκριμένα:

- Να σχεδιάσετε το διάγραμμα E-R
- Να σχεδιάσετε το λογικό σχήμα της βάσης δεδομένων.
- Να υλοποιήσετε το λογικό σχήμα της βάσης χρησιμοποιώντας της γλώσσα SQL.
- Να δημιουργήσετε τη βάση δεδομένων με τη χρήση του RDBMS SQL SERVER.

Τραπεζικό σύστημα
Διάγραμμα ER





customer (**#cid**, firstname,lastname,city)
account (**#accno**, bcode, balance)
branch (**#bcode**, bname, city, assets)
depositor (**#cid, #accno**)
loan (**#lnum**, bcode, amount)
borrower (#**cid, #lnum**)

Τραπεζικό Σύστημα
Λογικό Σχήμα σε SQL

ΟΠΑ AUEB

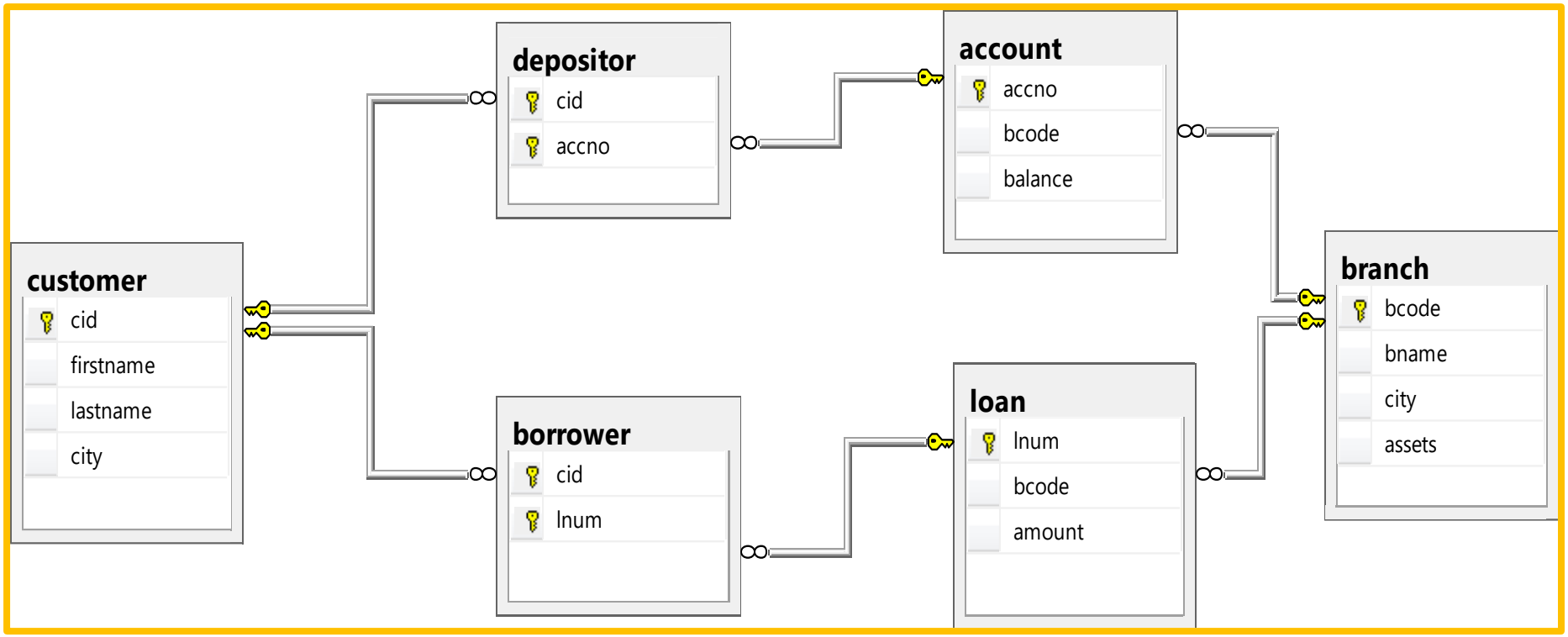
```
CREATE TABLE customer(
    cid int   NOT NULL,
    firstname varchar(30) NULL,
    lastname varchar(30) NOT NULL,
    city varchar(30) NOT NULL,
            PRIMARY KEY (cid)
);
CREATE TABLE branch(
    bcode int  NOT NULL,
    bname varchar(30) NOT NULL,
    city varchar(30) NOT NULL,
    assets numeric(18, 0) NOT NULL,
            PRIMARY KEY(bcode)
);
CREATE TABLE  account (
    accno   varchar(10)  NOT NULL,
    bcode   int  NOT NULL,
    balance numeric(18, 0) NOT NULL,
            PRIMARY KEY(accno),
            CONSTRAINT fk_account FOREIGN KEY (bcode) REFERENCES branch(bcode)
);
```

Τραπεζικό Σύστημα
Λογικό Σχήμα σε SQL

ΟΠΑ AUEB

```
CREATE TABLE depositor(
    cid int NOT NULL,
    accno varchar(10) NOT NULL,
        PRIMARY KEY(cid,accno),
        CONSTRAINT fk1_depositor FOREIGN KEY (cid) REFERENCES customer(cid),
        CONSTRAINT fk2_depositor FOREIGN KEY (accno) REFERENCES account (accno)
);
CREATE TABLE loan(
    lnum varchar(10) NOT NULL,
    bcode int NOT NULL,
    amount numeric(18, 0) NOT NULL,
        PRIMARY KEY(lnum),
        CONSTRAINT fk_loan FOREIGN KEY (bcode) REFERENCES branch(bcode)
);
CREATE TABLE borrower (
    cid  int  NOT NULL,
    lnum   varchar (10) NOT NULL,
        PRIMARY KEY(cid,lnum),
        CONSTRAINT fk1_borrower FOREIGN KEY (cid) REFERENCES customer(cid),
        CONSTRAINT fk2_borrower FOREIGN KEY (lnum) REFERENCES loan(lnum)
);
```

Σχήμα Παραδειγμάτων



Η ΓΛΩΣΣΑ SQL

- Βασική Δομή
- Πράξεις Συνόλων
- Συνοπτικές Συναρτήσεις
- Κενές Τιμές
- Ένθετα Υποερωτήματα
- Τροποποίηση Βάσης Δεδομένων
- Τύποι Συνδέσμων
- Γλώσσα Ορισμού Δεδομένων

Βασική Δομή

- Η SQL βασίζεται σε ορισμένες λειτουργίες συνόλων και σχέσεων με ορισμένες τροποποιήσεις και επεκτάσεις.
- Ένα τυπικό ερώτημα SQL έχει τη μορφή:

  select $A_1, A_2, ..., A_n$
  from $r_1, r_2, ..., r_m$
  where P

  - A_i αντιπροσωπεύει μία ιδιότητα
  - r_i *αντιπροσωπεύει μία σχέση*
  - P είναι ένα κατηγόρημα.

- Το αποτέλεσμα ενός ερωτήματος είναι μία σχέση

Βασική Δομή

- Η SQL βασίζεται σε ορισμένες λειτουργίες συνόλων και σχέσεων με ορισμένες τροποποιήσεις και επεκτάσεις.
- Ένα τυπικό ερώτημα SQL έχει τη μορφή:

  select $A_1, A_2, ..., A_n$
  from $r_1, r_2, ..., r_m$
  where P

  - A_i αντιπροσωπεύει μία ιδιότητα
  - r_i *αντιπροσωπεύει μία σχέση*
  - P είναι ένα κατηγόρημα.

- Το αποτέλεσμα ενός ερωτήματος είναι μία σχέση

Ο όρος select

- Ο όρος **select** προσδιορίζει τις επιθυμητές ιδιότητες των αποτελεσμάτων του ερωτήματος.
  - Π.χ. βρες τα ονόματα όλων των υποκαταστημάτων στη σχέση *branch*

select *bname*
from *branch*

- Η SQL δεν κάνει διάκριση μεταξύ πεζών και κεφαλαίων στα αναγνωριστικά.

Ο όρος select (συνέχεια)

- Η SQL επιτρέπει διπλότυπα στις σχέσεις όπως και στα αποτελέσματα των ερωτημάτων
- Για την απαλοιφή των διπλότυπων πρέπει να χρησιμοποιηθεί ο προσδιοριστής **distinct** μετά τον όρο **select.**
- Βρες όλους τους λογαριασμούς στη σχέση *depositor* και αφαίρεσε τα διπλότυπα.

  select distinct *accno*
  from *depositor*

- Ο προσδιοριστής **all** επιβάλει τη μη διαγραφή των διπλοτύπων. **select all** *accno*
  from *depositor*

Ο όρος select (συνέχεια)

- Το αστεράκι "*" μετά τον όρο select υποδηλώνει την εμφάνιση όλων των γνωρισμάτων.

  select *
  from *loan*

- Ο όρος **select** μπορεί να περιέχει αριθμητικές εκφράσεις με την χρήση των τελεστών +, –, *, and /.
- Το ερώτημα:

  select *lnum, bcode, amount* * 100
  from *loan*

  επιστρέφει μία σχέση η οποία είναι ίδια με τη σχέση loan, εκτός από το γνώρισμα amount το οποίο έχει πολλαπλασιαστεί με το 100.

Ο όρος where

- Ο όρος **where** προσδιορίζει την συνθήκη που πρέπει να πληρούν τα αποτελέσματα.
- Η παρακάτω εντολή εμφανίζει όλους τους αριθμούς των δανείων με ποσό μεγαλύτερο των 5000 Ευρώ.

  select *lnum*
  from *loan*
  where *amount* > 5000

- Τα αποτελέσματα μπορούν να συνδυάζονται με την χρήση των λογικών τελεστών **and, or,** και **not.**

Ο όρος where (Συνέχεια)

- Η SQL διαθέτει τον τελεστή σύγκρισης **between**
- Π.χ. Βρες τον αριθμό των δανείων αριθμό των δανείων των οποίων το ποσό κυμαίνεται μεταξύ 5000 και 15000

select *lnum*
from *loan*
where *amount* **between** 5000 **and** 15000

- Το ίδιο ερώτημα με χρήση λογικών τελεστών

select lnum
from loan
where amount >= 5000 **and** amount <=15000

Ο όρος from

- Ο όρος **from** ακολουθείται από τις σχέσεις που συμμετέχουν στο ερώτημα.
- Βρες τον κωδικό και το ονοματεπώνυμο όλων των πελατών που διαθέτουν λογαριασμό σε οποιοδήποτε υποκατάστημα της τράπεζας.

```
select distinct  customer.cid, firstname, lastname
        from customer, depositor
    where   customer.cid = depositor.cid
```

Η λειτουργία Μετονομασίας

- Η SQL επιτρέπει την μετονομασία σχέσεων και γνωρισμάτων με τη χρήση του προσδιοριστή the **as :**

  old-name **as** *new-name*

- Βρες το όνομα, τον αριθμό και το ποσό των δανείων όλων των πελατών. Μετονόμασε την στήλη lnum σε loan_number.

select *firstname, lastname, borrower.lnum* **as** *loan_number, amount*
from customer,*borrower, loan*
where customer.cid=borrower.cid **and**
borrower.lnum = loan.lnum

Μεταβλητές Εγγραφών

- Οι μεταβλητές εγγραφών ορίζονται στον όρο **from** με τη χρήση του προσδιοριστή **as**.
- Βρες το όνομα, τον αριθμό και το ποσό των δανείων όλων των πελατών.

select *firstname, lastname, B.lnum , amount*
from customer AS C, *borrower AS B, loan AS L*
where C.cid=B.cid **and**
B.lnum = L.lnum

- Βρες τα ονόματα όλων των υποκαταστημάτων με αποθεματικό μεγαλύτερο από ορισμένα υποκαταστήματα της Θεσσαλονίκης.

select distinct *T.bname*
from *branch* **as** *T, branch* **as** *S*
where *T.assets > S.assets* **and** *S.city = 'Θεσσαλονίκη'*

Λειτουργίες Συμβολοσειρών

- Η SQL υποστηρίζει την σύγκριση συμβολοσειρών με τη χρήση δύο ειδικών χαρακτήρων:
  - Τον χαρακτήρας αποκοπής %.
  - Τον χαρακτήρα αντικατάστασης _.

- Βρες όλα τα ονόματα των πελατών των οποίων το επώνυμο ξεκινάει με “Α”.

select *lastname*
from *customer*
where *lastname* **like** ***‘A%’***

- Εντόπισε την συμβολοσειρά “Main%”

like ‘Main\%’ **escape** ‘\’

Ταξινόμηση των Αποτελεσμάτων

- Εμφάνισε σε αλφαβητική σειρά τα ονόματα των πελατών που έχουν λογαριασμό με υπόλοιπο πάνω από 50000.

```
select distinct lastname, firstname
from    customer, depositor, account
where customer.cid=depositor.cid and
          depositor.accno = account.accno and
          balance >=50000
order by lastname, firstname
```

- Μπορούμε να ορίσουμε **desc** για φθίνουσα ή **asc** για αύξουσα ταξινόμηση.
  - Π.χ. **order by** *lastname* **desc**

Πράξεις Συνόλων

- Οι πράξεις συνόλων **union, intersect,** και **except** εφαρμόζουν σε σχέσεις και αντιστοιχούν στους εξής τελεστές της σχεσιακής άλγεβρας: $\cup$, $\cap$, $-$.
- Κάθε μία από τις παραπάνω λειτουργίες απαλείφει αυτόματα τα διπλότυπα. Για την διατήρηση των διπλότυπων πρέπει να χρησιμοποιηθούν οι τελεστές **union all, intersect all** και **except all.**

  Υποθέστε ότι μία πλειάδα εμφανίζεται *m* φορές στην σχέση *r* and και *n* φορές στην σχέση *s, τότε εμφανίζεται:*
  - *m + n* φορές στην σχέση *r* **union all** *s*
  - min(*m,n)* φορές στην σχέση r **intersect all** *s*
  - max(0, *m – n)* φορές στην σχέση *r* **except all** *s*

Πράξεις Συνόλων

Βρες όλους τους πελάτες που έχουν ένα δάνειο, ένα λογαριασμό ή και τα δύο:

```
select lastname,firstname
    from customer,depositor
  where  customer.cid=depositor.cid

union

select lastname,firstname
    from customer,borrower
  where  customer.cid=borrower.cid
```

Πράξεις Συνόλων

Βρες όλους τους πελάτες που διαθέτουν κάποιο λογαριασμό και επίσης έχουν πάρει δάνειο:

```
select lastname,firstname
      from customer,depositor
   where  customer.cid=depositor.cid
INTERSECT
select lastname,firstname
      from customer,borrower
   where  customer.cid=borrower.cid
```

Πράξεις Συνόλων

Βρες όλους τους πελάτες που διαθέτουν λογαριασμό και δεν έχουν πάρει δάνειο.

```
select lastname,firstname
      from customer,depositor
   where  customer.cid=depositor.cid
EXCEPT
select lastname,firstname
      from customer,borrower
   where  customer.cid=borrower.cid
```

Συνοπτικές Συναρτήσεις

- Οι παρακάτω συναρτήσεις εφαρμόζουν σε αν σύνολο τιμών μιας στήλης και επιστρέφουν ως αποτέλεσμα μία τιμή.

avg: μέση τιμή
min: ελάχιστη τιμή
max: μέγιστη τιμή
sum: άθροισμα τιμών
count: πλήθος τιμών

Συνοπτικές Συναρτήσεις (Συνέχεια)

- Βρες το μέσο όρο των λογαριασμών του υποκαταστήματος "Σταδίου".

```
select avg (balance)
        from branch,account
        where branch.bcode=account.bcode and
            bname = 'Σταδίου'
```

- Βρες τον αριθμό των πελατών της τράπεζας.

```
select count (cid)
        from customer
```

- Βρες τον αριθμό των καταθετών της τράπεζας.

```
select count (distinct cid)
        from depositor
```

Συνοπτικές Συναρτήσεις – Group By

- Βρες τον αριθμό των καταθετών ανά υποκατάστημα.

select *bname,* **count (distinct** *cid)*
from branch,*depositor, account*
where branch.bcode=account.bcode **and**
depositor.accno = account.accno
group by *bname*

Σημείωση: Τα γνωρίσματα που προσδιορίζονται με τον όρο **select,** εκτός των συνοπτικών συναρτήσεων, πρέπει να ακολουθούν τον προσδιοριστή **group by.**

Συνοπτικές Συναρτήσεις – ο όρος Having

- Βρες τα ονόματα των υποκαταστημάτων για τα οποία ο μέσος όρος των καταθέσεων είναι μεγαλύτερος από 10000

select *bname,* ***avg*** **(**balance*)*
from branch,*account*
where branch.bcode=account.bcode
group by *bname*
having avg*(balance) > 10000*

Σημείωση: η συνθήκη που ακολουθεί τον όρο **having** εφαρμόζεται μετά την δημιουργία των ομάδων, σε αντίθεση με τη συνθήκη που ακολουθεί τον όρο **where** η οποία εφαρμόζεται πριν την δημιουργία των ομάδων.

Κενές Τιμές

- Η συνθήκη **is null** μπορεί να χρησιμοποιηθεί για έλεγχο κενών τιμών.
- Π.χ. βρες όλους τους αριθμούς των δανείων στη σχέση *loan που περιέχουν στο amount την τιμή null.*

  select *lnum*
  from *loan*
  where *amount* **is null**

- Το αποτέλεσμα κάθε αριθμητικής έκφρασης που περιέχει την τιμή *null* είναι *null*
  - Π.χ. 5 + null = null
- Οι συνοπτικές συναρτήσεις απλά αγνοούν τις τιμές που είναι null.

Κενές τιμές (Συνέχεια)

- Κάθε σύγκριση με τιμές *null* επιστρέφει *unknown*
  - *Π.χ. 5 < null or null <> null or null = null*
- Λογικοί τελεστές με χρήστη της τιμής *unknown*:
  - OR: (*unknown* **or** *true*) = *true*, (*unknown* **or** *false*) = *unknown*
    (*unknown* **or** *unknown*) = *unknown*
  - AND: *(true* **and** *unknown) = unknown, (false* **and** *unknown) = false,*
    (unknown **and** *unknown) = unknown*
  - NOT*: (***not** *unknown) = unknown*
  - "*P* **is unknown"** αποτιμάται σε true αν η *P* αποτιμάται σε *unknown*
- Το αποτέλεσμα της συνθήκης του όρου **where** είναι false αν αποτιμάται σε *unknown*

Κενές τιμές (Συνέχεια)

- Συνολικό ποσό δανείων

  select sum (*amount)*
  from *loan*

  – Η παραπάνω εντολή αγνοεί όλες τις τιμές null.
- Όλες οι συνοπτικές συναρτήσεις εκτός από την **count(*)** αγνοούν τις κενές τιμές των πλειάδων στα γνωρίσματα που δέχονται ως όρισμα.

Ένθετα Υποερωτήματα

- Ένα υποερώτημα είναι μία παράσταση **select-from-where** που είναι ένθετο μέσα σε ένα άλλο ερώτημα.
- Μία συνηθισμένη χρήση των υποερωτημάτων είναι να εκτελούν ελέγχους αν ανήκουν σε ένα σύνολο, να κάνουν συγκρίσεις συνόλων και να προσδιορίσουν την τάξη ενός συνόλου.

Παράδειγμα Ερωτήματος

- Βρες όλους τους πελάτες που διαθέτουν λογαριασμό και έχουν πάρει δάνειο.

```
select  distinct lastname,firstname
      from customer,depositor
    where  customer.cid=depositor.cid and
           customer.cid IN (select cid from borrower)
```

- Βρες όλους τους πελάτες που διαθέτουν λογαριασμό και δεν έχουν πάρει δάνειο.

```
select distinct lastname,firstname
      from customer,depositor
    where  customer.cid=depositor.cid and
           customer.cid NOT IN (select cid from borrower)
```

Σύγκριση Συνόλων

- Βρείτε όλα τα υποκαταστήματα με κεφάλαια μεγαλύτερα τουλάχιστον από ένα υποκατάστημα που βρίσκεται στην Πάτρα.

```
select distinct  T.bname
        from branch as T, branch as S
        where  T.assets > S.assets and
               S.city = 'Πάτρα'
```

- Το ίδιο ερώτημα με τον προσδιοριστή **some**.

```
select bname
        from branch
        where assets > some
              (select assets
               from branch
                    where city ='Πάτρα')
```

Ορισμός του Προσδιοριστή Some

- F <comp> **some** *r* $\Leftrightarrow \exists\ t \in r$ *έτσι ώστε (*F <comp> *t*)
  όπου <comp> μπορεί να είναι : $<, \leq, >, =, \neq$

(5< **some** | 0 | 5 | 6 |) = true

(Σημαίνει: 5 < από μερικές πλειάδες της σχέσης)

(5< **some** | 0 | 5 |) = false

(5 = **some** | 0 | 5 |) = true

(5 ≠ **some** | 0 | 5 |) = true (επειδή 0 ≠ 5)

Ορισμός του προσδιοριστή all

- F <comp> **all** $r \Leftrightarrow \forall\ t \in r$ (F <comp> *t)*

(5< **all** {0, 5, 6}) = false

(5< **all** {6, 10}) = true

(Σημαίνει: 5 < από όλες πλειάδες της σχέσης)

(5 = **all** {4, 5}) = false

(5 ≠ **all** {4, 6}

Παράδειγμα Ερωτήματος

- Βρες τα ονόματα όλων των υποκαταστημάτων με κεφάλαια μεγαλύτερα από όλα τα υποκαταστήματα της Πάτρας.

select *bname*
from *branch*
where *assets* > ***all***
(select *assets*
from *branch*
where *city* = ‘Πάτρα’)

Τροποποίηση Βάσης – Διαγραφή

- Διέγραψε όλους τους λογαριασμούς των υποκαταστημάτων της Θεσσαλονίκης.

```
delete from depositor
where accno in
            (select accno
               from branch, account
               where account.bcode=branch.bcode
                 and city ='Θεσσαλονίκη')

delete from account
    where bcode in (select bcode
                      from branch
                     where city ='Θεσσαλονίκη')
```

Παραδείγματα Ερωτημάτων

- Διέγραψε όλες τις εγγραφές των λογαριασμών με υπόλοιπο μικρότερο του μέσου όρου της τράπεζας.

```
delete from account
    where balance < (select avg (balance)
            from account)
```

- **Πρόβλημα:** κατά την διαγραφή των πλειάδων ο μέσος όρος αλλάζει

Λύση στην SQL:

1. Πρώτα υπολογίζεται ο μέσος όρος **avg** και εντοπίζονται οι εγγραφές προς διαγραφή.
2. Στη συνέχεια διαγράφονται οι παραπάνω εγγραφές δίχως νέο υπολογισμό του μέσου όρου.

Τροποποίηση Βάσης – Εισαγωγή

- Πρόσθεσε μία νέα πλειάδα στον πίνακα **customer**

  insert into *account*
  values (‘A-9732’, ‘Perryridge’,1200)

  ή ισοδύναμα:

  insert into customer (cid, firstname, lastname, city)
  values (1, 'Μάριος', 'Αβραμίδης', 'Αθήνα')

- Πρόσθεσε μία νέα πλειάδα στον πίνακα account με την τιμή null στο πεδίο balance

  insert into *account*
  values (‘A100’,‘2’, *null*)

Τροποποίηση της βάσης – Εισαγωγή

- Δώσε δώρο σε όλους τους πελάτες που έχουν δάνειο στο υποκατάστημα “Πατησίων” ένα λογαριασμό καταθέσεων με υπόλοιπο 500 ευρώ. Ο αριθμός δανείου θα αποτελέσει τον αριθμό του νέου λογαριασμού για κάθε πελάτη.

insert into *account*
select *lnum, loan.bcode,* 500
from *loan, branch*
where *loan.bcode=branch.bcode* ***and*** *bname=‘Πατησίων’*
insert into *depositor*
select *cid, lnum*
from *loan, borrower,branch*
where loan.lnum=borrower.lnum **and**
loan.bcode=branch.bcode ***and*** *bname=‘Πατησίων’*

- Η πρόταση select from where αποτιμάται ολοκληρωτικά πρίν τα αποτελέσματά της καταχωρηθούν στην αντίστοιχη σχέση. Διαφορετικά υπάρχει πιθανότητα λάθους.

Τροποποίηση Βάσης – Ενημέρωση

- Δώσε αύξηση 6% σε όλους τους λογαριασμούς με υπόλοιπο μεγαλύτερο του 10000 ευρώ. Οι υπόλοιποι λογαριασμοί θα πάρουν αύξηση 5%.
  - Απαιτούνται δύο εντολές **update:**

update *account*
set *balance* = *balance* ∗ 1.06
where *balance* > 10000

update *account*
set *balance* = *balance* ∗ 1.05
where *balance* <= 10000

  - Η σειρά είναι σημαντική
  - Μπορεί να γραφεί καλύτερα με χρήση της εντολής **case**

Η εντολή Case

- Το ίδιο ερώτημα με το προηγούμενο αλλά με χρήση της εντολής **case**

```
update account
   set balance =  case
       when balance <= 10000 then balance *1,05
       else   balance * 1,06
   end
```

Σχέσεις Συνδέσμων

- Οι σχέσεις συνδέσμων παίρνουν δύο σχέσεις και επιστρέφουν ως αποτέλεσμα μία άλλη σχέση.
- Οι συνθήκες συνδέσμων προσδιορίζουν τις πλειάδες που ταιριάζουν μεταξύ των δύο σχέσεων καθώς επίσης και ποια γνωρίσματα εμφανίζονται στο αποτέλεσμα της σύνδεσης.
- Ο τύπος της σύνδεσης προσδιορίζει πως αντιμετωπίζονται οι πλειάδες σε κάθε σχέση που δεν ταιριάζουν με καμία πλειάδα στην άλλη σχέση.
- Τύποι Συνδέσμων:
  - **Inner Join**
  - **Left join**
  - **Right join**
  - **Full join**

Σχέσεις Συνδέσμων – Παραδείγματα

account

ACCNO	BALANCE	BCODE
A900	1000	100
A905	50000	150
A925	6000	700
A907	5000	800

branch

BCODE	BNAME	CITY	ASSETS
100	Σταδίου	Αθήνα	10000000
150	Πατησίων	Αθήνα	700000
250	Τσιμισκή	Θεσσαλονίκη	500000
350	Αμαλίας	Πάτρα	400000

- *account* **INNER JOIN** *branch* **ON**
  account.bcode = branch.bcode

ACCNO	BALANCE	BCODE	BCODE	BNAME	CITY	ASSETS
A900	1000	100	100	Σταδίου	Αθήνα	10000000
A905	50000	150	150	Πατησίων	Αθήνα	700000

Σχέσεις Συνδέσμων – Παραδείγματα

account

ACCNO	BALANCE	BCODE
A900	1000	100
A905	50000	150
A925	6000	700
A907	5000	800

branch

BCODE	BNAME	CITY	ASSETS
100	Σταδίου	Αθήνα	10000000
150	Πατησίων	Αθήνα	700000
250	Τσιμισκή	Θεσσαλονίκη	500000
350	Αμαλίας	Πάτρα	400000

- *account* ***LEFT*** **JOIN** *branch* **ON**
  account.bcode = branch.bcode

ACCNO	BALANCE	BCODE	BCODE	BNAME	CITY	ASSETS
A900	1000	100	100	Σταδίου	Αθήνα	10000000
A905	50000	150	150	Πατησίων	Αθήνα	700000
A925	6000	700	null	null	null	null
A907	5000	800	null	null	null	null

Σχέσεις Συνδέσμων – Παραδείγματα

account

ACCNO	BALANCE	BCODE
A900	1000	100
A905	50000	150
A925	6000	700
A907	5000	800

branch

BCODE	BNAME	CITY	ASSETS
100	Σταδίου	Αθήνα	10000000
150	Πατησίων	Αθήνα	700000
250	Τσιμισκή	Θεσσαλονίκη	500000
350	Αμαλίας	Πάτρα	400000

- *account* ***RIGHT*** **JOIN** *branch* **ON**
  account.bcode = branch.bcode

ACCNO	BALANCE	BCODE	BCODE	BNAME	CITY	ASSETS
A900	1000	100	100	Σταδίου	Αθήνα	10000000
A905	50000	150	150	Πατησίων	Αθήνα	700000
null	null	null	250	Τσιμισκή	Θεσσαλονίκη	500000
null	null	null	350	Αμαλίας	Πάτρα	400000

Σχέσεις Συνδέσμων – Παραδείγματα

account

ACCNO	BALANCE	BCODE
A900	1000	100
A905	50000	150
A925	6000	700
A907	5000	800

branch

BCODE	BNAME	CITY	ASSETS
100	Σταδίου	Αθήνα	10000000
150	Πατησίων	Αθήνα	700000
250	Τσιμισκή	Θεσσαλονίκη	500000
350	Αμαλίας	Πάτρα	400000

- *account* ***FULL*** **JOIN** *branch* **ON**

  account.bcode = branch.bcode

ACCNO	BALANCE	BCODE	BCODE	BNAME	CITY	ASSETS
A900	1000	100	100	Σταδίου	Αθήνα	10000000
A905	50000	150	150	Πατησίων	Αθήνα	700000
A925	6000	700	null	null	null	null
A907	5000	800	null	null	null	Null
null	null	null	250	Τσιμισκή	Θεσσαλονίκη	500000
null	null	null	350	Αμαλίας	Πάτρα	400000

Γλώσσα Ορισμού Δεδομένων (DDL)

- Επιτρέπει τον προσδιορισμό ενός συνόλου σχέσεων καθώς επίσης και των παρακάτω στοιχείων για κάθε σχέση:
  - Το σχήμα της σχέσης.
  - Το πεδίο τιμών κάθε γνωρίσματος της σχέσης.
  - Περιορισμούς ακεραιότητας
  - Τους δείκτες κάθε σχέσης
  - Πληροφορίες ασφάλειας και δικαιωμάτων κάθε σχέσης
  - Την φυσική δομή αποθήκευσης κάθε σχέσης.

Τύποι Πεδίου Τιμών στην SQL

- **char(n).** Συμβολοσειρά σταθερού μήκους με μήκος n καθορισμένο από το χρήστη
- **varchar(n).** Συμβολοσειρά μεταβλητού μήκους με μέγιστο μήκος n καθορισμένο από τον χρήστη
- **int.** Ακέραιος. Το μήκος εξαρτάται από την μηχανή
- **smallint.** Μικρός ακέραιος. Το μήκος εξαρτάται από την μηχανή.
- **numeric(p,d).** Αριθμός σταθερής υποδιαστολής με ακρίβεια καθορισμένη από το χρήστη. Ο αριθμός αποτελείται από p ψηφία (και ένα πρόσημο) και τα d από p ψηφία είναι δεξιά του δεκαδικού συμβόλου.
- **real, double precision.** Αριθμοί κινητής υποδιαστολής και διπλής ακρίβειας το μέγεθος των οποίων εξαρτάται από την μηχανή.
- **float(n).** Αριθμός κινητής υποδιαστολής με ακρίβεια τουλάχιστον n ψηφία.
- Η τιμή Null επιτρέπεται σε όλους τους τύπους.
- Η δομή **create domain** στην SQL-92 επιτρέπει την δημιουργία τύπων οριζόμενων από το χρήστη.

  create domain *person-name* **char**(20) **not null**

Τύποι Date/Time στην SQL

- **date.** Ημερομηνίες που περιέχουν 4 ψηφία για το έτος, τον μήνα και την ημέρα.
  - Π.χ. **date** '2001-7-27'
- **time.** Η ώρα της ημέρας σε ώρες, λεπτά και δευτερόλεπτα.
  - Π.χ. **time** '09:00:30' **time** '09:00:30.75'
- **timestamp**: Ημερομηνία και ώρα
  - Π.χ. **timestamp** '2001-7-27 09:00:30.75'
- **Interval**: χρονική περίοδος
  - Π.χ. Interval '1' day

Η εντολή Create Table

- Μια σχέση στην SQL ορίζεται με την εντολή **create table**:

  create table r (A_1 D_1, A_2 D_2, ..., A_n D_n,
  (integrity-constraint$_1$),
  ...,
  (integrity-constraint))

  - R είναι το όνομα της σχέσης
  - κάθε A_i είναι το όνομα ενός γνωρίσματος στο σχήμα της σχέσης r
  - D_i είναι ο τύπος δεδομένων του γνωρίσματος A_i

- Παράδειγμα:

```
CREATE TABLE branch(
    bcode int  NOT NULL,
    bname varchar(30) NOT NULL,
    city varchar(30) NOT NULL,
    assets numeric(18, 0) NOT NULL
)
```

Περιορισμοί ακεραιότητας στην εντολή Create Table

- **not null**
- **primary key** (A_1, ..., A_n)
- **check** *(P),* όπου *P* είναι ένα κατηγόρημα που πρέπει να ικανοποιείται από κάθε εγγραφή της σχέσης.

Παράδειγμα: Όρισε το γνώρισμα bcode ως πρωτεύον κλειδί της σχέσης branch και επιβεβαίωσε ότι οι τιμές του γνωρίσματος assets δεν είναι αρνητικοί αριθμοί.

```
CREATE TABLE branch(
    bcode int  NOT NULL,
    bname varchar(30) NOT NULL,
    city varchar(30) NOT NULL,
    assets numeric(18, 0) NOT NULL,
        PRIMARY KEY (bcode),
        CHECK (assets)>=0
)
```

Οι εντολές Drop και Alter Table

- Η εντολή **drop table** διαγράφει όλες τις πληροφορίες της μιας σχέσης από τη βάση.
- Η εντολή **alter table** μπορεί να χρησιμοποιηθεί για να προσθέσει νέα γνωρίσματα σε μία σχέση..

  alter table *r* **add** *A D όπου:*

  A είναι το όνομα της ιδιότητας που θα προστεθεί στην σχέση *r* και *D* ο τύπος δεδομένων του *A.*

  – Σε όλες τις υπάρχουσες εγγραφές μπαίνει η τιμή *null στο νέο γνώρισμα*
- Η εντολή **alter table** μπορεί να χρησιμοποιηθεί για διαγραφή γνωρισμάτων από μία σχέση

  alter table *r* **drop** *A*

  όπου *A* είναι το όνομα του προς διαγραφή γνωρίσματος της σχέσης *r*

  – Πολλές βάσεις δεδομένων δεν υποστηρίζουν την διαγραφή γνωρισμάτων

Βιβλιογραφία

- Γιαννακουδάκης, Εμμανουήλ Ι. (2014) Βάσεις δεδομένων: θεωρία και πράξη, Αθήνα: Μπένος.
- Silberschatz, Korth, Sudarsham (2001) Database system concepts. Fourth Edition, McGraw-Hill.