ΕΘΝΙΚΟ **Μ**ΕΤΣΟΒΙΟ **Π**ΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ & ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΕΞΑΜΗΝΙΑΙΑ ΕΡΓΑΣΙΑ ΣΤΙΣ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

Ακ. έτος 2018-2019, 6ο Εξάμηνο

Όνομα ομάδας: Ομάδα 60

Μέλη:
- Κρανιάς Δημήτριος, 031 16 030
- Μαρκουλέσκου Έλενα-Μπιάνκα, 031 15 126
- Μπέτζελος Χρήστος, 031 16 067

Η εργασία αφορά στην ανάπτυξη σε σχεσιακό σύστημα της βάσης της βιβλιοθήκης του Πολυτεχνείου, σύμφωνα με αυτά που περιγράφονται στην πρώτη Άσκηση του μαθήματος. Στηριχτήκαμε στην προτεινόμενη λύση που μας δόθηκε στη σελίδα του μαθήματος και υλοποιήσαμε τα ερωτήματα που μας ζητήθηκαν.

Μέρος (i)

Η πλατφόρμα που επιλέξαμε για την δημιουργία της βάσης δεδομένων ήταν το phpmyadmin του xampp, το οποίο χρησιμοποιεί την MySQL σαν σχεσιακό σύστημα διαχείρισης βάσεων δεδομένων. Η επιλογή αυτή έγινε διότι το phpmyadmin είναι εύκολο στην χρήση, με αρκετές δυνατότητες(με βασικότερη την μετατροπή των queries σε γλώσσα PHP που χρειαστήκαμε για το development του UI) και υπερκαλύπτει τις ανάγκες της εργασίας(μπορούμε να δούμε όλα τα tables, views, triggers, indexes χωρίς κόπο).
Ο λόγος που επιλέχθηκε αντί του MySQL Workbench ήταν όμως η εύκολη σύνδεση του με την PHP για την δημιουργία του UI.
Πέραν όμως της πλατφόρμας, δύο μειονεκτήματα της υλοποίησής μας(ίσως να υπάρχουν και άλλα που δεν παρατηρήσαμε) ήταν το auto_increment σε κάποια primary keys(ή μέρη των primary keys) και το year(4) στις στήλες pubyear του book και estyear του publisher. Το πρώτο είναι μειονέκτημα όταν πάμε να περάσουμε στοιχεία μέσω του UI στο εκάστοτε table που έχει στήλη με auto_increment, γιατί για παράδειγμα αν έχουμε 5 authid από το 1 έως το 5 και διαγράψουμε ένα από αυτά και μετά πάμε να βάλουμε ένα νέο, αυτό δεν θα πάρει την τιμή του διαγραμμένου αλλά την τιμή 6. Το δεύτερο αποτελεί πρόβλημα όταν πάμε να βάλουμε σαν έτος αριθμό μικρότερο του 1901 και μεγαλύτερο του 2155, όπου σε αυτές τις περιπτώσεις(πχ αν βάλουμε τον αριθμό 1850 σαν έτος έκδοσης ενός βιβλίου) θα εμφανίσει τον αριθμό 0000.

Μέρος (ii) & (iii)

Σχεδιασμός της Βάσης στο Σχεσιακό Μοντέλο και εξήγηση των περιορισμών

```
CREATE TABLE `author` (
  `authid` int NOT NULL AUTO_INCREMENT,
  `authlastname` varchar(45) NOT NULL,
  `authfirstname` varchar(45) NOT NULL,
  `authbirthdate` date DEFAULT NULL,
  PRIMARY KEY (`authid`),
  CONSTRAINT `auth_un` UNIQUE (`authlastname`, `authfirstname`, `authbirthdate`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `publisher` (
  `pubname` varchar(50) NOT NULL,
  `estyear` year(4) NOT NULL,
  `pubstreet` varchar(45) NOT NULL,
  `pubnumber` smallint NOT NULL,
  `pubpostcode` varchar(5) DEFAULT NULL,
  PRIMARY KEY (`pubname`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `book` (
  `ISBN` varchar(17) NOT NULL,
  `title` varchar(100) NOT NULL,
  `pubyear` year(4) NOT NULL,
  `numpages` smallint NOT NULL,
  `pubname` varchar(50) NOT NULL,
  PRIMARY KEY (`ISBN`),
  /*UNIQUE KEY `USBN_UNIQUE` (`ISBN`),*/
  /*KEY `pub_book` (`pubname`),*/
  CONSTRAINT `pub_book` FOREIGN KEY (`pubname`) REFERENCES `publisher` (`pubname`)
ON UPDATE CASCADE ON DELETE RESTRICT
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `copy` (
  `ISBN` varchar(17) NOT NULL,
  `copynr` smallint NOT NULL AUTO_INCREMENT,
  `shelf` varchar(10) NULL,
  PRIMARY KEY (`copynr`,`ISBN`),
  /*UNIQUE KEY `copynr_UNIQUE` (`copynr`),*/
  /*KEY `ISBN_idx` (`ISBN`),*/
  CONSTRAINT `ISBN` FOREIGN KEY (`ISBN`) REFERENCES `book` (`ISBN`) ON DELETE
CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
CREATE TABLE `written_by` (
  `ISBN` varchar(17) NOT NULL,
  `authid` int NOT NULL,
  PRIMARY KEY (`ISBN`,`authid`),
  /*KEY `fk_book_has_author_author1_idx` (`author_authid`),
  KEY `fk_book_has_author_book1_idx` (`book_ISBN`),*/
  CONSTRAINT `written_book` FOREIGN KEY (`ISBN`) REFERENCES `book` (`ISBN`) ON
DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `written_author` FOREIGN KEY (`authid`) REFERENCES `author` (`authid`)
ON DELETE RESTRICT ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `category` (
  `catname` varchar(50) NOT NULL,
  `maincatname` varchar(50),
  PRIMARY KEY (`catname`),
  CONSTRAINT `cat_fk` FOREIGN KEY (`maincatname`) REFERENCES `category` (`catname`)
ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `member` (
  `memid` int NOT NULL AUTO_INCREMENT,
  `memfirstname` varchar(45) NOT NULL,
  `memlastname` varchar(45) NOT NULL,
  `membirthdate` date NOT NULL,
  `memstreet` varchar(45) NOT NULL,
  `memnr` smallint NOT NULL,
  `mempostcode` varchar(5) DEFAULT NULL,
  PRIMARY KEY (`memid`),
  CONSTRAINT `mem_un` UNIQUE (`memlastname`, `memfirstname`, `membirthdate`)
  /*UNIQUE KEY `memid_UNIQUE` (`memid`)*/
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `employee` (
  `empid` int NOT NULL AUTO_INCREMENT,
  `salary` int NOT NULL,
  `emplastname` varchar(45) NOT NULL,
  `empfirstname` varchar(45) NOT NULL,
  PRIMARY KEY (`empid`)
  /*UNIQUE KEY `empid_UNIQUE` (`empid`)*/
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `permanent` (
  `empid` int NOT NULL,
  `hiringdate` date NOT NULL,
  PRIMARY KEY (`empid`),
  CONSTRAINT `empidperm` FOREIGN KEY (`empid`) REFERENCES `employee` (`empid`) ON
DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
CREATE TABLE `temporary` (
  `empid` int NOT NULL,
  `contractnr` varchar(20) NOT NULL,
  PRIMARY KEY (`empid`),
  CONSTRAINT `empidtemp` FOREIGN KEY (`empid`) REFERENCES `employee` (`empid`) ON
DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `belongs_to` (
  `ISBN` varchar(17) NOT NULL,
  `catname` varchar(50) NOT NULL,
  PRIMARY KEY (`ISBN`,`catname`),
  /*KEY `fk_book_has_category_category1_idx` (`category_catname`),
  KEY `fk_book_has_category_book1_idx` (`book_ISBN`),*/
  CONSTRAINT `book_bel` FOREIGN KEY (`ISBN`) REFERENCES `book` (`ISBN`) ON DELETE
CASCADE ON UPDATE CASCADE,
  CONSTRAINT `cat_bel` FOREIGN KEY (`catname`) REFERENCES `category` (`catname`) ON
DELETE RESTRICT ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `borrows` (
  `memid` int NOT NULL,
  `copynr` smallint NOT NULL,
  `ISBN` varchar(17) NOT NULL,
  `date_of_borrowing` date NOT NULL,
  `date_of_return` date DEFAULT NULL,
  `due_date` date DEFAULT NULL,
  PRIMARY KEY (`memid`,`copynr`,`ISBN`,`date_of_borrowing`),
  /*KEY `fk_member_has_copy_copy1_idx` (`copy_copynr`,`copy_ISBN`),
  KEY `fk_member_has_copy_member1_idx` (`member_memid`),*/
  CONSTRAINT `mem_borrows` FOREIGN KEY (`memid`) REFERENCES `member` (`memid`) ON
DELETE RESTRICT ON UPDATE CASCADE,
  CONSTRAINT `mem_book` FOREIGN KEY (`ISBN`) REFERENCES `book` (`ISBN`) ON DELETE
RESTRICT ON UPDATE CASCADE,
  CONSTRAINT `copy_borrows` FOREIGN KEY (`copynr`, `ISBN`) REFERENCES `copy`
(`copynr`, `ISBN`) ON DELETE RESTRICT ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
CREATE TABLE `reminder` (
  `empid` int NOT NULL,
  `memid` int NOT NULL,
  `ISBN` varchar(17) NOT NULL,
  `copynr` smallint NOT NULL,
  `date_of_borrowing` date NOT NULL,
  `date_of_reminder` date NOT NULL,
  PRIMARY KEY
(`empid`,`memid`,`ISBN`,`copynr`,`date_of_borrowing`,`date_of_reminder`),
  /*KEY `rem_borrow` (`memberid`,`copynr`,`rem_ISBN`,`date_of_borrowing`),
  KEY `rem_copy` (`rem_ISBN`,`copynr`),*/
  CONSTRAINT `fk_reminder_1` FOREIGN KEY (`empid`) REFERENCES `employee` (`empid`)
ON DELETE RESTRICT ON UPDATE CASCADE,
  CONSTRAINT `rem_borrow` FOREIGN KEY (`memid`, `copynr`, `ISBN`,
`date_of_borrowing`) REFERENCES `borrows` (`memid`, `copynr`, `ISBN`,
`date_of_borrowing`) ON DELETE RESTRICT ON UPDATE CASCADE,
  CONSTRAINT `rem_isbn` FOREIGN KEY (`ISBN`) REFERENCES `book` (`ISBN`) ON DELETE
RESTRICT ON UPDATE CASCADE,
  CONSTRAINT `rem_mem` FOREIGN KEY (`memid`) REFERENCES `member` (`memid`) ON
DELETE RESTRICT ON UPDATE CASCADE,
  CONSTRAINT `rem_copy` FOREIGN KEY (`copynr`, `ISBN`) REFERENCES `copy` (`copynr`,
`ISBN`) ON DELETE RESTRICT ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Περιορισμοί - Ενέργειες που γίνονται στη βάση για να υποστηρίζονται κατά τις αλλαγές

Όσον αφορά τους περιορισμούς ακεραιότητας, αυτοί επιλέχθηκαν ως εξής:

- Τα μοναδικά Id των συγγραφέων, των εργαζομένων, των μελών κλπ τέθηκαν ως ακέραιοι αριθμοί και ως AUTO_INCREMENT έτσι ώστε να μη χρειάζεται κάθε φορά που προσθέτουμε ένα στοιχείο στον αντίστοιχο πίνακα να θυμόμαστε ποιος ήταν ο τελευταίος αύξων αριθμός, αλλά αυτός να συμπληρώνεται από μόνος του
- Όλα τα ονόματα (ονοματεπώνυμα, τίτλοι βιβλίων, ονόματα εκδοτικών οίκων, ονόματα οδών) είναι string από χαρακτήρες με μήκη ανάλογα με το είδος του ονόματος. Οι αριθμοί των οδών ορίστηκαν ως smallint αφού είναι σχεδόν αδύνατο να πάρουν μια τόσο μεγάλη τιμή ώστε να συμβεί κάποιο overflow. Οι ταχυδρομικοί κώδικες ορίστηκαν ως string 5 χαρακτήρων, αφού στη χώρα μας έχουν μορφή πενταψήφιου αριθμού.
- Οι ημερομηνίες και οι χρονιές τέθηκαν ως date και year(4) αντίστοιχα. Έτσι, ορίζοντας κατάλληλο User Interface θα μπορούσε κανείς να τις εισάγει μέσω ημερολογίου.
- Όσον αφορά τις NULL τιμές, απαιτήσαμε από τα περισσότερα πεδία να συμπληρώνονται υποχρεωτικά, πέραν ελαχίστων εξαιρέσεων, όπως για παράδειγμα ο ταχυδρομικός κώδικας ή το ράφι ενός αντιγράφου σε περίπτωση που ο βιβλιοθηκάριος δεν έχει από την αρχή πληροφόρηση για το πού θα πρέπει να μπει.
- Τα πρωτεύοντα κλειδιά προφανώς δεν επιτρέπεται να είναι κενά, όπως και τα ξένα κλειδιά, τα οποία ορίσαμε με βάση το σχεσιακό μοντέλο που δόθηκε.
- Κάποια πεδία ή συνδυασμός πεδίων ορίστηκαν ως UNIQUE. Για παράδειγμα, δε μπορούμε να έχουμε δύο συγγραφείς ή δύο μέλη που έχουν το ίδιο ονοματεπώνυμο και ημερομηνία γέννησης.
- Συγκεκριμένα, για το ISBN του βιβλίου, αυτό τέθηκε ως string 17 χαρακτήρων γιατί βρέθηκε πως αποτελείται από 4 μέρη 10 συνολικών χαρακτήρων ή από 5 μέρη 13 χαρακτήρων συνολικά (τα μέρη είναι χωρισμένα μεταξύ του με μία παύλα).

Ευρετήρια

Κατά την δημιουργία των tables με το εργαλείο phpmyadmin του xampp(που χρησιμοποιεί MySQL) σχηματίζονται indexes σε όλα τα primary keys(πχ στον πίνακα reminder όλα τα columns μαζί σχηματίζουν το index primary) και σε όλα τα foreign και unique keys(όπου έχουμε CONSTRAINT στο DDL).
Επιπλέον, δημιουργούμε μερικά ακόμη ευρετήρια για επιτάχυνση κατά την αναζήτηση σε συγκεκριμένα columns:

```
CREATE INDEX book_title_year ON book (`title`, `pubyear`)
CREATE INDEX bor_late ON borrows (`date_of_return`, `due_date`)
CREATE INDEX employee_name ON employee (`emplastname`, `empfirstname`)
CREATE INDEX rem_reminder ON reminder (`date_of_reminder`)
```

Ερωτήσεις - Queries

- 1st Query(aggregate)
  Βρίσκουμε τον μέσο μισθό των εργαζομένων (μονίμων και συμβασιούχων).

```
SELECT AVG(E.salary)
FROM employee AS E, permanent AS P
WHERE E.empid=P.empid

SELECT AVG(E.salary)
FROM employee AS E, temporary AS T
WHERE E.empid=T.empid
```

- 2nd Query(aggregate)
  Βρίσκουμε τον αριθμό των μελών της βιβλιοθήκης

```
SELECT COUNT(*) AS Total_Members
FROM member
```

- 3rd Query(order by)
  Βρίσκουμε τα ονοματεπώνυμα των μελών της βιβλιοθήκης (και το unique id) που έχουν καθυστερήσει να επιστρέψουν βιβλίο ή δεν έχουν επιστρέψει ακόμη και τα ταξινομούμε κατά επώνυμο.

```
SELECT M.memid, M.memlastname, M.memfirstname
FROM member AS M, borrows AS B
WHERE M.memid=B.memid AND (B.due_date<B.date_of_return OR B.date_of_return IS NULL)
ORDER BY M.memlastname
```

- 4th Query(group by)
  Βρίσκουμε τα βιβλία που είναι διαθέσιμα (έστω 1) και τυπώνουμε το ISBN, τον τίτλο, τον συγγραφέα και τις εκδόσεις τους.

```
SELECT C.ISBN, B.title, A.authlastname, A.authfirstname, B.pubname
FROM borrows AS BR, copy AS C, book AS B, author AS A, written_by AS W
WHERE B.ISBN=C.ISBN AND C.ISBN=BR.ISBN AND C.copynr=BR.copynr
      AND BR.date_of_return IS NOT NULL AND A.authid=W.authid
           AND W.ISBN=C.ISBN
GROUP BY B.title
```

- 5th Query(group by-having)
  Βρίσκουμε τα άτομα που έχουν δανειστεί 5 βιβλία και δεν τα έχουν επιστρέψει ακόμα. Αυτό γίνεται προκειμένου να μην μπορούν να δανειστούν και 6ο, όπως έλεγε και η εκφώνηση της 1ης άσκησης.

```
SELECT M.memid, M.memlastname, M.memfirstname
FROM member AS M, borrows AS B
WHERE M.memid=B.memid AND B.date_of_return IS NULL
GROUP BY memid
HAVING COUNT(*)=5
ORDER BY memlastname
```

- 6th Query(inner joins)
  Βρίσκουμε όλα τα στοιχεία για κάθε βιβλίο (ISBN, τίτλο, χρονιά έκδοσης, σελίδες, ονοματεπώνυμο συγγραφέα, εκδότη και κατηγορία)

```
SELECT B.ISBN, B.title, C.copynr, B.pubyear, B.numpages, A.authlastname,
A.authfirstname, B.pubname, Cat.catname
FROM book B
     INNER JOIN copy C ON C.ISBN=B.ISBN
     INNER JOIN belongs_to BT ON BT.ISBN=C.ISBN
     INNER JOIN category Cat ON Cat.catname=BT.catname
     INNER JOIN written_by WB ON WB.ISBN=B.ISBN
     INNER JOIN author A ON A.authid=WB.authid
ORDER BY B.title
```

- 7th Query(inner joins+left join)
  Βρίσκουμε τα στοιχεία κάθε μέλους που έχει δανειστεί βιβλίο, το βιβλίο που δανείστηκε, πότε το δανείστηκε, μέχρι πότε πρέπει να το επιστρέψει, και αν του έχει σταλεί reminder. Όλα αυτά εμφανίζονται μόνο στην περίπτωση που δεν το έχει επιστρέψει. (Το left-join γίνεται για να εμφανίζονται οι περιπτώσεις που δεν έχει σταλεί reminder, κάτι που είναι πολύ σημαντικό ώστε να ξέρουν να του στείλουν.)

```
SELECT M.memid, M.memlastname, M.memfirstname, B.ISBN, B.title, C.copynr,
BR.date_of_borrowing, R.date_of_reminder, BR.due_date
FROM member M
     INNER JOIN borrows BR ON BR.memid=M.memid
     INNER JOIN book B ON B.ISBN=BR.ISBN
     INNER JOIN copy C ON C.ISBN=B.ISBN
     LEFT JOIN reminder R ON R.memid=M.memid
WHERE BR.date_of_return IS NULL
GROUP BY M.memlastname
```

- 8th Query(nested query)
  Βρίσκουμε τα στοιχεία των υπαλλήλων που δεν έχουν στείλει ειδοποίηση σε κάποιον για επιστροφή βιβλίου (ώστε να μην στέλνουν συνέχεια οι ίδιοι) αλλά και να αμείβεται με λιγότερα από 1500. (Θεωρούμε ότι με παραπάνω είναι υψηλόβαθμος οπότε δεν είναι αρμοδιότητά του. Αυτό εννοείται πως είναι δικιά μας παραδοχή.)

```
SELECT empid, emplastname, empfirstname
FROM employee
WHERE salary<1500 AND empid NOT IN (SELECT DISTINCT empid
                                    FROM reminder)
```

Όψεις - Views

- Updatable view: Εμφανίζεται το ISBN και το copynr ενός αντιτύπου χωρίς το ράφι στο οποίο βρίσκεται, καθώς το shelf μπορεί να πάρει και την τιμή NULL. (Αυτό έγκειται στο γεγονός ότι μπορεί ένα αντίτυπο να έχει έρθει στη βιβλιοθήκη αλλά να μην έχει προλάβει ακόμη να τοποθετηθεί σε κάποιο συγκεκριμένο ράφι). Άρα, προσθέτοντας μια tuple στο συγκεκριμένο view θα αναπαρασταθεί και στον πίνακα copy με τιμή στο self=NULL. Επομένως η όψη είναι updatable.

```
CREATE VIEW `copy_view` AS
SELECT ISBN, copynr
FROM copy
```

- Non Updatable view: Εμφανίζεται ο συνολικός αριθμός των βιβλίων που έχουν δανειστεί ανά ημέρα. Αυτό μπορεί να φανεί ιδιαίτερα χρήσιμο στη αποτίμηση στατιστικών που πιθανόν να ενδιαφέρουν την βιβλιοθήκη. Είναι προφανές ότι αφού εμπεριέχεται aggregate και group-by clause το view θα είναι σίγουρα non updatable.

```
CREATE VIEW `books_borrowed_per_day` AS
SELECT COUNT(copynr) AS Books_Borrowed, date_of_borrowing
FROM borrows
GROUP BY date_of_borrowing
```

Triggers

- Για κάθε νέα εισαγωγή στον πίνακα borrows συμπληρώνεται αυτόματα το πεδίο due_date με τιμή 30 ημέρες ύστερα από την ημερομηνία δανεισμού.

```
CREATE TRIGGER `trigg_due_date`
BEFORE INSERT ON `borrows`
FOR EACH ROW
	SET new.due_date = date_add(new.date_of_borrowing, INTERVAL 30 day);
```

- Τα δύο επόμενα triggers υλοποιούν το disjoint specialization ανάμεσα στον μόνιμο και τον συμβασιούχο υπάλληλο. Συγκεκριμένα, αν για παράδειγμα θέλουμε να εισάγουμε στον πίνακα permanent έναν υπάλληλο του οποίου το id υπάρχει ήδη στον πίνακα temporary, τότε το id της νέας γραμμής τίθεται ως NULL, με αποτέλεσμα να μη καθίσταται δυνατή η εισαγωγή της νέας σειράς στη βάση, αφού το πεδίο id πρέπει να είναι NOT NULL.

```
DELIMITER $$
CREATE TRIGGER `perm_trigg`
BEFORE INSERT ON `permanent`
FOR EACH ROW
	IF NEW.empid IN (SELECT empid FROM temporary)
		THEN SET NEW.empid = NULL;
	END IF;
$$
DELIMITER ;

DELIMITER $$
CREATE TRIGGER `temp_trigg`
BEFORE INSERT ON `temporary`
FOR EACH ROW
	IF NEW.empid IN (SELECT empid FROM permanent)
		THEN SET NEW.empid = NULL;
	END IF;
$$
DELIMITER ;
```