ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ
ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

Εξαμηνιαίο Project

Ακαδημαϊκό Έτος: 2012-12013

Σαρλής Δημήτριος (03109078)

Σταθακοπούλου Χρυσούλα (03109065)

Τζαννέτος Δημήτριος (03109010)

ΛΕΠΤΟΜΕΡΙΕΣ ΥΛΟΠΟΙΗΣΗΣ

Για την υλοποίηση του project χρησιμοποιήσαμε:

- Σύστημα Διαχείρισης Βάσεων Δεδομένων: MySql
- Δημιουργία User Interface: php
- Μορφοποίηση User Interface: css
- Web Servrer: Apache
- Λειτουργικό σύστημα: Linux (Ubuntu 12.04)

Επιπλέον χρησιμοποιήθηκε το phpMyAdmin για το σχεδιασμό της βάσης και την εισαγωγή των στοιχείων της και η InnoDB για DataBase Storage Engine.

Τέλος για την υλοποίηση του User Interface χρησιμοποιήθηκε η MVC αρχιτεκτονική (Model/View/Controller). Αναλυτικότερα μέσω του model γίνεται η επικοινωνία με τη βάση δεδομένων. Το view καθορίζει τη διεπαφή με το χρήστη. Τέλος ο controller υλοποιεί την επικοινωνία μεταξύ των views και models.

Πλεονεκτήματα:

Οι επιλογές μας έχουν τα εξής πλεονεκτήματα:

mySQL:

- Σύστημα Διαχείρισης Σχεσιακών Βάσεων Δεδομένων (RDBMS) ανοιχτού κώδικα σε ευρεία χρήση, γνωστό για την ταχύτητα και αξιοπιστία του, με ενεργή κοινότητα ανάπτυξης και συντήρησής του. Έτσι, εύκολα μπορούσαμε να έχουμε πρόσβαση σε λύσεις για τα προβλήματα που αντιμετωπίζαμε με αναζήτηση στο διαδίκτυο.
- Είναι συμβατή και με Linux σε αντίθεση με τον SQL Server που είναι συμβατός μόνο με τα Windows.

php:

- Συμβατή με τη mySQL ώστε να μπορούμε μέσα από την php να καλούμε απευθείας ερωτήματα στη βάση.
- Εύκολη Εκμάθηση
- Ευνοεί την παράλληλη ανεξάρτητη ανάπτυξη λογισμικού το οποίο εύκολα συντίθεται στο τελικό αποτέλεσμα.

phpMyAdmin:

- Γρήγορη εγκατάσταση
- Εύκολο στη χρήση.
- Δίνει τη δυνατότητα του σχεδιασμού της βάσεις μέσω User Interface ώστε να αποφεύγονται συντακτικά λάθη που προκύπτουν με την απευθείας σύνταξη κώδικα.
- Επιτρέπει την εξαγωγή του κώδικα σχεδιασμού της βάσης.
- Υποστηρίζει τη δυνατότητα απευθείας εισαγωγής κώδικα για το σχεδιασμό λεπτομερειών της βάσης όπως τα constraints.

InnoDB Storage Engine:

- Αξιόπιστη και υψηλής απόδοσης Storage Engine για την MySQL
- Σχεδιασμένη σύμφωνα με το μοντέλο ACID . Οι συναλλαγές υποστηρίζουν τις λειτουργίες commit , roll-back και crash-recovery προστατεύοντας τα δεδομένα .
- Αυτόματα οργανώνει τα δεδομένα στο δίσκο σε clustered indeces έτσι ώστε να βελτιστοποιεί συνηθισμένα queries σε πρωτεύοντα κλειδιά, ελαχιστοποιώντας τις λειτουργίες I/O για την προσκόμιση των κλειδιών.
- Υποστηρίζει Αναφορική Ακεραιότητα Ξένου Κλειδιού

MVC αρχιτεκτονική:

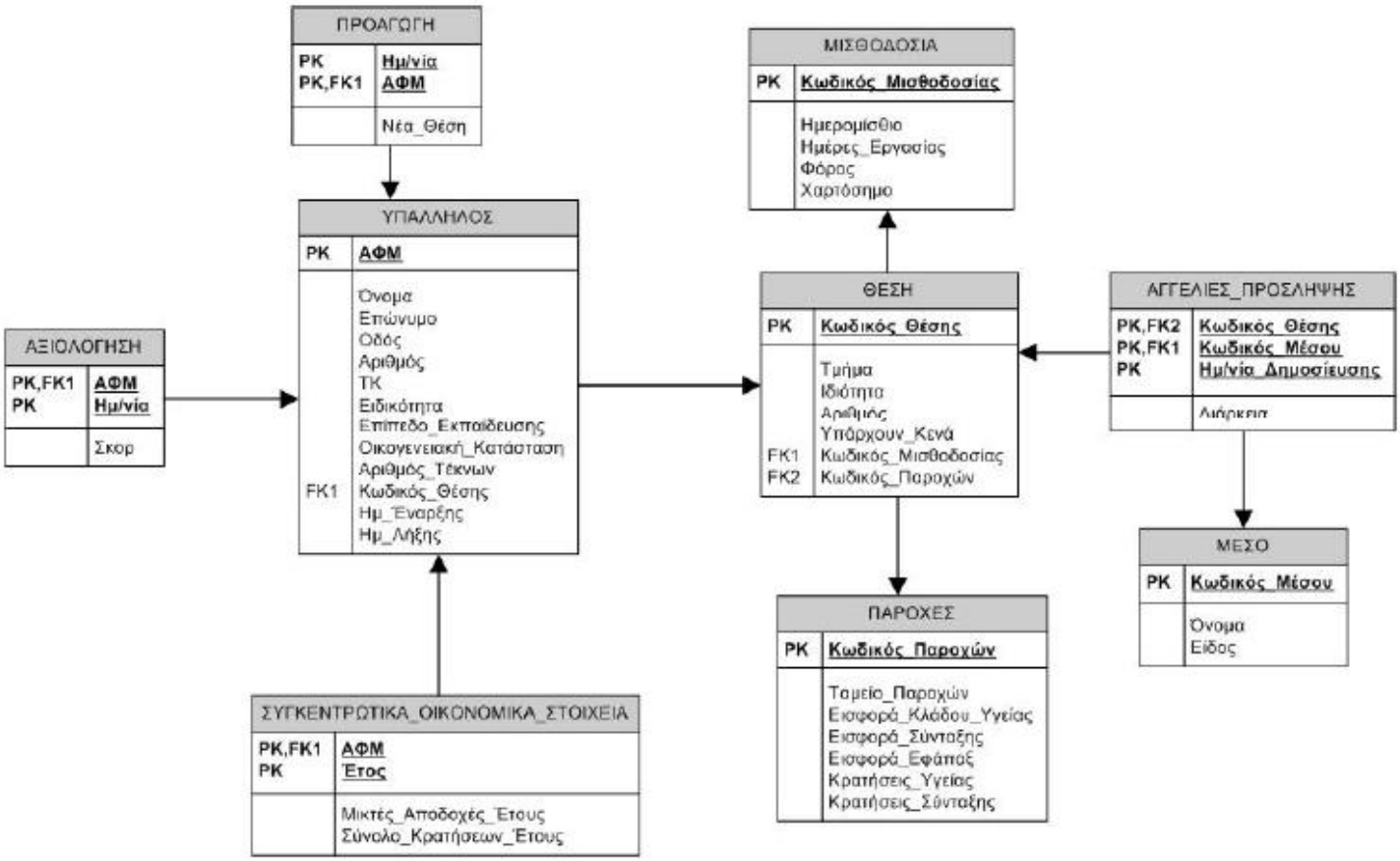
- Παρέχει οργάνωση που διευκολύνει τη κλιμάκωση της εφαρμογής, τις αλλαγές και τη συντήρηση κώδικα.
- Επιτρέπει την παράλληλη και συνεπώς ταχύτερη ανάπτυξη κώδικα.

Μειονεκτήματα:

- Η mySQL δεν υποστηρίζει των έλεγχο των constraints ο οποίος θα πρέπει να γίνεται μέσω της php.
- Η InnoDB storage engine είναι λίγο πιο αργή σε σχέση με την MyISAM σε περιπτώσεις όπου χρειάζεται να κάνουμε πολλές αναγνώσεις δεδομένων.

ΣΧΕΔΙΑΣΜΟΣ ΤΗΣ ΒΑΣΗΣ

Ο σχεδιασμός της Βάσης Δεδομένων της GREX ΑΕ έχει γίνει σύμφωνα σχεσιακό μοντέλο που αναπαρίσταται στο κάτωθι σχήμα:



Περιορισμοί Ακεραιότητας

Σε κάθε σχέση που περιέχει ξένο κλειδί υπάρχει ο περιορισμός ξένου κλειδιού, ώστε το πεδίο που αποτελεί το ξένο κλειδί να συμπληρώνεται υποχρεωτικά από υπαρκτή τιμή του αντίστοιχου πίνακα στον οποίο είναι πρωτεύον κλειδί.

Έχουν εισαχθεί περιορισμοί ακεραιότητας *NOT NULL* έτσι ώστε να μην μπορούν να τεθούν NULL οι αντίστοιχες τιμές.

Επιπλέον στα ξένα κλειδιά κάθε σχέσης υπάρχουν περιορισμοί ακεραιότητας που καθορίζουν τι γίνεται στα *DELETE* και τα *UPDATE*. Συγκεκριμένα:

Για τις σχέσεις *EVALUATION , PROMOTION και AGGREGATED_FINANCIAL_DATA* που έχουν εξωτερικό κλειδί το *SSN* της σχέσης *EMPLOYEE* επιλέγουμε *ON DELETE CASCADE* , *ώστε* κατά τη διαγραφή του υπαλλήλου να διαγραφούν και τα αντίστοιχα πεδία των σχέσεων που αφορούν αυτόν τον υπάλληλο, και *ON UPDATE NO ACTION* επειδή θεωρούμε ότι το SSN (ΑΦΜ) ενός ανθρώπου γενικά δεν αλλάζει.

Για τη σχέση *EMPLOYEE* που έχει εξωτερικό κλειδί το *pid* (κωδικός θέσης) της σχέσης *POSITION* επιλέγουμε *ON DELETE SET NULL* , ώστε όταν καταργηθεί μία θέση να μην διαγραφούν οι υπάλληλοι της εταιρίας που έχουν αυτή τη θέση αλλά το *pid* τους να τεθεί *NULL* . Επιλέγουμε ωστόσο *ON UPDATE CASCADE* ώστε όταν αλλάξει ο κωδικός θέσης μίας θέσης να τροποποιηθεί κατάλληλα το αντίστοιχο πεδίο του υπαλλήλου.

Αντίστοιχα για τη σχέση *POSITION* που έχει εξωτερικά κλειδιά το *sal_id* (*κωδικό* μισθοδοσίας) και το *ben_id* (κωδικό παροχών) των σχέσεων *SALARY, BENEFITS* αντίστοιχα επιλέγουμε *ON DELETE SET NULL ON UPDATE CASCADE* ώστε να μην διαγράφεται μια θέση όταν καταργείται ένα πρόγραμμα μισθοδοσίας ή παροχών.

Τέλος για τη σχέση *RECRUITMENT_AD* που έχει εξωτερικά κλειδιά το *pid* (κωδικός θέσης) και *media_id* (κωδικός μέσου) επιλέγουμε *ON DELETE CASCADE* ώστε όταν καταργηθεί μία θέση ή ένα μέσο να καταργείται και η αντίστοιχη αγγελία και *ON UPDATE CASCADE* ώστε αν αλλάξουν ο κωδικός μέσου ή ο κωδικός θέσης να τροποποιείται η αγγελία κατάλληλα.

Ευρετήρια

Κατασκευάζονται πρωτεύοντα ευρετήρια στο primary key κάθε σχέσης καθώς είναι ένα πεδίο στο όποιο γίνεται συχνά πρόσβαση, για παράδειγμα σε λειτουργίες διαγραφής (delete) , ενημέρωσης (update) και σε joints.

Κατασκευάζονται ευρετήρια στα πεδία των σχέσεων που είναι εξωτερικά κλειδιά καθώς σ' αυτά πραγματοποιούνται συχνά προσπελάσεις για σύνδεση των πινάκων (στο πεδίο “where” των ερωτημάτων).

Σε ένα σύστημα έχει νόημα να εισάγουμε ευρετήρια και σε πεδία που προσπελάζονται συχνά σειριακά σε ερωτήματα που περιλαμβάνουν συγκρίσεις. Ενδεικτικά φτιάχνουμε ένα τέτοιο ευρετήριο στη στήλη *from_date* (ημερομηνία πρόσληψης) του πίνακα *EMPLOYEE* που μπορούμε να χρησιμοποιήσουμε στην κατασκευή της όψης με τους υπαλλήλους που προσελήφθησαν το 2012.

```
ALTER TABLE `EMPLOYEE` ADD INDEX `recruit_index` (`from_date`)
```

Γενικά, η InnoDB storage engine φροντίζει μόνη της να κατασκευάσει ευρετήρια στα πρωτεύοντα κλειδιά κάθε σχέσης, γιατί αυτά είναι χρήσιμα πεδία για πρόσβαση στις καταχωρήσεις της βάσης.

ΚΑΤΑΣΚΕΥΗ ΒΑΣΗΣ

Παρακάτω παρατίθενται τα DDL που κατασκευάζουν τη βάση:

```
--
-- Database: `GREX AE`
--

-- --------------------------------------------------------

--
-- Table structure for table `AGGREGATED_FINANCIAL_DATA`
--

CREATE TABLE IF NOT EXISTS `AGGREGATED_FINANCIAL_DATA` (
  `SSN` int(11) NOT NULL,
  `year` year(4) NOT NULL,
  `gross_year_earnings` float NOT NULL,
  `total_year_holdings` float NOT NULL,
  PRIMARY KEY (`SSN`,`year`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- --------------------------------------------------------

--
-- Table structure for table `BENEFITS`
--

CREATE TABLE IF NOT EXISTS `BENEFITS` (
  `ben_id` int(11) NOT NULL,
  `fund_benefits` varchar(20) NOT NULL,
  `health_contribution` float NOT NULL,
  `pension_contribution` float NOT NULL,
  `lump_sum_contribution` float NOT NULL,
  `health_tax` float NOT NULL,
  `pension_tax` float NOT NULL,
  PRIMARY KEY (`ben_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;


-- --------------------------------------------------------

--
-- Table structure for table `EMPLOYEE`
--

CREATE TABLE IF NOT EXISTS `EMPLOYEE` (
  `SSN` int(11) NOT NULL,
  `name` varchar(20) NOT NULL,
  `surname` varchar(20) NOT NULL,
  `street` varchar(50) NOT NULL,
  `number` varchar(5) NOT NULL,
  `PC` int(11) NOT NULL,
  `specialization` varchar(20) NOT NULL,
  `education_level` varchar(20) NOT NULL,
  `marital_status` varchar(20) NOT NULL,
  `children_no` int(11) NOT NULL,
  `pid` int(11),
```

```
  `from_date` date NOT NULL,
  `to_date` date DEFAULT NULL,
  PRIMARY KEY (`SSN`),
  KEY `pid` (`pid`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- --------------------------------------------------------


--
-- Table structure for table `EVALUATION`
--

CREATE TABLE IF NOT EXISTS `EVALUATION` (
  `SSN` int(11) NOT NULL,
  `date` date NOT NULL,
  `score` int(11) NOT NULL,
  PRIMARY KEY (`SSN`,`date`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;


-- --------------------------------------------------------

--
-- Table structure for table `MEDIA`
--

CREATE TABLE IF NOT EXISTS `MEDIA` (
  `media_id` int(11) NOT NULL,
  `name` varchar(50) NOT NULL,
  `kind` varchar(20) NOT NULL,
  PRIMARY KEY (`media_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;


-- --------------------------------------------------------

--
-- Table structure for table `POSITION`
--

CREATE TABLE IF NOT EXISTS `POSITION` (
  `pid` int(11) NOT NULL,
  `department` varchar(20) NOT NULL,
  `role` varchar(20) NOT NULL,
  `number` int(11) NOT NULL,
  `is_empty` int(11) NOT NULL,
  `sal_id` int(11),
  `ben_id` int(11),
  PRIMARY KEY (`pid`),
  KEY `sal_id` (`sal_id`),
  KEY `ben_id` (`ben_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `POSITION`
--

INSERT INTO `POSITION` (`pid`, `department`, `role`, `number`,
`is_empty`, `sal_id`, `ben_id`) VALUES
(1, 'General Management', 'Executive Manager', 1, 0, 1, 1),
```

```
(2, 'General Management', 'Assistant Manager', 2, 0, 2, 2),
(3, 'General Management', 'Secretary', 3, 0, 7, 7),
(4, 'Personnel', 'Manager', 1, 0, 3, 3),
(5, 'Personnel', 'Secretary', 2, 1, 8, 8),
(6, 'Personnel', 'Psychologist', 1, 0, 5, 5),
(7, 'Sales', 'Manager', 1, 0, 3, 3),
(8, 'Sales', 'Secretary', 2, 0, 8, 8),
(9, 'Sales', 'Advertiser', 2, 0, 6, 6),
(10, 'Sales', 'Salesman', 3, 1, 6, 6),
(11, 'Technical', 'Programmer', 3, 1, 5, 5),
(12, 'Technical', 'Web Developer', 4, 1, 5, 5),
(13, 'Technical', 'Project Manager', 1, 0, 3, 3),
(14, 'PR', 'Manager', 1, 0, 3, 3),
(15, 'PR', 'Secretary', 2, 0, 8, 8),
(16, 'Finance', 'Accountant', 2, 0, 6, 6),
(17, 'Finance', 'Analyst', 1, 0, 4, 4);


-- --------------------------------------------------------

--
-- Table structure for table `PROMOTION`
--

CREATE TABLE IF NOT EXISTS `PROMOTION` (
  `date` date NOT NULL,
  `SSN` int(11) NOT NULL,
  `new_pos` int(11) NOT NULL DEFAULT '0',
  PRIMARY KEY (`date`,`SSN`),
  KEY `SSN` (`SSN`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- --------------------------------------------------------

--
-- Table structure for table `RECRUITMENT_AD`
--

CREATE TABLE IF NOT EXISTS `RECRUITMENT_AD` (
  `pid` int(11) NOT NULL,
  `media_id` int(11) NOT NULL,
  `publication_date` date NOT NULL,
  `duration` int(11) NOT NULL,
  PRIMARY KEY (`pid`,`media_id`,`publication_date`),
  KEY `media_id` (`media_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- --------------------------------------------------------

--
-- Table structure for table `SALARY`
--

CREATE TABLE IF NOT EXISTS `SALARY` (
  `sal_id` int(11) NOT NULL,
  `wage` float NOT NULL,
  `work_days` int(11) NOT NULL,
  `tax` float NOT NULL,
  `stamp` float NOT NULL,
  PRIMARY KEY (`sal_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

```
--
-- Stand-in structure for view `wage_stats`
--
CREATE TABLE IF NOT EXISTS `wage_stats` (
`department` varchar(20)
,`avg(wage)` double
);
-- --------------------------------------------------------

--
-- Constraints for dumped tables
--

--
-- Constraints for table `AGGREGATED_FINANCIAL_DATA`
--
ALTER TABLE `AGGREGATED_FINANCIAL_DATA`
  ADD CONSTRAINT `AFD1` FOREIGN KEY (`SSN`) REFERENCES `EMPLOYEE`
(`SSN`) ON DELETE CASCADE ON UPDATE NO ACTION;

--
-- Constraints for table `EMPLOYEE`
--
ALTER TABLE `EMPLOYEE`
  ADD CONSTRAINT `Emp1` FOREIGN KEY (`pid`) REFERENCES `POSITION`
(`pid`) ON DELETE SET NULL ON UPDATE CASCADE;

--
-- Constraints for table `EVALUATION`
--
ALTER TABLE `EVALUATION`
  ADD CONSTRAINT `Eval1` FOREIGN KEY (`SSN`) REFERENCES `EMPLOYEE`
(`SSN`) ON DELETE CASCADE ON UPDATE NO ACTION;

--
-- Constraints for table `POSITION`
--
ALTER TABLE `POSITION`
  ADD CONSTRAINT `Pos1` FOREIGN KEY (`sal_id`) REFERENCES `SALARY`
(`sal_id`) ON DELETE SET NULL ON UPDATE CASCADE,
  ADD CONSTRAINT `Pos2` FOREIGN KEY (`ben_id`) REFERENCES `BENEFITS`
(`ben_id`) ON DELETE SET NULL ON UPDATE CASCADE;


--
-- Constraints for table `PROMOTION`
--
ALTER TABLE `PROMOTION`
  ADD CONSTRAINT `Pro1` FOREIGN KEY (`SSN`) REFERENCES `EMPLOYEE`
(`SSN`) ON DELETE CASCADE ON UPDATE NO ACTION;

--
-- Constraints for table `RECRUITMENT_AD`
--
ALTER TABLE `RECRUITMENT_AD`
  ADD CONSTRAINT `Rec1` FOREIGN KEY (`pid`) REFERENCES `POSITION`
(`pid`) ON DELETE CASCADE ON UPDATE CASCADE,
  ADD CONSTRAINT `Rec2` FOREIGN KEY (`media_id`) REFERENCES `MEDIA`
(`media_id`) ON DELETE CASCADE ON UPDATE CASCADE;
```

Στα DDL περιλαμβάνονται επιπλέον τα insert για να καταχωρηθούν οι εγγραφές στους πίνακες. Στην αναφορά περιλαμβάνονται ενδεικτικά τα insert μόνο για τον πίνακα POSITION. Ο πλήρης κώδικας υπάρχει στο CD με τον κώδικα για το project.

Triggers

Κάθε βάση δεδομένων πρέπει να έχει triggers που φροντίζουν για την εσωτερική συνέπεια της βάσης. Στα πλαίσια αυτά κατασκευάσαμε ενδεικτικά τα ακόλουθα triggers:

Κατασκευάσαμε ένα trigger ώστε όταν ενημερώνεται ο πίνακας *PROMOTION* να ενημερώνεται κατάλληλα η θέση του υπαλλήλου τον οποίο αφορά η προαγωγή αλλά και ο αριθμός θέσεων της νέας και παλιά θέσης του και ο αριθμός κενών θέσεων. Συγκεκριμένα ο κωδικός θέσης του υπαλλήλου παίρνει τον κωδικό θέσης της θέσης για την οποία πήρε προαγωγή (εφόσον οι δύο κωδικοί είναι διαφορετικοί). Επιπλέον αυξάνεται κατά μία θέση ο αριθμός άδειων θέσεων για την παλιά θέση του υπαλλήλου. Ο αριθμός θέσεων της νέας θέσης που πήρε ο υπάλληλος αυξάνεται κατά μία ενώ ο αριθμός κενών θέσεων της νέας θέσης στην οποία πήγε ο υπάλληλος μειώνεται κατά μία (αν είναι θετικός – διαφορετικά παραμένουν 0)

```
CREATE TRIGGER `promo_update` AFTER UPDATE ON `PROMOTION`
 FOR EACH ROW begin
    update EMPLOYEE
    set EMPLOYEE.pid = NEW.new_pos
    where EMPLOYEE.SSN = NEW.SSN and NEW.new_pos > 0 and NEW.new_pos
<> OLD.new_pos;
    update POSITION
    set POSITION.is_empty = POSITION.is_empty+1
    where POSITION.pid = OLD.new_pos and NEW.new_pos > 0 and
NEW.new_pos <> OLD.new_pos;
    update POSITION
    set POSITION.number = POSITION.number+1
    where POSITION.pid = NEW.new_pos and POSITION.is_empty = 0 and
NEW.new_pos > 0 and NEW.new_pos <> OLD.new_pos;
    update POSITION
    set POSITION.is_empty = POSITION.is_empty-1
    where POSITION.pid = NEW.new_pos and POSITION.is_empty > 0 and
NEW.new_pos > 0 and NEW.new_pos <> OLD.new_pos;

end
```

Το δεύτερο trigger που κατασκευάσαμε καταργεί τις αγγελίες πρόσληψης που υπάρχουν για μία θέση όταν δεν υπάρχουν κενές θέσεις για τη θέση αυτή.

```
CREATE TRIGGER `advert_delete` AFTER UPDATE ON `POSITION`
 FOR EACH ROW delete from RECRUITMENT_AD
where RECRUITMENT_AD.pid = NEW.pid and NEW.is_empty = 0
```

Όψεις

Έχουν κατασκευαστεί ενδεικτικά δύο όψεις: μία ενημερώσιμη και μία μη ενημερώσιμη.

Ενημερώσιμη Όψη

Κατασκευάσαμε μια όψη με τους υπαλλήλους που προσελήφθησαν μέσα στο 2012.

```
CREATE VIEW `employees_2012` AS
select `EMPLOYEE`.`SSN` AS `SSN`,
       `EMPLOYEE`.`name` AS `name`,
       `EMPLOYEE`.`surname` AS `surname`,
       `EMPLOYEE`.`street` AS `street`,
       `EMPLOYEE`.`number` AS `number`,
       `EMPLOYEE`.`PC` AS `PC`,
       `EMPLOYEE`.`specialization` AS `specialization`,
       `EMPLOYEE`.`education_level` AS `education_level`,
       `EMPLOYEE`.`marital_status` AS `marital_status`,
       `EMPLOYEE`.`children_no` AS `children_no`,
       `EMPLOYEE`.`pid` AS `pid`,
       `EMPLOYEE`.`from_date` AS `from_date`
from `EMPLOYEE`
where (`EMPLOYEE`.`from_date` between '2012-01-01' and '2012-12-31');
```

Μη Ενημερώσιμη Όψη

Κατασκευάσαμε επιπλέον μία μη ενημερώσιμη όψη που παρουσιάζει τον μέσο μισθό ανά τμήμα.

```
CREATE VIEW `wage_stats` AS
select `p`.`department` AS `department`,
       avg(`s`.`wage`) AS `avg(wage)`
from (`SALARY` `s` join `POSITION` `p`)
where (`s`.`sal_id` = `p`.`sal_id`)
group by `p`.`department`;
```

SQL ΕΡΩΤΗΜΑΤΑ

Στο σύστημα μας έχουμε χρησιμοποιήσει παραμετρικά ερωτήματα που καλούμε μέσω της php για να πραγματοποιούμε insert, delete και update λειτουργίες στη βάση σε όποιον πίνακα επιθυμεί ο χρήστης.

Ακολουθεί ένα ενδεικτικό παράδειγμα για την εισαγωγή στοιχείου στα Συγκεντρωτικά Οικονομικά Στοιχεία :

```
<? include "../../database.php";
   include "../../models/AGGREGATED_FINANCIAL_DATA/functions.php";
$array= $_GET['array'];
$ssn = $_POST[ 'ssn'];
$year = $_POST[ 'year'];
$gross_year_earnings = $_POST['gross_year_earnings'];
$total_year_holdings = $_POST['total_year_holdings'];
$test = valid($ssn , $year, $gross_year_earnings,
$total_year_holdings);
if ($test){
    $suc =  insert(intval($ssn) , intval($year),
floatval($gross_year_earnings), floatval($total_year_holdings));
    if ($suc){
                include "../../views/inserted.php";
        }
        else {
                include "../../views/error_db.php";
        }
}
else{
        include "../../views/error.php";
}
?>
```

όπου η συνάρτηση insert ορίζεται στο αντίστοιχο αρχείο functions.php ως :

```
function insert($ssn,$year,$gross_year_earnings,$total_year_holdings){

        global $con ;
        $sql = "INSERT INTO AGGREGATED_FINANCIAL_DATA (SSN, year,
gross_year_earnings, total_year_holdings) VALUES
            ($ssn,$year,$gross_year_earnings,$total_year_holdings)";
        $suc = mysqli_query($con,$sql);
        mysqli_close($con);
        return $suc;
    }
```

Επιπλέον έχουμε χρησιμοποιήσει παραμετρικά ερωτήματα ώστε να μπορεί ο χρήστης να ταξινομεί όποιον πίνακα επιθυμεί με βάση όποιο πεδίο επιθυμεί είτε σε αύξουσα είτε σε φθίνουσα σειρά.

Τέλος δίνουμε τη δυνατότητα στο χρήστη να επιλέξει ένα από τα παρακάτω ερωτήματα:

- Ο υπάλληλος που εργάζεται στην εταιρία με το μεγαλύτερο σκορ

```
select surname, name, specialization, sum(score) as max_score
from EMPLOYEE as e1, EVALUATION as ev1
where e1.SSN = ev1.SSN and ISNULL(e1.to_date)
group by e1.SSN
having sum(score) >= all (select sum(score)
              from EVALUATION as ev2
              group by ev2.SSN)
```

- Υπάλληλοι που εργάζονται στην εταιρία με μισθό πάνω από το μέσο μισθό του τμήματός τους

```
select surname, name, from_date
from EMPLOYEE as e1, POSITION as p1, SALARY as s1
where e1.pid = p1.pid and p1.sal_id = s1.sal_id and
ISNULL(e1.to_date) and s1.wage*s1.work_days >
            (select avg(s.wage*s.work_days)
             from EMPLOYEE as e2, POSITION as p, SALARY as s
             where e2.pid = p.pid and p.sal_id = s.sal_id and
p1.department = p.department
                 group by p.department)
```

- Συνολικές μεικτές αποδοχές υπαλλήλων ανά έτος

```
select year, sum(gross_year_earnings) as total_amount_spent
from AGGREGATED_FINANCIAL_DATA as afd
group by year
```

- Άδειες θέσεις ανά τμήμα

```
select department, count(is_empty) as empty_spots_no
from POSITION as p
where p.is_empty > 0
group by p.department
```

- Καθαρές αποδοχές κάθε υπαλλήλου που εργάζεται στη εταιρεία

```
select SSN, surname, name, (s.wage*s.work_days - s.tax - s.stamp -
b.health_tax - b.pension_tax) as net_salary
from EMPLOYEE as e, POSITION as p, SALARY as s, BENEFITS as b
where e.pid = p.pid and p.sal_id = s.sal_id and p.ben_id = b.ben_id
and ISNULL(e.to_date)
```

- Συνολικός αριθμός των μέσων που έχουν αγγελία για κάθε θέση

```
select  pid, count(media_id)
from    RECRUITMENT_AD as r
group by pid
```

- Υπάλληλοι με μισθό κάτω από 2000 και σκορ πάνω από 300

```
select surname, name, specialization, education_level
from EMPLOYEE as e, EVALUATION as ev, POSITION as p, SALARY as s
where e.pid = p.pid and e.SSN = ev.SSN and p.sal_id = s.sal_id and
wage*work_days < 2500 and ISNULL(e.to_date)
group by e.SSN
having sum(score) >= 200
```

- Πολύτεκνοι υπάλληλοι με μισθό κάτω από 2000

```
select surname, name, children_no , wage*work_days as salary
from EMPLOYEE as e, POSITION as p, SALARY as s
where e.pid = p.pid and s.sal_id = p.sal_id and e.children_no >= 3
and wage*work_days < 2000
```

- Υπάλληλοι που προσελήφθησαν από το 2012 και μετά και έχουν ήδη συγκεντρώσει πάνω από 200 βαθμούς

```
select surname, name, from_date, sum(ev.score) as total_score
from EMPLOYEE as e, EVALUATION as ev
where e.SSN = ev.SSN
group by e.SSN
having sum(ev.score) >= 200
        and exists (select from_date
            from EMPLOYEE as e2
            where e.SSN = e2.SSN and from_date >= '2012-01-01')
```