ΑΝΩΤΑΤΟ ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΠΑΤΡΑΣ

ΣΧΟΛΗ ΔΙΟΙΚΗΣΗΣ ΚΑΙ ΟΙΚΟΝΟΜΙΑΣ

ΤΜΗΜΑ ΛΟΓΙΣΤΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

« ΣΧΕΔΙΑΣΗ ΚΑΙ ΥΛΟΠΟΙΗΣΗ ΤΥΠΟΠΟΙΗΜΕΝΗΣ ΔΙΑΔΙΚΑΣΙΑΣ ΜΙΑΣ ΕΤΑΙΡΕΙΑΣ »

ΣΠΟΥΔΑΣΤΡΙΕΣ

ΑΛΕΞΑΝΔΡΟΠΟΥΛΟΥ ΔΗΜΗΤΡΑ

ΒΑΣΙΛΑΚΟΠΟΥΛΟΥ ΑΓΓΕΛΙΚΗ

ΚΟΡΔΑ ΑΓΓΕΛΙΚΗ

ΕΙΣΗΓΗΤΗΣ

ΜΟΥΝΤΖΟΥΡΗΣ ΙΩΑΝΝΗΣ

ΠΑΤΡΑ 2006

ΕΥΧΑΡΙΣΤΙΕΣ

Για την διεκπεραίωση της Πτυχιακής Εργασίας μας, με το θέμα :Σχεδίαση και Υλοποίηση Τυποποιημένης Διαδικασίας μιας Εταιρείας , ευχαριστούμε θερμά τον καθηγητή μας κύριο Ιωάννη Μουντζούρη για την πολύτιμη βοήθειά του, που κατέβαλε στην ολοκλήρωση της εργασίας.

Κατά την διάρκεια της Πτυχιακής Εργασίας, είχαμε την ευκαιρία να εμπλουτίσουμε τις γνώσεις μας, πάνω σε ένα θέμα το οποίο μας ήταν εντελώς άγνωστο και καταφέραμε να δημιουργήσουμε κάτι μοναδικό και επικοδομητικό.

ΑΝΑΛΥΤΙΚΟΣ ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

1 ΕΙΣΑΓΩΓΗ

1.1 ΑΝΤΙΚΕΙΜΕΝΟ ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑΣ

Η συγκεκριμένη πτυχιακή εργασία έχει ως σκοπό τη δημιουργία ενός εύχρηστου πληροφοριακού συστήματος που θα υλοποιεί αποτελεσματικά την διαδικασία αίτησης, έγκρισης, και αποπληρωμής ταξιδίων μιας εταιρείας. Η οργάνωση μιας τέτοιας διαδικασίας, για την εταιρία συμπεριλαμβάνει την Αίτησης Ταξιδίου, την έγκριση της από κάποιο ανώτερο υπεύθυνο στέλεχος και την ενημέρωση του λογιστηρίου της εταιρίας. Για την ομαλή επικοινωνία των παραπάνω τμημάτων μεταξύ τους χρειάζεται ένα εύχρηστο και αυτοματοποιημένο πληροφορικό σύστημα.

Στην εργασία που ακολουθεί υλοποιείται η δημιουργία μιας λειτουργικής βάσης δεδομένων που να ενημερώνεται μέσω εύχρηστων φορμών, η οποία θα μπορεί να συγκρατεί όλες τις πληροφορίες για τις κινήσεις των στελεχών αλλά και των τμημάτων της εταιρίας και θα μπορεί να ανταποκρίνεται δίνοντας στον εκάστοτε χρήστη τις πληροφορίες που χρειάζεται.

Για τη δημιουργία του πακέτου αυτού των βάσεων δεδομένων, των πινάκων, των αιτημάτων και των φορμών επικοινωνίας των χρηστών με τις βάσεις χρησιμοποιούνται οι δυνατότητες του προγράμματος DELPHI 7.0 Enterprise σε συνδυασμό με την βάση δεδομένων Microsoft Access.

Το συγκεκριμένο αντικείμενο της εργασίας, είναι η σχεδίαση και δημιουργία ενός συστήματος βάσεων δεδομένων βασισμένο στο λογισμικό πρόγραμμα DELPHI 7.0 σε συνδυασμό με την Microsoft Access για την υλοποίηση ενός λειτουργικού για την εταιρία πληροφοριακού συστήματος που θα εξυπηρετεί την «Αίτησης Ταξιδίου της Εταιρείας».

Οπότε ο εξειδικευμένος στόχος της εργασίας είναι η χρήση των βάσεων δεδομένων για την καλύτερη εξυπηρέτηση των υπαλλήλων της εταιρείας, στην συμπλήρωση των αιτήσεων ταξιδίου.

Στην περίπτωση της συγκεκριμένης εργασίας η λογική είναι, να δημιουργήσουμε το γραφικό περιβάλλον το οποίο χρησιμοποιεί ο χρήστης στην γλώσσα προγραμματισμού DELPHI 7.0 σε συνδυασμό με την βάση δεδομένων MICROSOFT ACCESS στην οποία δημιουργήσαμε κάποιους πίνακες, ερωτήματα κ.τ.λ. τα οποία συνδέσαμε μέσω του ODBC με το DELPHI έτσι ώστε να υλοποιήσουμε την σχεδίαση των φορμών και να εκτελεί ορισμένες εργασίες όπως αίτηση, έγκριση, και αποπληρωμή ταξιδίων μιας εταιρείας.

Για να φτάσουμε σε αυτό το επίπεδο πρέπει να χωρίσουμε τις εργασίες μας σε τμήματα – φάσεις.

Στο ***πρώτο κεφάλαιο*** αναλύουμε τον σκοπό και το αντικείμενο της πτυχιακής εργασίας. Κάνουμε μια αναφορά στο σενάριο εφαρμογής για την καλύτερη κατανόηση της υλοποίησης του προγράμματος.

Στο ***δεύτερο κεφάλαιο*** γίνεται στην αρχή μια αναλυτική περιγραφή στις Βάσεις Δεδομένων, αναφέροντας λίγα λόγια για την Ιστορική Αναδρομή των Βάσεων Δεδομένων. Στην συνέχεια αναλύουμε τις Σχεσιακές Βάσεις Δεδομένων και τα Συστήματα Διαχείρισης. Περιγράφουμε την αρχιτεκτονική της Microsoft Access δηλαδή από τι αποτελείται (πίνακες, ερωτήματα, φόρμες, Δεδομένα, Πληροφορίες, Εκθέσεις, Ερωτήματα, Εγγραφές) καθώς και στις σχέσεις μεταξύ των πινάκων. Αναφέρουμε ποιά βήματα πρέπει να ακολουθήσουμε για την κατασκευή μιας εφαρμογής βάσεων δεδομένων. Το κεφάλαιο κλείνει παραθέτοντας τα δικά μας παραδείγματα που έχουμε δημιουργήσει στην Microsoft Access για την σχεδίαση και την υλοποίηση του προγράμματος.

Στο ***τρίτο κεφάλαιο*** αναλύουμε το Delphi, αναφερόμαστε στις εκδόσεις του αλλά και στο περιβάλλον προγραμματισμού του. Ακολουθεί, η δημιουργία μιας απλής εφαρμογής που ο ρόλος της είναι να γνωρίσουμε το πρόγραμμα και τις ιδιαιτερότητές του.

Στο ***τέταρτο κεφάλαιο*** υλοποιείται η πτυχιακή εργασία. Αναφέρουμε την διαδικασία δημιουργίας ενός ψευδωνύμου ODBC που είναι ο συνδετικός κρίκος ανάμεσα στους πίνακες που έχουμε δημιουργήσει στην Microsoft Access και πως υλοποιείται η σχεδίαση φορμών στο Delphi. Στην συνέχεια παραθέτουμε τα αντικείμενα του της παλέτας του Delphi που έχουμε χρησιμοποιήσει στην εργασία μας. . Έπειτα προσπαθούμε να υλοποιήσουμε την εφαρμογή δημιουργώντας της φόρμες όπου χρειάζονται καθώς επίσης και τον κώδικα προγραμματισμού των φορμών για να μας δώσουν το τελικό πρόγραμμα στο σύνολο του.

Στο ***πέμπτο κεφάλαιο*** κλείνουμε την εργασία μας, με τον επίλογο.

Στη συνέχεια ακολουθεί η βιβλιογραφία που έχουμε χρησιμοποιήσει για την ολοκλήρωση της εργασίας καθώς και οι κώδικες. προγραμματισμού των φορμών που έχουν δημιουργηθεί σε γλώσσα προγραμματισμού του DELPHI 7.0.

1.2 ΣΕΝΑΡΙΟ ΕΦΑΡΜΟΓΗΣ

Παρακάτω αναφέρουμε λίγα λόγια για το σενάριο της εργασίας. Συγκεκριμένα, αναλύουμε πως υλοποιείται η διαδικασία διεκπεραίωσης της εφαρμογής.
Η εταιρεία μας είναι μια υποθετική Ανώνυμη Εταιρεία και η έδρα της βρίσκεται στην Πάτρα. Αποτελείται από τα εξής τμήματα:

- Διευθυντής
- Λογιστήριο
- Υπάλληλοι
- Τεχνικό Προσωπικό – Διαχειριστής

Στα πλαίσια της εταιρείας οι υπάλληλοι κάθε τμήματος θα πρέπει να πραγματοποιούν κατά καιρούς διάφορα ταξίδια για χάρη της εταιρείας (π.χ. σεμινάρια λογιστών, σεμινάρια τεχνικών πωλήσεων, επισκέψεις σε υποκαταστήματα, κ.τ.λ.).

Οι δαπάνες των ταξιδιών καλύπτονται από την εταιρεία εφόσον κάθε υπάλληλος που πραγματοποιήσει κάποιο ταξίδι, προσκομίσει στο λογιστήριο τα απαραίτητα δικαιολογητικά (εισιτήρια, αποδείξεις, τιμολόγια κ.τ.λ.).

Η διαδικασία που θα πρέπει να ακολουθήσει ένας υπάλληλος για να πραγματοποιήσει ένα ταξίδι είναι η εξής :

Ο υπάλληλος κάνει εισαγωγή στο σύστημα, δίνει όνομα και κωδικό χρήστη και μπαίνει στην κύρια φόρμα (MainForm) . Στην οθόνη εμφανίζονται διάφορες φόρμες ,αλλά ο υπάλληλος δεν έχει πρόσβαση σε όλες. Επιλέγει τη φόρμα " ΑΙΤΗΣΗ ΤΑΞΙΔΙΟΥ " και συμπληρώνει όλα τα απαραίτητα πεδία. Αφού κρατήσει ένα αντίγραφο, η αίτηση παραμένει εκκρεμής μέχρι να ελεγχθεί από τον υπάλληλο του Λογιστηρίου.

Όταν ο υπάλληλος του λογιστηρίου κάνει εισαγωγή στο σύστημα, δίνοντας όνομα και κωδικό χρήστη, θα ενημερωθεί για τυχόν εκκρεμής αιτήσεις. Αφού τις ελέγξει, κρατά αντίγραφο και τις διεκπεραιώνει στον Διευθυντή.

Ο Διευθυντής, με την σειρά του, κάνοντας εισαγωγή στο σύστημα, δίνει όνομα και κωδικό χρήστη και μπαίνει στην κύρια φόρμα (MainForm). ελέγχει κατά τον ίδιο τρόπο, αν υπάρχουν εκκρεμής αιτήσεις και τις εγκρίνει.

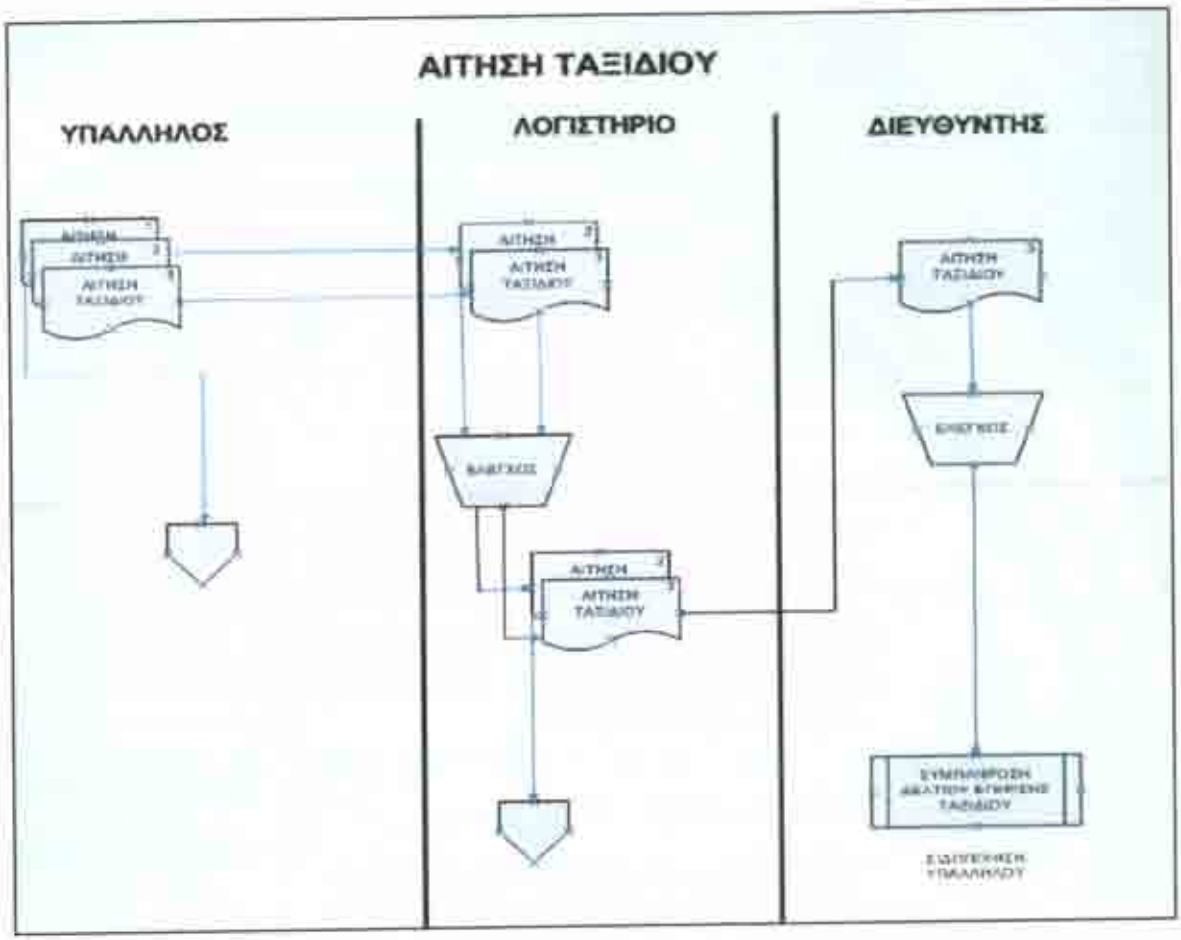
Επιλέγει την Φόρμα " ΔΕΛΤΙΟ ΕΓΚΡΙΣΗΣ ΤΑΞΙΔΙΟΥ ", συμπληρώνει τα απαραίτητα πεδία, το εκτυπώνει, το υπογράφει, κρατά αντίγραφο και το διεκπεραιώνει στο Λογιστήριο, το οποίο ενημερώνει τον υπάλληλο ότι μπορεί να πραγματοποιήσει το ταξίδι.

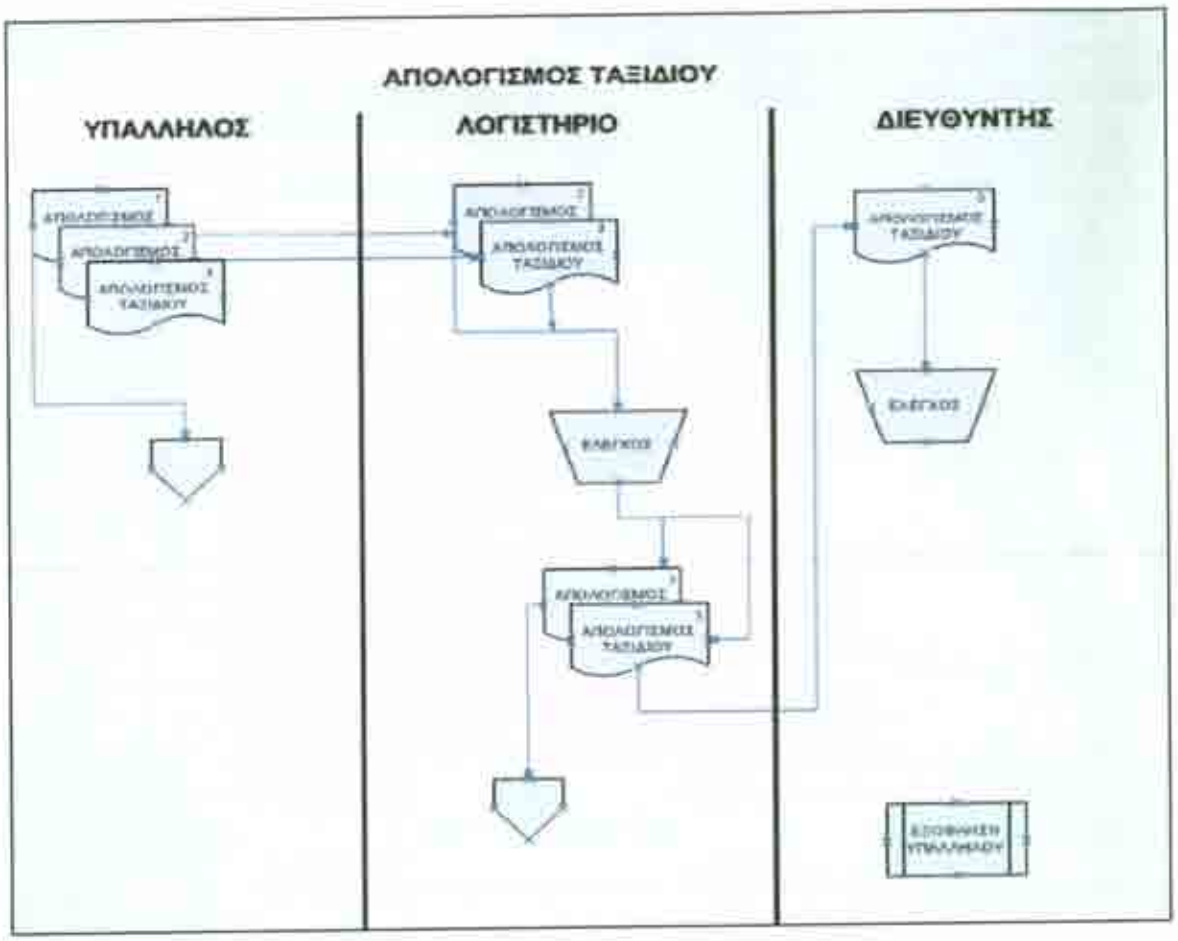
Όταν ο υπάλληλος, επιστρέψει από το ταξίδι, κάνει εισαγωγή στο σύστημα, δίνει όνομα και κωδικό χρήστη και εισάγεται στην κύρια φόρμα.

Επιλέγει την φόρμα " ΑΠΟΛΟΓΙΣΜΟΣ ΤΑΞΙΔΙΟΥ ΕΣΩΤΕΡΙΚΟΥ", συμπληρώνει τα απαραίτητα πεδία και προσκομίζει στο Λογιστήριο τα απαραίτητα δικαιολογητικά.

Αυτό με την σειρά του κάνει το έλεγχο και διεκπεραιώνει τον Απολογισμό στον Διευθυντή. Αφού ο Διευθυντής, τον ελέγξει και τον υπογράψει ενημερώνει το Λογιστήριο ώστε να μπορέσει ο υπάλληλος να εισπράξει το ποσό που δαπάνησε για χάρη της εταιρείας.

Ακολουθούν τα Διαγράμματα Ροής Αίτησης Ταξιδίου και Απολογισμός Ταξιδίου Εσωτερικού




ΑΠΟΛΟΓΙΣΜΟΣ ΤΑΞΙΔΙΟΥ
ΥΠΑΛΛΗΛΟΣ
ΛΟΓΙΣΤΗΡΙΟ
ΔΙΕΥΘΥΝΤΗΣ
ΑΠΟΛΟΓΙΣΜΟΣ
ΑΠΟΛΟΓΙΣΜΟΣ ΤΑΞΙΔΙΟΥ
ΕΛΕΓΧΟΣ

2 ΣΧΕΔΙΑΣΗ ΚΑΙ ΥΛΟΠΟΙΗΣΗΣ ΤΗΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ ΤΗΣ ΕΦΑΡΜΟΓΗΣ ΜΕ ΤΗ ΧΡΗΣΗ ΤΗΣ MICROSOFT ACCESS

2.1 ΑΝΤΙΚΕΙΜΕΝΟ ΤΟΥ ΚΕΦΑΛΑΙΟΥ

Σκοπός αυτού του κεφαλαίου είναι να κατανοήσουμε τι είναι οι σχεσιακές *βάσεις δεδομένων*, πως λειτουργούν, και σε ποιες περιπτώσεις τις χρησιμοποιούμε. Επίσης, θα εξοικειωθούμε με ορισμένους από τους βασικούς όρους που χρησιμοποιούνται στις βάσεις δεδομένων και θα γνωρίσουμε τα κύρια συστατικά στοιχεία τους. Θα μάθουμε τι αποκαλούμε δεδομένα και τι πληροφορίες, καθώς και ποιος είναι ο ρόλος των πινάκων, των φορμών, των ερωτημάτων, και των εκθέσεων.

Μια βάση δεδομένων την χρησιμοποιούμε για την *συλλογή*, την *αποθήκευση*, την *οργάνωση*, την *επεξεργασία*, και την *εξαγωγή δεδομένων* και *πληροφοριών*.

Σε έναν υπολογιστή χρησιμοποιούμε μηχανογραφημένες βάσεις δεδομένων, οι οποίες είναι ισχυρές και ευέλικτες. Μας επιτρέπει να παρουσιάζουμε τα δεδομένα τους με διάφορους τρόπους, να τα ταξινομούμε, να τα φιλτράρουμε και να χρησιμοποιούμε τα κατάλληλα κριτήρια ώστε αν παίρνουμε ακριβώς τις πληροφορίες που επιθυμούμε.

2.2 ΙΣΤΟΡΙΚΗ ΑΝΑΔΡΟΜΗ ΣΤΙΣ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

Βάση δεδομένων είναι η οργάνωση και καταχώρηση της πληροφορίας σε τρόπο ώστε να ενημερώνεται και να ανακαλείται με τον πλέον ευέλικτο τρόπο.

Αρχίζοντας από τα τέλη του 18ου αιώνα, η μηχανική, αυτόματη επεξεργασία της πληροφορίας έχει περάσει από διάφορους σταθμούς για να γνωρίσει ραγδαία ανάπτυξη από την στιγμή που η κωδικοποίηση άρχισε να γίνετε ηλεκτρονικά.

Ανάπτυξη σχεδόν επαναστατική, που επηρεάζει όλο και περισσότερο το σύνολο της ανθρώπινης δραστηριότητας. Ο τρόπος με τον οποίο ο σύγχρονος άνθρωπος εργάζεται, εκπαιδεύεται, ψυχαγωγείται, ακόμα και σε ένα μεγάλο βαθμό αυτή καθαυτή η κοινωνική του

συμπεριφορά προσαρμόζονται διαρκώς στις ραγδαίες αλλαγές που επιφέρει η σύγχρονη κοινωνία της πληροφόρησης, όπως ονομάζεται.

Στο κείμενο που ακολουθεί επιχειρείται μια μακροσκοπική θεώρηση και χρονολογική κατάταξη των κυριοτέρων σταδίων από τα οποία περάσαμε για να φτάσουμε εκεί που είμαστε σήμερα, στα Συστήματα Διαχείρισης Βάσεων Δεδομένων (Database Management System : DBMSes

1880 Εφεύρεση της διάτρητης κάρτας. Γίνεται το κατεξοχήν μέσο αποθήκευσης της πληροφορίας .

1940 Ο πρώτος ηλεκτρονικός υπολογιστής στα 1946. Χρησιμοποιείται για μαθηματικούς υπολογισμούς. Δεν υπάρχει ακόμα μεγάλη ανάγκη μόνιμης αποθήκευσης της πληροφορίας. Εφευρίσκεται η μαγνητική ταινία. Γρήγορη και ευέλικτη, πάντως σειριακή καταχώρηση της πληροφορίας.

1950 Αρχίζουν να εμφανίζονται τα πρώτα πληροφοριακά συστήματα, κυρίως μισθοδοσίες. Συχνά σχεδιάζονται και λειτουργούν σε απομόνωση το ένα από το άλλο. Δεν γίνεται μεταφορά δεδομένων από υπολογιστή σε υπολογιστή ούτε καν από εφαρμογή σε εφαρμογή. Το τυπικό περιβάλλον αποτελείται από πολλά μικρά προγράμματα και πολλά αρχεία. Το καθένα από τα αρχεία περιέχει τμηματική πληροφορία όσον αφορά το σύνολο της επιχείρησης. Δύσκολη η δημιουργία ολοκληρωμένου περιβάλλοντος με τα δεδομένα που χρησιμοποιούνται από πολλές εφαρμογές.

1960 Εφευρίσκεται ο μαγνητικός δίσκος. Ταχύτερη, πλέον, η πρόσβαση στα στοιχεία. Η προσπέλαση στα δεδομένα εκτός από σειριακή γίνεται και άμεση. Εμφανίζονται συστήματα όπου πολλά αρχεία χρησιμοποιούνται από πολλές εφαρμογές ταυτόχρονα. Προβλήματα συντονισμού και συγχρονισμού της στα κοινά αρχεία. Ακόμη, προβλήματα έλλειψης αυτόνομης λογικής συσχέτισης των δεδομένων από αρχείο σε αρχείο.

1970 Γίνεται πλέον συνείδηση ότι: 1)Χρειάζεται το ολοκληρωμένο πληροφοριακό περιβάλλον (Βάση Δεδομένων) 2)Με τον όρο ολοκληρωμένο νοείται η υποστήριξη του συνόλου των πληροφοριακών αναγκών ενός οργανισμού από μία και μόνη Βάση Δεδομένων. 3)Το σχήμα δημιουργεί πρόσθετες απαιτήσεις όσον αφορά τον σχεδιασμό και τη λειτουργία του λογισμικού : Η συγχρονισμένη συνλειτουργία των διαφόρων υποσυστημάτων διαχειρίζεται τα ίδια αξιόπιστα δεδομένα όπου η κάθε στιγμή πληροφορίας καταχωρείται μία και μόνη φορά.

2.3 ΣΥΣΤΗΜΑΤΑ ΔΙΑΧΕΙΡΙΣΗΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ

Οι βάσεις δεδομένων χρησιμεύουν στη συλλογή, την αποθήκευση, την οργάνωση, την επεξεργασία, και την εξαγωγή δεδομένων και πληροφοριών. Μπορούμε να πούμε ότι μια βάση δεδομένων είναι μια συλλογή από εγγραφές και αρχεία τα όποια είναι οργανωμένα με τέτοιο τρόπο να εξυπηρετούν τέτοιο σκοπό. Το λογισμικό που χρησιμοποιείται για την οργάνωση των περιεχομένων ονομάζεται *Σύστημα Διαχείρισης Βάσης Δεδομένων* (Database Management System – DBMS).

Τα συστήματα DBMS απαιτούν μεγάλη υπολογιστική ισχύ γιατί αρχικά χρησιμοποιούνταν μόνο σε μεγάλα υπολογιστικά συστήματα (Mainframes).Σήμερα όμως λόγω της αύξησης των δυνατοτήτων των προσωπικών υπολογιστών εκτελούνται και λειτουργούν ικανοποιητικά ακόμα και σε μικρούς υπολογιστές.

2.4 ΣΧΕΣΙΑΚΕΣ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

Σήμερα, τα περισσότερα σύγχρονα συστήματα διαχείρισης βάσεων δεδομένων χρησιμοποιούν το σχεσιακό μοντέλο, γιατί και ονομάζονται *Συστήματα Διαχείρισης Βάσεων Δεδομένων* (Relational Database Management Systems – RDMS).

Οι σχεσιακές Βάσεις Δεδομένων μας επιτρέπουν, να καταχωρίζουμε μοναδικά στοιχεία σε κάθε εγγραφή και παράλληλα να συσχετίζουμε τους πίνακες της βάσης δεδομένων μεταξύ τους ώστε να μην χρειάζεται να καταχωρίζουμε τα ίδια στοιχεία σε διαφορετικούς πίνακες.

2.5 ΟΡΟΛΟΓΙΑ

Οι όροι και οι έννοιες των Βάσεων Δεδομένων που θα συναντήσουμε στην συνέχεια είναι οι εξής :

- Δεδομένα
- Πληροφορίες
- Βάση Δεδομένων
- Εγγραφή
- Πεδίο
- Πίνακας
- Φόρμα
- Έκθεση
- Ερώτημα

2.5.1 Δεδομένα

Δεδομένα είναι οι στατικές τιμές που καταχωρίζουμε σε μια βάση δεδομένων. Οι συνηθέστεροι τύποι δεδομένων που μπορούμε να καταχωρίσουμε στην Microsoft Access είναι οι εξής :

- Κείμενο (Αλφαβητικούς Χαρακτήρες)
- Υπόμνημα (Αλφαβητικούς Χαρακτήρες και Αριθμητικές Τιμές)
- Ημερομηνία Ώρα (Ημερομηνία / Ώρα)
- Νομισματική Μονάδα (Αριθμητικές Τιμές)
- Αυτόματη Αρίθμηση (Αριθμητικές Τιμές)
- Ναι / Όχι (Γραφικά, ήχος δεδομένα που υποστηρίζουν τα Windows)
- Αντικείμενο Ole (Εικόνες γραφήματα ή άλλα αντικείμενα από άλλες εφαρμογές των Windows)
- Λίστα Αναζήτησης (Εμφανίζει δεδομένα από άλλον πίνακα)
- Δεσμός (Σύνδεσμος σε πηγή Internet)

Εικόνα 1.1 Παράδειγμα Τύπων Δεδομένων

2.5.2 Πληροφορίες

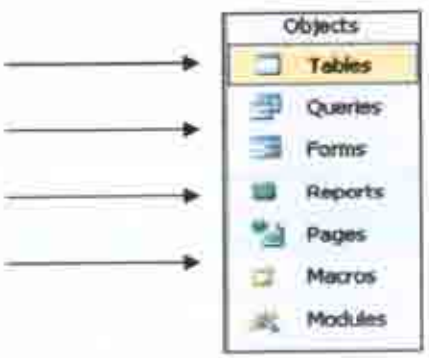
Πληροφορίες είναι τα δεδομένα που ανακτούμε από μια βάση δεδομένων, τα οποία είναι οργανωμένα με τέτοιο τρόπο ώστε να έχουν νόημα για το άτομα που τα εξετάζει. Δηλαδή, σε μία βάση δεδομένων αποθηκεύουμε δεδομένα και ανακτούμε πληροφορίες.

2.5.3 Βάση Δεδομένων

Βάση δεδομένων, είναι μια οργανωμένη συλλογή δεδομένων τα οποία συσχετίζονται μεταξύ τους. Ένα πρόγραμμα διαχείρισης βάσεων δεδομένων, όπως η Microsoft Access, μας επιτρέπει να καταχωρίζουμε, να οργανώνουμε, και να επεξεργαζόμαστε τα δεδομένα που μας ενδιαφέρουν. Τα δεδομένα και οι πληροφορίες που απορρέουν από αυτά έχουν κοινά χαρακτηριστικά, καθώς και συγκεκριμένο σκοπό η θέμα.

Τα βασικά στοιχεία μιας βάσης δεδομένων είναι τα παρακάτω :

- Πίνακες
- Ερωτήματα
- Φόρμες
- Εκθέσεις



2.5.4 Πίνακες

Όλα τα δεδομένα που καταχωρίζουμε στην Microsoft Access αποθηκεύονται σε έναν ή περισσότερους πίνακες. Ένας πίνακας είναι μια συλλογή δεδομένων που περιγράφουν ομοειδή αντικείμενα. Για παράδειγμα , ένας πίνακας μπορεί να περιέχει όλα τα στοιχεία των υπαλλήλων μιας εταιρείας, ένας άλλος περιέχει τα στοιχεία των πελατών της.

	ΚΩΔΙΚΟΣ ΥΠ	ΟΝΟΜΑ	ΕΠΩΝΥΜΟ	ΠΑΤΡΩΝΥΜΟ
▶	1	ΑΓΓΕΛΙΚΗ	ΚΟΡΔΑ	ΔΗΜΗΤΡΙΟΣ
	2	ΔΗΜΗΤΡΑ	ΑΛΕΞΑΝΔΡΟΠΟΥΛΟΥ	ΚΩΝΣΤΑΝΤΙΝΟΣ
	3	ΑΓΓΕΛΙΚΗ	ΒΑΣΙΛΑΚΟΠΟΥΛΟΥ	ΑΝΔΡΕΑΣ
	4	ΙΩΑΝΝΗΣ	ΜΟΥΝΤΖΟΥΡΗΣ	ΣΠΥΡΟΣ

Εικόνα 1.2 Παράδειγμα πίνακα

2.5.5 Εγγραφές

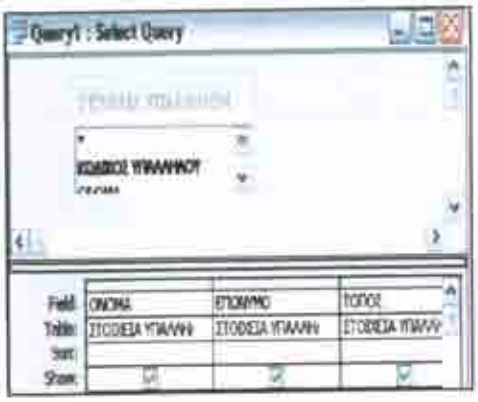
Κάθε γραμμή σε έναν πίνακα μίας βάσης δεδομένων ονομάζεται εγγραφή, και περιέχει όλα τα δεδομένα που περιγράφουν μια συγκεκριμένη καταχώριση του πίνακα. Για παράδειγμα, μια εγγραφή σε έναν πίνακα υπαλλήλων περιέχει όλα τα στοιχεία ενός υπαλλήλου, δηλαδή το όνομα επώνυμο κ.τ.λ.

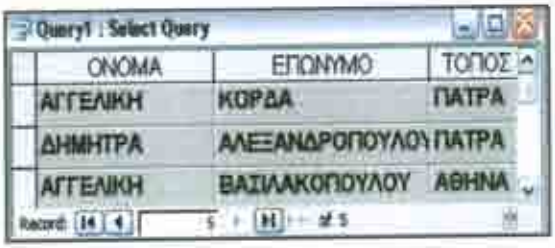
▶	1	ΑΓΓΕΛΙΚΗ	ΚΟΡΔΑ	ΔΗΜΗΤΡΙΟΣ	ΠΑΤΡΑ	ΕΡΜΟΥ 62

Εικόνα 1.3 Παράδειγμα εγγραφής

2.5.6 Ερωτήματα

Με την βοήθεια των ερωτημάτων μπορούμε, χρησιμοποιώντας τα κατάλληλα κριτήρια, να εντοπίζουμε και να παρουσιάζουμε τα δεδομένα που μας ενδιαφέρουν, από μια βάση δεδομένων. Για παράδειγμα, με ένα ερώτημα θα μπορούσαμε να εμφανίσουμε τα στοιχεία των υπαλλήλων, της εταιρείας που κατοικούν στην Πάτρα.





Εικόνα 1 4Ερώτημα σε προβολή σχεδίασης (αριστερά) και τα αποτελέσματα σε προβολή Φύλλου Δεδομένων (δεξιά)

2.5.7 Φόρμες

Η Microsoft Access μας παρέχει έναν εύκολο τρόπο για την καταχώριση και την επεξεργασία των δεδομένων της. Μια φόρμα είναι άμεσα συνδεμένη με έναν οι περισσότερους πίνακες και εμφανίζει συγκεντρωμένα τα στοιχεία που αφορούν κάθε εγγραφή. Οι φόρμες της Access είναι ένας όμορφος τρόπος απεικόνισης των περιεχομένων των Πινάκων (Tables) ή των Ερωτημάτων (Queries) μιας Βάσης Δεδομένων. Μια φόρμα αναφέρεται πάντα σ' έναν πίνακα (table) ή σ' ένα ερώτημα (query) της Access απ' όπου και παίρνει τα δεδομένα που απεικονίζει στην οθόνη. Σ' έναν πίνακα μπορούμε να αντιστοιχίσουμε όσες φόρμες θέλουμε, δηλ. διαφορετικούς τρόπους εμφάνισης των δεδομένων μας.

2.5.8 Εκθέσεις

Οι εκθέσεις αποτελούν έναν αποτελεσματικό τρόπο για την παρουσίαση και την εκτύπωση επιλεγμένων πληροφοριών της βάσης δεδομένων μας. Με τις εκθέσεις μπορούμε εύκολα να δημιουργούμε έναν τηλεφωνικό κατάλογο των υπαλλήλων, με τις εγγραφές τους ομαδοποιημένα ανά πόλη ή νομό και ταξινομημένες κατά επώνυμο και έπειτα κατά όνομα. Οι ***αναφορές*** *(reports)* της Access είναι ένας ωραίος τρόπος εκτύπωσης των δεδομένων που υπάρχουν σ' έναν πίνακα (table) ή σ' ένα ερώτημα (query) της Access. Τα δεδομένα ενός πίνακα μπορούμε να τα εκτυπώσουμε και με την αντίστοιχη εντολή (εικονίδιο) που υπάρχει στις επιλογές ενός πίνακα,

αλλά με τις αναφορές μπορούμε να κάνουμε πολλά περισσότερα πράγματα, με κυριότερο την ομαδοποίηση και την εμφάνιση αθροισμάτων (sum) για τα αριθμητικά πεδία.

2.6 ΑΝΑΛΥΤΙΚΗ ΠΕΡΙΓΡΑΦΗ ΠΙΝΑΚΩΝ

Για να δημιουργήσουμε πίνακες μιας βάσης δεδομένων, πατάμε στο κουμπί *Πίνακες* το οποίο βρίσκεται στο τμήμα *Αντικείμενα* του παραθύρου βάσης δεδομένων και χρησιμοποιούμε ένα από τα εξής τρία εικονίδια συντομεύσεων:

- Δημιουργία πίνακα σε προβολή σχεδίασης
- Δημιουργία πίνακα με την χρήση οδηγού
- Δημιουργία πίνακα με πληκτρολόγηση δεδομένων

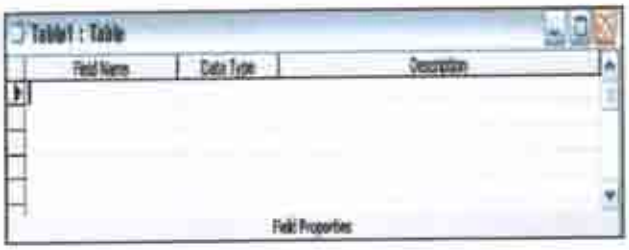
2.6.1 Δημιουργία πίνακα σε Προβολή Σχεδίασης

Για να δημιουργήσουμε ένα πίνακα σε προβολή σχεδίασης, κάνουμε τα παρακάτω :

- Πατάμε στο κουμπί *Πίνακες* στο τμήμα *Αντικείμενα* του παραθύρου Βάσης Δεδομένων.
- Πατάμε στο εικονίδιο συντόμευσης *Δημιουργία πίνακα σε προβολή σχεδίασης*.

Εικόνα 1.5 Το κλείσιμο διαλόγου Δημιουργία Πίνακα

- Εμφανίζεται ένας κενός πίνακας με το όνομα *Πίνακας 1* σε προβολή *Σχεδίασης*.



Εικόνα 1.6 Ο νέος πίνακας σε προβολή σχεδίασης

2.6.2 Προσθήκη πεδίου και ορισμός τύπου δεδομένων

Όπως βλέπουμε στην εικόνα στα αριστερά του πρώτου κελιού εμφανίζεται ένα δεξιό βέλος το οποίο δείχνει ότι το συγκεκριμένο κελί είναι το ενεργό. Επίσης, στο εσωτερικό του πρώτου κελιού εμφανίζεται το σημείο εισαγωγής.

- Στο πρώτο κελί της στήλης *όνομα πεδίου* πληκτρολογούμε το όνομα του πρώτου πεδίου του πίνακα.
- Πατάμε το πλήκτρο Tab.

Στη στήλη αυτή καθορίζουμε τον τύπο των δεδομένων κάθε πεδίου. Όταν μετακινούμαστε σε ένα κελί της στήλης *τύπος δεδομένων* αμέσως μετά την καταχώριση ενός νέου ονόματος πεδίου, το πρόγραμμα μας προτείνει τον τύπο δεδομένων *Κείμενο*. Ταυτόχρονα, στη δεξιά πλευρά του κελιού εμφανίζεται ένα βέλος που δείχνει προς τα κάτω. Αν πατήσουμε σε αυτό το βέλος, εμφανίζεται ένας πτυσσόμενος κατάλογος από τον οποίο μπορούμε να επιλέξουμε τον τύπο των δεδομένων που θέλουμε.

Εικόνα 1.7 Ο πτυσσόμενος κατάλογος των διαθέσιμων τύπων δεδομένων

2.6.3 Ορισμός πρωτεύοντος κλειδιού

Το Πρωτεύον Κλειδί είναι ένα πεδίο ενός πίνακα της Access, που χαρακτηρίζει μοναδικά μία εγγραφή μέσα σ' ολόκληρο τον πίνακα. Δηλ., δεν μπορεί να υπάρχουν δύο ή περισσότερες εγγραφές που να έχουν ίδια τιμή στο πρωτεύον κλειδί ενός πίνακα.

Ακόμη, οι εγγραφές του πίνακα ταξινομούνται αυτόματα με βάση το πρωτεύον κλειδί. Σ' έναν πίνακα, μπορούμε να ορίσουμε σαν πρωτεύον κλειδί και έναν συνδυασμό δύο ή περισσοτέρων πεδίων, όταν ένα πεδίο μόνο του δεν μπορεί να ορίσει μοναδικά μια εγγραφή.

Για να ορίσουμε ένα πεδίο ως πρωτεύον κλειδί, ακολουθούμε τα παρακάτω βήματα :

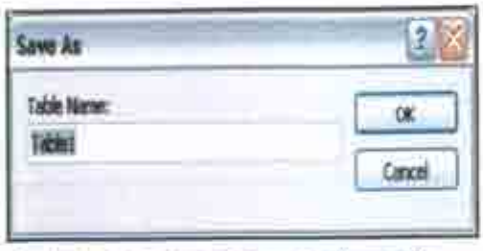
- Επιλέγουμε το πεδίο που θέλουμε να ορίσουμε ως πρωτεύον κλειδί
- Πατάμε το αριστερό πλήκτρο του ποντικιού και επιλέγουμε τη γραμμή με τα στοιχεία του πεδίου
- Πατάμε στο κουμπί *πρωτεύον κλειδί* της γραμμής εργαλείων *σχεδίαση πίνακα*
- Στα αριστερά του πεδίου εμφανίζεται ένα εικονίδιο κλειδιού, το οποίο μας ενημερώνει ότι το πεδίο έχει οριστεί ως πρωτεύον κλειδί

2.6.4 Αποθήκευση και κλείσιμο πίνακα

Για να αποθηκεύσουμε τον πίνακα κάνουμε τα εξής :

- Πατάμε στο κουμπί *αποθήκευση*

Στην οθόνη μας εμφανίζεται το πλαίσιο διαλόγου *αποθήκευση ως*



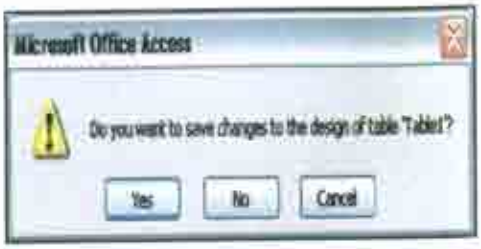
Εικόνα 1.8 Το πλαίσιο διαλόγου Αποθήκευση Ως

- Πληκτρολογούμε το όνομα του πίνακα και πατάμε *OK*

Για να κλείσουμε έναν πίνακα κάνουμε τα εξής :

- Πατάμε στο κουμπί *κλεισίματος* του παραθύρου του *πίνακα*
- Επιλέγουμε την εντολή *κλείσιμο* από το μενού *αρχείο*

Πατάμε στο *ναι* για να αποθηκεύσουμε τις αλλαγές



Εικόνα 1.9
Το πλαίσιο διαλόγου Κλείσιμο Πίνακα

2.6.5 Άνοιγμα πίνακα σε προβολή Σχεδίασης

Για να ανοίξουμε έναν πίνακα κάνουμε τα εξής :

- Επιλέγουμε το πίνακα στο παράθυρο βάσης δεδομένων
- Πατάμε στο κουμπί *σχεδίαση* της γραμμής εργαλείων
- Ο πίνακας εμφανίζεται σε προβολή σχεδίασης

2.6.6 Άνοιγμα πίνακα σε προβολή Φύλλου Δεδομένων

Για να ανοίξουμε έναν πίνακα κάνουμε τα εξής :

- Επιλέγουμε το πίνακα στο παράθυρο βάσης δεδομένων
- Πατάμε στο κουμπί *Άνοιγμα* της γραμμής εργαλείων
- Ο πίνακας εμφανίζεται σε προβολή φύλλου δεδομένων

Εικόνα 1.10 Παράδειγμα πίνακα σε προβολή φύλλου δεδομένων

2.6.7 Ιδιότητες πινάκων

Οι ιδιότητες των πεδίων μας επιτρέπουν να προσαρμόσουμε τον τρόπο με τον οποίο η Access αποθηκεύει, εμφανίζει, και χειρίζεται τα δεδομένα ενός πεδίου.

Στη επόμενη εικόνα βλέπουμε τις ιδιότητες των πεδίων

Εικόνα1.11 Ιδιότητες Πεδίων

- Μέγεθος πεδίου
- Μορφή
- Μάσκα Εισαγωγής
- Λεζάντα
- Προεπιλεγμένη Τιμή
- Κανόνας Επικύρωσης
- Κείμενο Επικύρωσης
- Απαιτείται
- Μηδενικό Μήκος
- Με Ευρετήριο
- Δεκαδικές Θέσεις
- Νέες Τιμές

2.7 ΣΧΕΣΙΑΚΕΣ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ ΜΕ ΤΗ ΧΡΗΣΗ ΤΗΣ MICROSOFT ACCESS

Με την δημιουργία των πινάκων της βάσης δεδομένων, πρέπει να τους συνδέσουμε μεταξύ τους ώστε να μπορούμε να δημιουργήσουμε ερωτήματα, φόρμες, και εκθέσεις που θα παίρνουν δεδομένα από περισσότερους από έναν πίνακες, ταυτόχρονα. Η σύνδεση μεταξύ των πινάκων αποκαλείται σχέση.

Τύποι σχέσεων

Η δημιουργία μεταξύ των πινάκων προϋποθέτει τη σύνδεση δύο πεδίων, ενός από κάθε πίνακα. Συνήθως , η σχέση δημιουργείται μεταξύ ενός πεδίου του πρώτου πίνακα, το οποίο περιέχει μοναδικές τιμές και αποκαλείται πρωτεύον κλειδί, και ενός πεδίου του δευτέρου πίνακα το οποίο έχει συνήθως το ίδιο όνομα και αποκαλείται συνήθως *εξωτερικό κλειδί* η *ξένο κλειδί*.

Το πρόγραμμα Microsoft Access μας επιτρέπει να δημιουργήσουμε τους τρείς παρακάτω διαφορετικούς τύπους σχέσεων :

- Σχέση *ένα προς ένα* (ή αμφιμονοσήμαντη – one to one).
- Σχέση *ένα προς πολλά* (μονοσήμαντη – one to many).
- Σχέση *πολλά προς πολλά* (πολυσήμαντη – many to many).

Σχέση ένα προς ένα

Στον τύπο σχέσης έναν προς ένα κάθε εγγραφή του πρώτου πίνακα πρέπει να αντιστοιχεί σε μια εγγραφή του δευτέρου πίνακα, και αντίστροφα. Για να είναι δυνατή η δημιουργία μιας τέτοιας σχέσης πρέπει να είναι πρωτεύοντα κλειδιά και στους δύο πίνακες ή να έχουν μοναδικές τιμές, δηλαδή η ιδιότητα ΕΥΡΕΤΗΡΙΟ των πεδίων να είναι αληθής, χωρίς να επιτρέπονται διπλότυπα.

Σχέση ένα προς πολλά

Στον τύπο σχέσης έναν προς πολλά κάθε εγγραφή του πρώτου πίνακα πρέπει να αντιστοιχεί σε πολλές εγγραφές του δευτέρου πίνακα. Αντίθετα, μια εγγραφή του δευτέρου πίνακα, μπορεί να αντιστοιχεί σε μια εγγραφή του πρώτου. Για να είναι δυνατή η δημιουργία μιας τέτοιας σχέσης πρέπει μόνα το ένα να είναι πρωτεύον κλειδί η ή να έχει μοναδικές τιμές.

Σχέση πολλά προς πολλά

Στον τύπο σχέσης πολλά προς πολλά, κάθε εγγραφή του πρώτου πίνακα μπορεί να αντιστοιχεί σε πολλές εγγραφές του δευτέρου πίνακα, και αντίστροφα. Για να είναι δυνατή η δημιουργία μιας τέτοιας σχέσης πρέπει να δημιουργηθεί ένας τρίτος πίνακας, ο οποίος αποκαλείται *πίνακας σύνδεσης*. Το πρωτεύον κλειδί του πίνακα σύνδεσης πρέπει να αποτελείται από το συνδυασμό δύο πεδίων, τα οποία είναι τα πρωτεύοντα κλειδιά των πρώτων πινάκων.

Δημιουργία Σχέσης

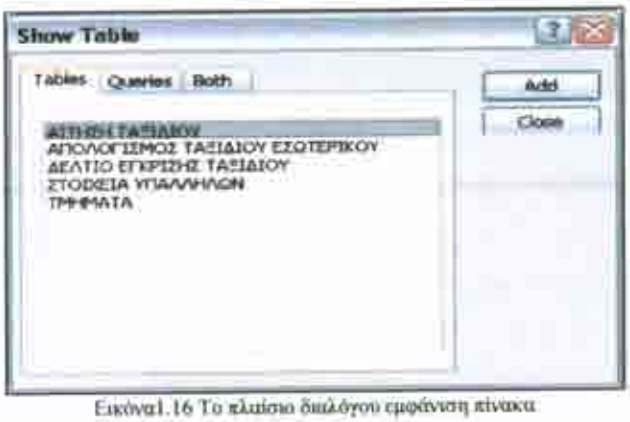
Για να δημιουργήσουμε μια σχέση κάνουμε τα εξής :

- Βεβαιωνόμαστε ότι όλοι οι πίνακες της βάσης δεδομένων είναι κλειστοί.
- Πατάμε το κουμπί *Σχέσεις* της γραμμής εργαλείων
- Στην οθόνη μας εμφανίζεται το παράθυρο Σχέσεις όπου υπάρχουν οι λίστες των πεδίων των πινάκων για τους οποίους έχει δημιουργηθεί η σχέση

Για να εμφανίσουμε τους υπόλοιπους πίνακες, ώστε να προσθέσουμε αυτούς που θέλουμε στο παράθυρο σχέσεις κάνουμε τα εξής :

- Πατάμε στο κουμπί εμφάνιση πίνακα της γραμμής εργαλείων Σχέση
- Στην οθόνη ανοίγει το πλαίσιο διαλόγου *Εμφάνιση Πίνακα*

- Με τις τρείς καρτέλες αυτού του πλαισίου διαλόγου (πίνακες, ερωτήματα, και τα δύο) μπορούμε να προσπελάσουμε τα αντίστοιχα αντικείμενα της βάσης δεδομένων ώστε να προσθέσουμε κάποιο από αυτά στο παράθυρο σχέσεις.
- Επιλέγουμε το πίνακα που θέλουμε και πατάμε στο κουμπί *Προσθήκη*
- Πατάμε το κουμπί κλείσιμο του παραθύρου εμφάνιση πίνακα

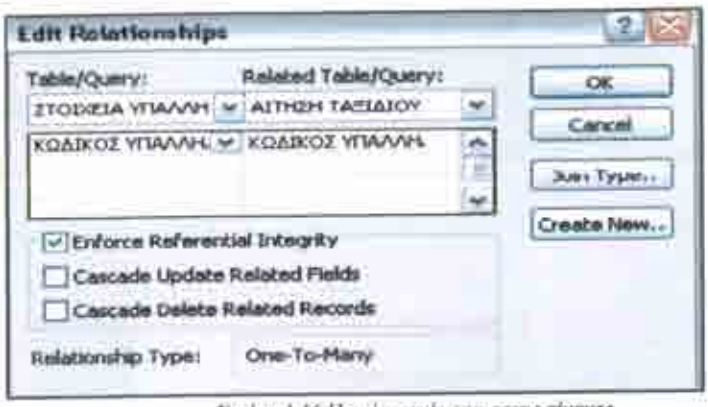


Εικόνα1.16 Το πλαίσιο διαλόγου εμφάνιση πίνακα

Επεξεργασία μιας σχέσης

Για να επεξεργαστούμε μια σχέση κάνουμε τα παρακάτω:

- Βεβαιωνόμαστε ότι όλοι οι πίνακες της βάσης δεδομένων είναι κλειστοί.
- Πατάμε το κουμπί *Σχέσεις* της γραμμής εργαλείων
- Το παράθυρο *Σχέσεις* εμφανίζεται στην οθόνη μας
- Διπλοπατάμε στην γραμμή της σχέσης που θέλουμε να επεξεργαστούμε , δηλαδή στη γραμμή που συνδέει τους δύο πίνακες.
- Ανοίγει το πλαίσιο διαλόγου Επεξεργασία Σχέσεων όπου μπορούμε να τροποποιήσουμε της επιλεγμένη σχέση.



Εικόνα 1.16 Η σχέση ανάμεσα στους πίνακες *Στοιχεία Υπαλλήλων* και *Αίτηση Ταξιδίου*

Ακεραιότητα Αναφορών

Για να αποφύγουμε την αλλοίωση των δεδομένων σε συσχετισμένους πίνακες, μπορούμε να χρησιμοποιήσουμε τη λειτουργία της *Ακεραιότητας Αναφορών*. Η λειτουργία αυτή είναι το σύνολο των κανόνων οι οποίοι εξασφαλίζουν την εγκυρότητα των σχέσεων μεταξύ των πινάκων και την ακεραιότητα των συσχετισμένων δεδομένων.

Για να μας επιτρέψει το πρόγραμμα να ενεργοποιήσουμε την λειτουργία *Ακεραιότητας Αναφορών*, πρέπει να ισχύουν τα παρακάτω :

- Το συσχετισμένο πεδίο του πρωτεύοντος πίνακα, πρέπει είτε να είναι *πρωτεύον κλειδί είτε να έχει μοναδικές εγγραφές*.
- Τα πεδία που σχετίζονται μεταξύ τους, να έχουν τον ίδιο τύπο δεδομένων.
- Οι πίνακες που συσχετίζονται θα πρέπει να βρίσκονται στην ίδια βάση δεδομένων.

Διαδοχικές ενημερώσεις και διαγραφές

Διαδοχικές ενημερώσεις σημαίνει ότι, αν κάνω κάποια αλλαγή στο πεδίο που συνδέει δύο πίνακες σε μια σχέση, στον πίνακα που έχει τη σχέση "ένα", τότε ενημερώνονται αυτόματα όλες οι εγγραφές που περιέχουν αυτό το πεδίο στον πίνακα που έχει τη σχέση "πολλά".

Διαδοχικές διαγραφές σημαίνει ότι αν σε μια σχέση που συνδέει δύο πίνακες, διαγράψω μια εγγραφή στον πίνακα που έχει τη σχέση "ένα", τότε διαγράφονται αυτόματα όλες οι εγγραφές που περιέχουν αυτό το πεδίο στον πίνακα που έχει τη σχέση "πολλά".

Διαγραφή μιας σχέσης

Για να διαγράψουμε μια σχέση που έχουμε δημιουργήσει κάνουμε τα παρακάτω :

- Επιλέγουμε τη γραμμή της σχέσης στο παράθυρο Σχέσεις.
- Πατάμε το πλήκτρο DELETE.
- Επιβεβαιώνουμε τη διαγραφή της σχέσης.

2.8 ΥΛΟΠΟΙΗΣΗ ΤΗΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

Παρακάτω παραθέτουμε τους πίνακες που χρησιμοποιήθηκαν στην εφαρμογή για την σχεδίαση της εταιρείας.

Ο πίνακας **Τμήματα** αποτελείται από τα παρακάτω πεδία, όπου ο Κωδικός Τμήματος αποτελεί το πρωτεύον κλειδί.

ΤΜΗΜΑΤΑ : Table

Field Name	Data Type	Description
ΚΩΔΙΚΟΣ ΤΜΗΜΑΤΟΣ	AutoNumber	ΚΩΔΙΚΟΣ ΑΝΑΓΝΩΡΙΣΗΣ ΤΜΗΜΑΤΟΣ
ΟΝΟΜΑ ΤΜΗΜΑΤΟΣ	Text	ΟΝΟΜΑ ΤΜΗΜΑΤΟΣ

Εικόνα 1.12 Τμήματα

Ο πίνακας **Στοιχεία Υπαλλήλων** αποτελείται από τα παρακάτω πεδία, όπου ο Κωδικός Υπαλλήλου αποτελεί το πρωτεύον κλειδί.

ΣΤΟΙΧΕΙΑ ΥΠΑΛΛΗΛΩΝ : Table

Field Name	Data Type	Description
ΚΩΔΙΚΟΣ ΥΠΑΛΛΗΛΟΥ	AutoNumber	ΚΩΔΙΚΟΣ ΑΝΑΓΝΩΡΙΣΗΣ ΥΠΑΛΛΗΛΟΥ
ΟΝΟΜΑ	Text	ΟΝΟΜΑ ΤΟΥ ΥΠΑΛΛΗΛΟΥ
ΕΠΩΝΥΜΟ	Text	ΕΠΩΝΥΜΟ ΤΟΥ ΥΠΑΛΛΗΛΟΥ
ΠΑΤΡΩΝΥΜΟ	Text	ΠΑΤΡΩΝΥΜΟ ΥΠΑΛΛΗΛΟΥ
ΤΟΠΟΣ	Text	ΤΟΠΟΣ ΥΠΑΛΛΗΛΟΥ
ΔΙΕΥΘΥΝΣΗ	Text	ΔΙΕΥΘΥΝΣΗ ΚΑΤΟΙΚΙΑΣ ΤΟΥ ΥΠΑΛΛΗΛΟΥ
T_K	Number	ΤΑΧΥΔΡΟΜΙΚΟΣ ΚΩΔΙΚΑΣ
ΤΗΛΕΦΩΝΟ	Number	ΤΗΛΕΦΩΝΟ ΤΟΥ ΥΠΑΛΛΗΛΟΥ
ΟΝΟΜΑ ΧΡΗΣΤΗ	Text	USERNAME
ΚΩΔΙΚΟΣ ΧΡΗΣΤΗ	Number	PASSWORD
ΤΜΗΜΑ	Text	ΣΕ ΠΟΙΟ ΤΜΗΜΑ ΑΝΗΚΕΙ Ο ΥΠΑΛΛΗΛΟΣ
ΚΩΔΙΚΟΣ ΤΜΗΜΑΤΟΣ	Number	

Εικόνα 1.13 Στοιχεία Υπαλλήλων

Ο πίνακας **Αίτησης ταξιδίου** αποτελείται από τα παρακάτω πεδία, όπου ο Κωδικός Αίτησης Ταξιδίου αποτελεί το πρωτεύον κλειδί.

ΑΙΤΗΣΗ ΤΑΞΙΔΙΟΥ : Table

Field Name	Data Type	Description
[redacted]	AutoNumber	ΚΩΔΙΚΟΣ ΑΝΑΓΝΩΡΙΣΗΣ ΤΑΞΙΔΙΟΥ
ΚΩΔΙΚΟΣ ΥΠΑΛΛΗΛΟΥ	Number	ΚΩΔΙΚΟΣ ΑΝΑΓΝΩΡΙΣΗΣ ΥΠΑΛΛΗΛΟΥ
ΤΟΠΟΣ	Text	ΠΡΟΟΡΙΣΜΟΣ
ΗΜΕΡΟΜΗΝΙΑ ΑΝΑΧΩΡΗΣΗΣ	Date/Time	ΗΜΕΡΟΜΗΝΙΑ ΑΝΑΧΩΡΗΣΗΣ
ΗΜΕΡΟΜΗΝΙΑ ΕΠΙΣΤΡΟΦΗΣ	Date/Time	ΗΜΕΡΟΜΗΝΙΑ ΕΠΙΣΤΡΟΦΗΣ
PENDING	Yes/No	ΠΡΟΣ ΔΙΕΚΠΕΡΑΙΩΣΗ
ΟΝΟΜΑ	Text	ΟΝΟΜΑ ΥΠΑΛΛΗΛΟΥ
ΕΠΩΝΥΜΟ	Text	ΕΠΩΝΥΜΟ ΥΠΑΛΛΗΛΟΥ

Εικόνα 1.13 Αίτηση Ταξιδίου

Ο πίνακας **Δελτίο Έγκρισης Ταξιδίου** αποτελείται από τα παρακάτω πεδία, όπου ο Κωδικός Ταξιδίου αποτελεί το πρωτεύον κλειδί.

ΔΕΛΤΙΟ ΕΓΚΡΙΣΗΣ ΤΑΞΙΔΙΟΥ : Table

Field Name	Data Type	Description
[redacted]	AutoNumber	ΚΩΔΙΚΟΣ ΑΙΤΗΣΗΣ ΤΑΞΙΔΙΟΥ
ΤΟΠΟΣ	Text	ΤΟΠΟΣ ΠΡΟΟΡΙΣΜΟΥ
ΗΜΕΡΟΜΗΝΙΑ ΑΝΑΧΩΡΗΣΗΣ	Date/Time	ΗΜΕΡΟΜΗΝΙΑ ΑΝΑΧΩΡΗΣΗΣ ΠΡΟΟΡΙΣΜΟΥ
ΩΡΑ ΑΝΑΧΩΡΗΣΗΣ	Date/Time	ΩΡΑ ΑΝΑΧΩΡΗΣΗΣ
ΗΜΕΡΟΜΗΝΙΑ ΕΠΙΣΤΡΟΦΗΣ	Date/Time	ΗΜΕΡΟΜΗΝΙΑ ΕΠΙΣΤΡΟΦΗΣ
ΩΡΑ ΕΠΙΣΤΡΟΦΗΣ	Date/Time	ΩΡΑ ΕΠΙΣΤΡΟΦΗΣ
ΞΕΝΟΔΟΧΕΙΟ	Yes/No	ΧΡΗΣΗ Ή ΜΗ ΞΕΝΟΔΟΧΕΙΟΥ
ΧΡΗΣΗ ΚΑΡΤΑΣ	Yes/No	ΧΡΗΣΗ Ή ΜΗ ΚΑΡΤΑΣ
ΠΡΟΚΑΤΑΒΟΛΗ	Yes/No	ΠΡΟΚΑΤΑΒΟΛΗ Ή ΜΗ
ΠΟΣΟ ΠΡΟΚΑΤΑΒΟΛΗΣ	Currency	ΠΟΣΟ ΠΡΟΚΑΤΑΒΟΛΗΣ
ΣΥΝΟΛΟ ΗΜΕΡΩΝ	Number	ΣΥΝΟΛΟ ΗΜΕΡΩΝ ΤΑΞΙΔΙΟΥ
ΚΩΔΙΚΟΣ ΑΙΤΗΣΗΣ	Number	ΚΩΔΙΚΟΣ ΑΝΑΓΝΩΡΙΣΗΣ ΑΙΤΗΣΗΣ
ΕΓΓΡΙΣΗ	Yes/No	ΕΓΓΡΙΣΗ

Εικόνα 1.14 Δελτίο Έγκρισης Ταξιδίου

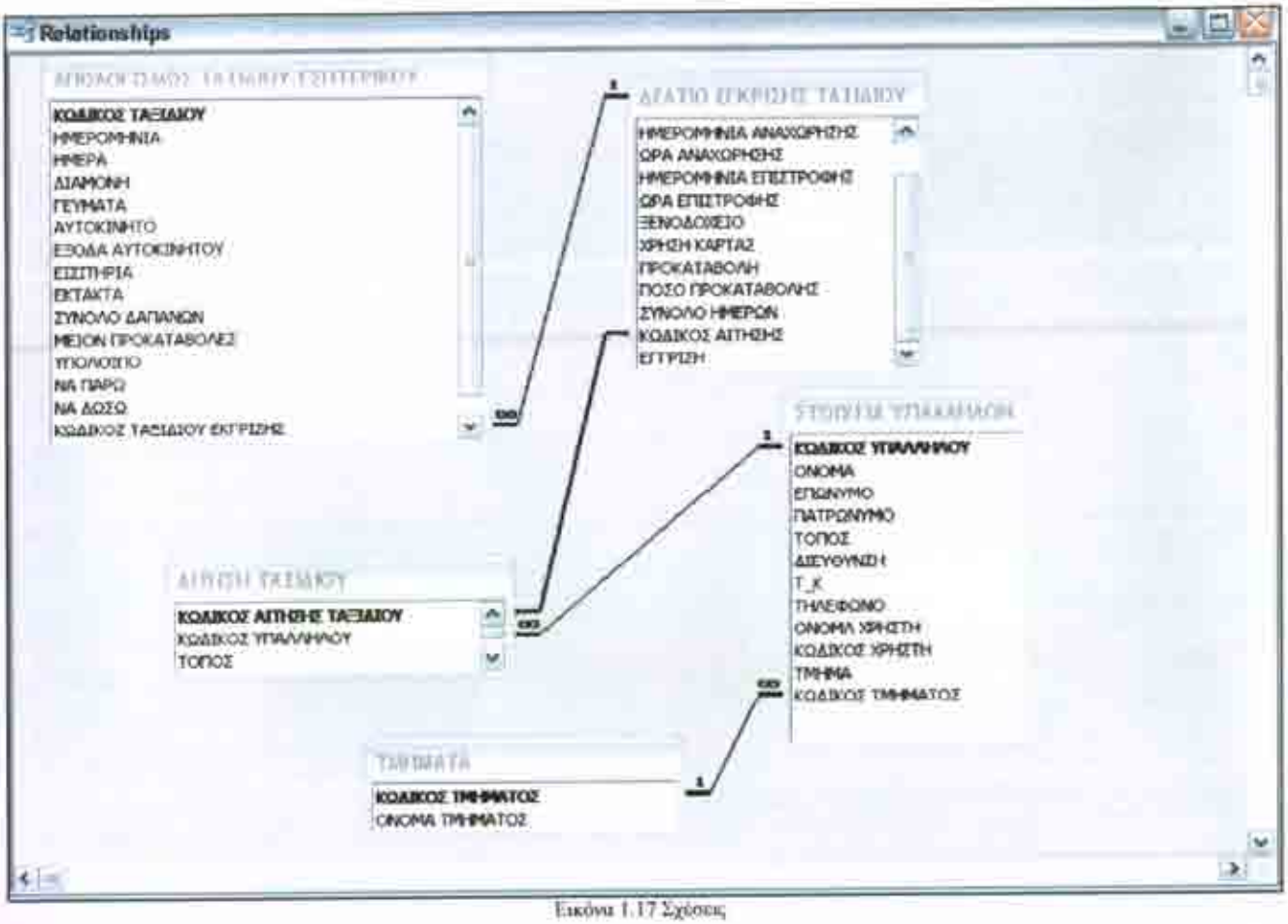
Ο πίνακας **Απολογισμός Ταξιδίου Εσωτερικού** αποτελείται από τα παρακάτω πεδία, όπου ο Κωδικός Ταξιδίου αποτελεί το πρωτεύον κλειδί.

ΑΠΟΛΟΓΙΣΜΟΣ ΤΑΞΙΔΙΟΥ ΕΣΩΤΕΡΙΚΟΥ : Table

Field Name	Data Type	Description
[redacted]	AutoNumber	ΚΩΔΙΚΟΣ ΑΝΑΓΝΩΡΙΣΗΣ ΤΑΞΙΔΙΟΥ
ΗΜΕΡΟΜΗΝΙΑ	Date/Time	ΗΜΕΡΟΜΗΝΙΑ
ΗΜΕΡΑ	Text	ΗΜΕΡΑ
ΔΙΑΜΟΝΗ	Currency	ΕΞΟΔΑ ΔΙΑΜΟΝΗΣ
ΓΕΥΜΑΤΑ	Currency	ΕΞΟΔΑ ΓΕΥΜΑΤΩΝ
ΑΥΤΟΚΙΝΗΤΟ	Yes/No	ΧΡΗΣΗ Η ΜΗ ΑΥΤΟΚΙΝΗΤΟΥ
ΕΞΟΔΑ ΑΥΤΟΚΙΝΗΤΟΥ	Currency	ΧΙΛΙΟΜΕΤΡΑ ΒΕΝΖΙΝΗ PARKING
ΕΙΣΙΤΗΡΙΑ	Currency	ΕΞΟΔΑ ΕΙΣΙΤΗΡΙΩΝ
ΕΚΤΑΚΤΑ	Currency	ΕΚΤΑΚΤΑ ΕΞΟΔΑ
ΣΥΝΟΛΟ ΔΑΠΑΝΩΝ	Currency	ΣΥΝΟΛΟ ΔΑΠΑΝΩΝ
ΜΕΙΟΝ ΠΡΟΚΑΤΑΒΟΛΕΣ	Currency	ΜΕΙΟΝ ΠΟΣΟ ΠΡΟΚΑΤΑΒΟΛΗΣ
ΥΠΟΛΟΙΠΟ	Yes/No	ΥΠΟΛΟΙΠΟ ΔΑΠΑΝΩΝ
ΝΑ ΠΑΡΩ	Currency	ΧΡΕΩΣΤΙΚΟ ΥΠΟΛΟΙΠΟ
ΝΑ ΔΩΣΩ	Currency	ΠΙΣΤΩΤΙΚΟ ΥΠΟΛΟΙΠΟ
ΚΩΔΙΚΟΣ ΤΑΞΙΔΙΟΥ ΕΚΤΡΙΣΗ	Number	
Approval	Yes/No	ΕΓΚΡΙΣΗ ΑΠΟΛΟΓΙΣΜΟΥ

Εικόνα 1.15 Απολογισμός Ταξιδίου Εσωτερικού

Παρακάτω παραθέτουμε τις σχέσεις που χρησιμοποιήθηκαν στην εφαρμογή της εργασίας μας ώστε να γίνει η σύνδεση των πινάκων :



Εικόνα 1.17 Σχέσεις

2.9 ΕΡΩΤΗΜΑΤΑ

Τα ερωτήματα είναι μια από τις σημαντικότερες δυνατότητες των Βάσεων Δεδομένων. Το βασικότερο πλεονέκτημά τους είναι ότι μας επιτρέπουν να εντοπίζουμε και να εμφανίζουμε ακριβώς τις πληροφορίες που θέλουμε από τις Βάσεις Δεδομένων μας.

2.9.1 Δημιουργία ερωτήματος

Για να δημιουργήσουμε ένα ερώτημα κάνουμε τα παρακάτω :

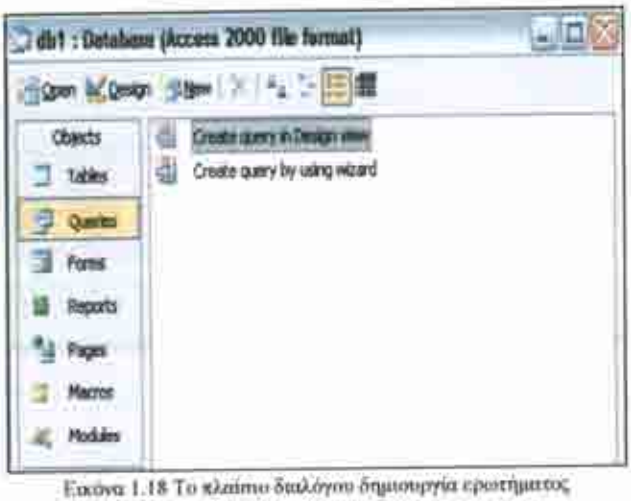
- Πατάμε στο κουμπί *Ερωτήματα* του τμήματος *Αντικείμενα* του παραθύρου Βάσης Δεδομένων
- Πατάμε στο κουμπί *Δημιουργία* της γραμμής εργαλείων οπότε στην οθόνη μας εμφανίζεται

το πλαίσιο διαλόγου *Δημιουργία Ερωτήματος*.

Στο πλαίσιο διαλόγου *Δημιουργία Ερωτήματος* επιλέγουμε το είδος του ερωτήματος που θέλουμε να δημιουργήσουμε :

- Η επιλογή *Προβολή Σχεδίασης* μας επιτρέπει να δημιουργήσουμε ένα ερώτημα, καθορίζοντας μόνοι μας τους πίνακες, τα πεδία, αλλά και τα κριτήρια που θα χρησιμοποιήσουμε .

- Η επιλογή *Οδηγός Απλών Ερωτημάτων* μας επιτρέπει να δημιουργήσουμε ένα ερώτημα το οποίο ρυθμίζεται κατάλληλα από το πρόγραμμα μετά από επιλογές που κάνουμε στα πλαίσια διαλόγου.



Εικόνα 1.18 Το κλείσιμο διαλόγου δημιουργία ερωτήματος

2.9.2 Κριτήρια

Κριτήρια είναι οι περιορισμοί που ορίζουμε στα ερωτήματα ώστε να επιστρέφουν από τη Βάση Δεδομένων μόνο οι εγγραφές που μας ενδιαφέρουν. Συγκεκριμένα, Τα κριτήρια είναι παραστάσεις που, στη απλούστερη και πιο συνηθισμένη μορφή τους, περιέχουν σταθερές τιμές και τελεστές. Μερικοί από τους τελεστές που χρησιμοποιούνται συχνά σε κριτήρια υπάρχουν στον πίνακα που ακολουθεί.

ΤΕΛΕΣΤΕΣ ΣΥΓΚΡΙΣΗΣ		
Τελεστής	Παράδειγμα	Επιστρέφονται οι εγγραφές των οποίων το συγκεκριμένο πεδίο έχει τιμή.
Between	Between 20 And 30	Μεταξύ 20 και 30
In	In (Πάτρα, Αθήνα)	Ίδια με μια από τις τιμές μέσα στις παρενθέσεις
=	=35	Ίση με 35
>	>36	Μεγαλύτερη από 36
<	<40	Μικρότερη από 40
<>	<>0	Διάφορο του 0
>=	>=50	Μεγαλύτερη ή ίση με 50
<=	<=46	Μικρότερη ή ίση με 46

ΛΟΓΙΚΟΙ ΤΕΛΕΣΤΕΣ		
Τελεστής	Παράδειγμα	Επιστρέφονται οι εγγραφές των οποίων το συγκεκριμένο πεδίο έχει τιμή.
And	<50 And >40	Μικρότερη από 50 και Μεγαλύτερη από 40
Or	48 Or 52	48 ή 52
Not	Not "ΜΑΡΙΑ"	Εκτός "ΜΑΡΙΑ"

ΧΡΗΣΗ ΤΕΛΕΣΤΗ Like	
Κριτήριο	Επιστρέφει τις εγγραφές των οποίων το αντίστοιχο πεδίο :
Like "Γ*"	Αρχίζει με το γράμμα Γ
Like "*ΟΥ"	Τελειώνει σε ΟΥ
Like "?Β*"	Το Δεύτερο γράμμα είναι το Β
Like "*ΝΑ"	Περιέχει το ζεύγος ΝΑ
Like "[Α-Γ]*"	Αρχίζει με τους χαρακτήρες Α ΕΩΣ Γ
Like "2#5"	Τριψήφιοι αριθμοί που ξεκινούν από 2 και τελειώνουν σε 5

2.9.3 Εκτέλεση – Διαγραφή Ερωτήματος

Όταν έχουμε ορίσει τα κριτήρια ενός ερωτήματος, μπορούμε να το εκτελέσουμε για να εξετάσουμε τα αποτελέσματα του :

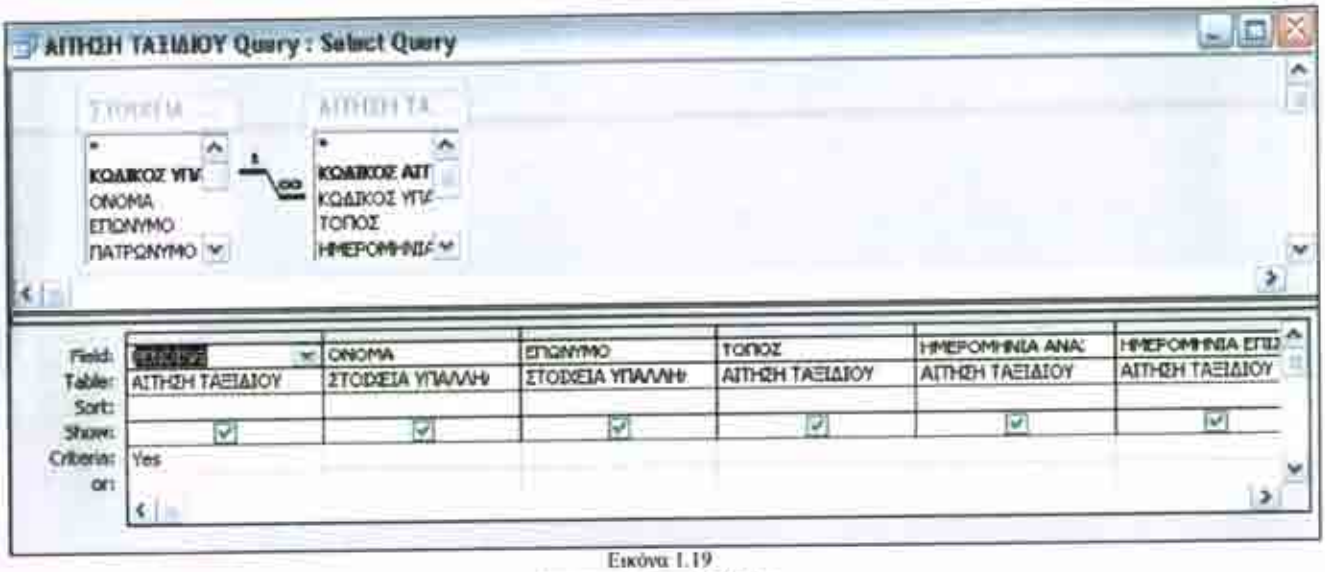
- Πατάμε το κουμπί *εκτέλεση* στην γραμμή εργαλείων.
- Τα αποτελέσματα του ερωτήματος εμφανίζονται στην οθόνη μας σε *Προβολή Φύλλου Δεδομένων.*

Για να διαγράψουμε ένα ερώτημα, έχουμε τις εξής επιλογές :

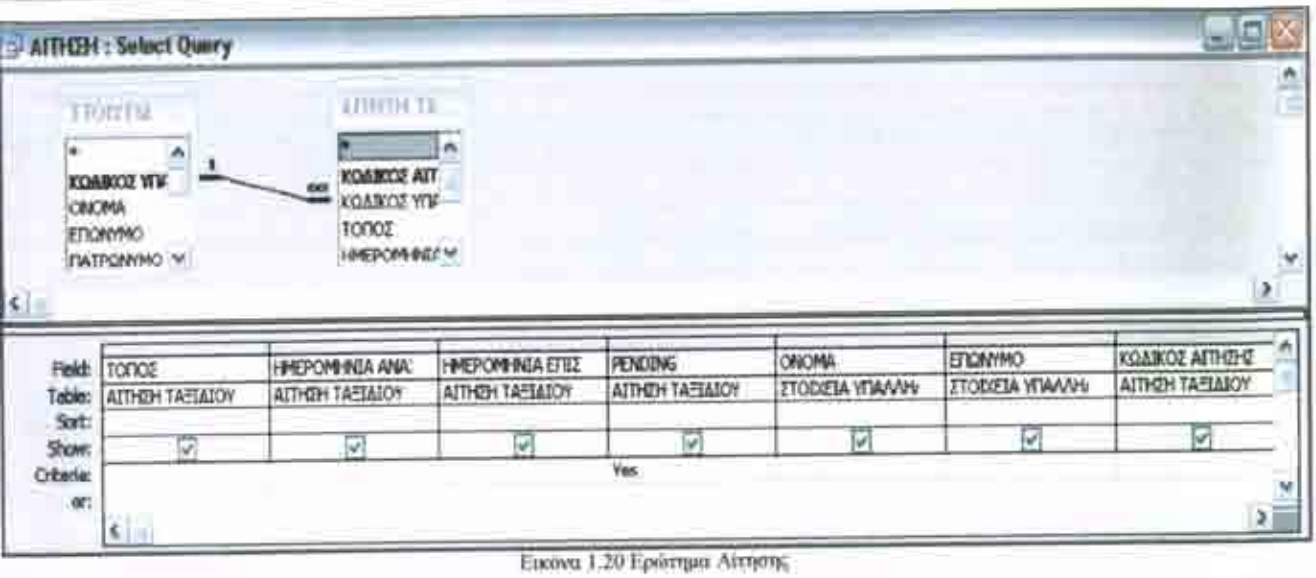
- Επιλέγουμε το ερώτημα στο παράθυρο Βάσης Δεδομένων και μετά διαλέγουμε την εντολή Διαγραφή από το μενού Επεξεργασία
- Εμφανίζουμε το ερώτημα στο παράθυρο Βάσης Δεδομένων και με το δεξί πλήκτρο του ποντικιού επιλέγουμε την εντολή Διαγραφή
- Στην συνέχεια απλώς επιβεβαιώνουμε την διαγραφή του ερωτήματος.

2.10 ΕΡΩΤΗΜΑΤΑ ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΗΘΗΚΑΝ ΣΤΗΝ MICROSOFT ACCESS

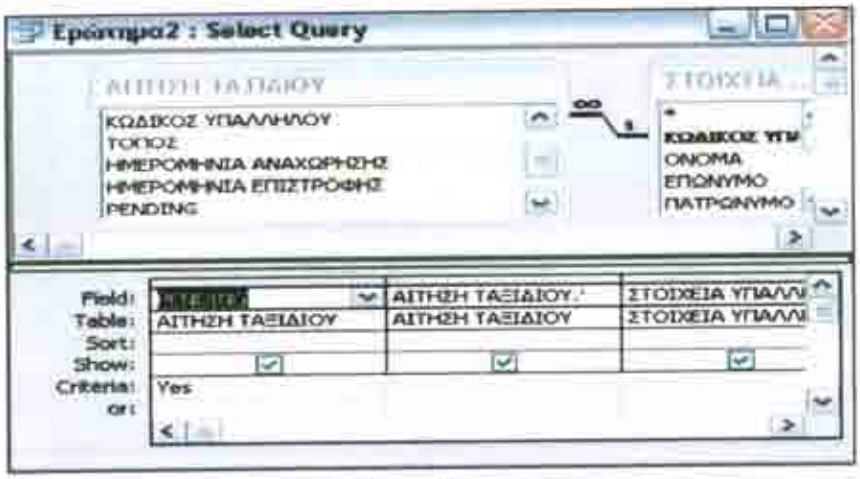
Παρακάτω παραθέτουμε τα ερωτήματα, που σχεδιάσαμε με προβολή σχεδίασης για να εντοπίζονται και να εμφανίζονται τυχόν εκκρεμής αιτήσεις.



Εικόνα 1.19
Ερώτημα Αίτησης Ταξιδίου



Εικόνα 1.20 Ερώτημα Αίτησης



Εικόνα 1.21 Ερώτημα Εκκρεμών Αιτήσεων

3 ΤΟ DELPHI 7.0 ΚΑΙ ΤΟ ΠΕΡΙΒΑΛΛΟΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ ΤΟΥ

3.1 ΑΝΤΙΚΕΙΜΕΝΟ ΤΟΥ ΚΕΦΑΛΑΙΟΥ

Σκοπός αυτού του κεφαλαίου είναι να κατανοήσουμε την γλώσσα προγραμματισμού του Delphi 7.0 Enterprise. Δίνουμε ιδιαίτερη βαρύτητα στην περιγραφή, αλλά και στο περιβάλλον εργασίας του Delphi 7.0 αναλύοντας της προηγούμενες εκδόσεις και επισημαίνοντας τις βασικές διαφορές μεταξύ τους αλλά και σημαντικές προσθήκες,.

Θα καλύψουμε τις βάσεις για τα πιο συνηθισμένα στοιχεία ελέγχου, την ανάπτυξη προχωρημένου περιβάλλοντος χρήστη, καθώς επίσης και την αρχιτεκτονική των εφαρμογών Delphi και την χρήση βιβλιοθηκών και πακέτων.

3.2 ΤΙ ΕΙΝΑΙ ΤΟ DELPHI

Το Delphi είναι μια εφαρμογή των Windows που στηρίζεται στον *αντικειμενοστραφή προγραμματισμό (object-oriented programming)* και στη γλώσσα προγραμματισμού *Pascal*. Όποιος έχει μια εμπειρία από άλλες αντικειμενοστραφείς εφαρμογές, όπως είναι η Visual Basic, θα βρει αρκετές ομοιότητες στην ανάπτυξη των εφαρμογών.

Αντίθετα με τα παλαιότερα, «παραδοσιακά» συστήματα ανάπτυξης εφαρμογών, το Delphi είναι *καθοδηγούμενο από συμβάντα (event driven)*, που σημαίνει ότι δεν εκτελεί διαδοχικά τις εντολές ενός προγράμματος, από την αρχή μέχρι το τέλος, αλλά περιμένει την εμφάνιση συμβάντων (events) και εκτελεί τον κώδικα του προγράμματος (διαδικασία ή συνάρτηση) που σχετίζεται με τα συμβάντα αυτά.

Ένα *συμβάν (event)* είναι μια ενέργεια που προέρχεται από τον χρήστη ή από το σύστημα και παραδείγματα συμβάντων είναι το πάτημα (κλικ) με το ποντίκι πάνω σ' ένα πλήκτρο εντολής, η μετακίνηση του ποντικιού πάνω από ένα αντικείμενο, η επιλογή ενός αντικειμένου σ' ένα μενού κ.ά.

3.3 ΕΚΔΟΣΕΙΣ ΤΟΥ DELPHI

Το Delphi είναι ένα σπουδαίο εργαλείο αλλά έχει επίσης ένα περίπλοκο περιβάλλον προγραμματισμού που περιλαμβάνει πολλά στοιχεία. Αποτελείται από επτά εκδόσεις οι οποίες αναλύονται παρακάτω :

Μερικά από τα αρχικά χαρακτηριστικά του Delphi ήταν η προσέγγιση που βασιζόταν στις φόρμες και στον αντικειμενοστραφή προγραμματισμό, ο γρήγορος μεταγλωττιστής, οι δυνατότητες υποστήριξης βάσεων δεδομένων, η ενσωμάτωση με τον προγραμματισμό των Windows και οι τεχνολογίες συστατικών. Αλλά το πιο σημαντικό στοιχείο ήταν η γλώσσα Object Pascal, η οποία αποτελεί το θεμέλιο για όλα τα παραπάνω στοιχεία.

Το Delphi 2 ήταν ακόμα καλύτερο. Μεταξύ των πιο σημαντικών προσθηκών ήταν : το βελτιωμένο πλέγμα βάσης δεδομένων, το Multi Record Object, η υποστήριξη αυτοματισμού Ole, οι τύποι δεδομένων Variant, Long String, η Visual Form Inheritance και η πλήρης ενσωμάτωση με τον προγραμματισμό των Windows 95.

Το Delphi 3 είχε σημαντικές προσθήκες, όπως τεχνολογία Code Insight, πρότυπα συστατικών, υποστήριξη αποσφαλμάτωσης DLL, το Tree Chart, το Decision Cube, την τεχνολογία Web Broker, τα πακέτα συστατικών, τις Active Form και τέλος μια εκπληκτική ενοποίηση με το Com, χάρη στις διασυνδέσεις.

Το Delphi 4 μας έδωσε τον επεξεργαστή AppBrowser, εκτεταμένα χαρακτηριστικά βάσεων δεδομένων, βελτιωμένη υποστήριξη Ole και Com, καθώς και σημαντικές προσθήκες στις κλάσεις VCL, συμπεριλαμβανομένης και της υποστήριξης για την σταθεροποίηση, περιορισμό των στοιχείων ελέγχου, πλήρης ενσωμάτωση με τον προγραμματισμό των Windows 98.

Το Delphi 5 πρόσθεσε πολλές βελτιώσεις στο περιβάλλον IDE, εκτεταμένη υποστήριξη για βάσεις δεδομένων, δυνατότητες μετάφρασης, τα πλαίσια, το εργαλείο ελέγχου εκδόσεων Team Source και μια βελτιωμένη έκδοση του Midas με υποστήριξη Internet.

Το Delphi 6 πρόσθεσε σε όλες αυτές τις λειτουργίες υποστήριξη για ανάπτυξη σε πολλαπλές πλατφόρμες με το Component Library fro Cross – Platform (CIX), μια εκτεταμένη βιβλιοθήκη εκτέλεσης, την νέα μηχανή βάσεων δεδομένων dbExpress, Web υπηρεσίες με δυνατή βάση ανάπτυξης τους και εξαιρετική υποστήριξη XML, περισσότερες βελτιώσεις στο IDE και πληθώρα νέων συστατικών και κλάσεων.

Το Delphi 7 έκανε μερικές από αυτές τις νεώτερες τεχνολογίες πιο δυνατές, με βελτιώσεις και διορθώσεις και υποστηρίζει τα θέματα των Windows XP,αλλά το πιο σημαντικό είναι ότι έκανε άμεσα διαθέσιμο ένα ενδιαφέρον σύνολο από εργαλεία τρίτων κατασκευαστών : την τεχνολογία ανάπτυξης Web, τη μηχανή αναφορών Rave και το περιβάλλον σχεδίασης Model Maker. Τέλος ανοίγει ένα τελείως νέο κόσμο τον πρώτο μεταγλωττιστή Borland. Για την γλώσσα Pascal / Delphi που έχει στόχο την πλατφόρμα .NET CIL.

Υπάρχουν τέσσερις εκδόσεις του Delphi :

- Η έκδοση *Personal* (Προσωπική) είναι χρήσιμοι για τους αρχάριους και τους περιστασιακούς προγραμματιστές του Delphi και δεν έχει υποστήριξη για τις προχωρημένες λειτουργίες.
- Η έκδοση *Professional Studio* (Επαγγελματική) είναι για τους επαγγελματίες προγραμματιστές. Συμπεριλαμβάνει όλα τα βασικά χαρακτηριστικά, συν επεκταμένη υποστήριξη για τον προγραμματισμό Βάσεων Δεδομένων και κάποια εξωτερικά εργαλεία, όπως το Model Maker και IntraWeb.
- Η έκδοση *Enterprise Studio* (Πλήρης), απευθύνεται στους προγραμματιστές που σχεδιάζουν επαγγελματικές εφαρμογές. Περιλαμβάνει όλες τις νέες XML και προχωρημένες Web τεχνολογίες, υποστήριξη CORBA, διεθνοποίηση, αρχιτεκτονική τριών βαθμίδων και πολλά άλλα εργαλεία.
- Η έκδοση *Architect Studio* έχει επιπλέον από την έκδοση Enterprise υποστήριξη για το Bold, ένα περιβάλλον για δημιουργία εφαρμογών που καθοδηγούνται κατά την εκτέλεση από ένα μοντέλο UML και με ικανότητα να απεικονίζει τα αντικείμενά του και σε μια βάση δεδομένων και σε ένα περιβάλλον χρήστη, χάρη σε μια πληθώρα από προχωρημένα συστατικά.

3.4 ΤΟ ΠΕΡΙΒΑΛΛΟΝ ΤΟΥ DELPHI 7.0

Σε ένα οπτικό πεδίο προγραμματισμού όπως είναι το Delphi ο ρόλος του ολοκληρωμένου περιβάλλοντος ανάπτυξης (*Integrated Development Environment – IDE*) είναι σημαντικός, κάποιες φορές μάλιστα πολύ περισσότερο από την γλώσσα προγραμματισμού. Το Delphi 7 παρουσιάζει κάποια ενδιαφέροντα νέα χαρακτηριστικά πάνω από το πλούσιο IDE του Delphi 6. Το ολοκληρωμένο περιβάλλον ανάπτυξής του είναι αρκετά απλό στην χρήση του.

Όταν δουλεύουμε με ένα οπτικό περιβάλλον ανάπτυξης υπάρχουν δύο οπτικά μέρη της εφαρμογής : στους *οπτικούς σχεδιαστές* και στον *επεξεργαστή κώδικα*. Οι σχεδιαστές μας επιτρέπουν να δουλεύουμε με συστατικά σε οπτικό επίπεδο (όπως όταν τοποθετούμε ένα κουμπί σε μια φόρμα) ή σε μη οπτικό επίπεδο (όπως όταν τοποθετούμε ένα συστατικό Data Set σε μια λειτουργική μονάδα δεδομένων). Και στις δύο περιπτώσεις, οι σχεδιαστές μας επιτρέπουν να επιλέγουμε τα συστατικά που χρειαζόμαστε και να ορίζουμε την αρχική τιμή των ιδιοτήτων των συστατικών.

Ο επεξεργαστής κώδικα είναι εκεί που γράφουμε τον κώδικα. Ο πιο προφανής τρόπος να γράψουμε κώδικα σε ένα οπτικό περιβάλλον περιλαμβάνει την απόκριση σε συμβάντα, ξεκινώντας με συμβάντα που είναι συνδεδεμένα σε λειτουργίες που εκτελούνται από τους χρήστες του προγράμματος όπως το κλικ σε ένα κουμπί ή η επιλογή ενός στοιχείου από ένα πλαίσιο λίστας. Μπορούμε να χρησιμοποιήσουμε την ίδια προσέγγιση για να χειριστούμε εσωτερικά συμβάντα, όπως συμβάντα που περιλαμβάνουν αλλαγές στην Βάση Δεδομένων ή ειδοποιήσεις από το λειτουργικό σύστημα.

Το IDE του Delphi επιτρέπει στους προγραμματιστές να το προσαρμόζουν με διάφορους τρόπους, γενικά, ανοίγοντας πολλά παράθυρα, διατάσσοντάς τα και σταθεροποιώντας τα. Ωστόσο πολλές φορές θα χρειαστεί να ανοίξουμε ένα σύνολο παραθύρων κατά την διάρκεια της σχεδίασης και ένα διαφορετικό σύνολο κατά την αποσφαλμάτωση. Παρόμοια μπορεί να χρειαστούμε μια διάταξη όταν δουλεύουμε με φόρμες και μια εντελώς διαφορετική διάταξη όταν γράφουμε συστατικά ή κάποιον κώδικα χαμηλού επιπέδου χρησιμοποιώντας μόνο τον επεξεργαστή.

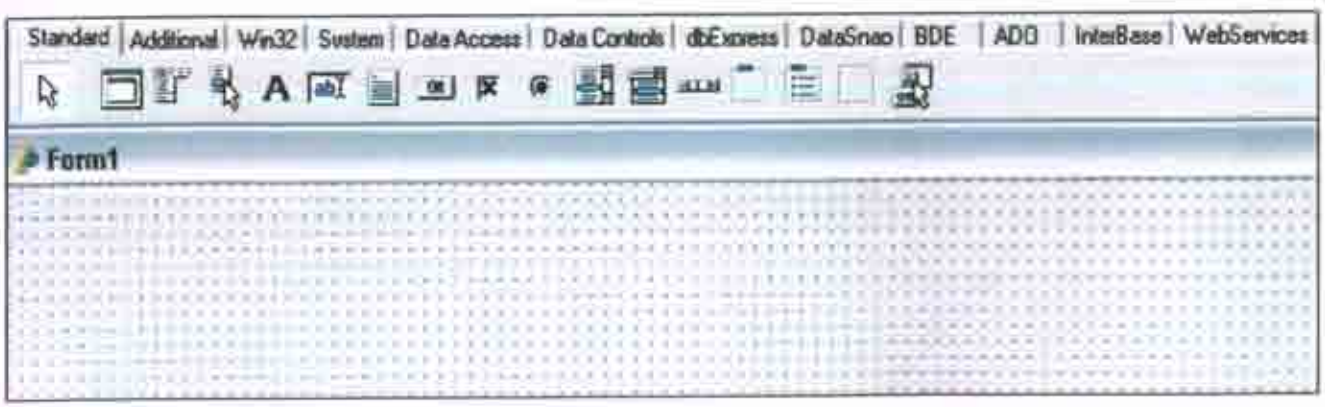
Η αναδιάταξη του περιβάλλοντος IDE αποτελεί μια επίμονη εργασία γιατί τον λόγο κάθε φορά που έχουμε να κάνουμε με μια διάταξη του περιβάλλοντος των παραθύρων του IDE (που ονομάζεται επιφάνεια εργασίας, ή καθολική επιφάνεια εργασίας σε αντίθεση με την επιφάνεια

εργασίας του έργου) μπορούμε να αποθηκεύσουμε την διάταξη με ένα όνομα και να την επαναφέρουμε στην συνέχεια αρκετά εύκολα.

3.5 ΠΕΡΙΓΡΑΦΗ ΤΟΥ DELPHI 7.0

Μόλις φορτωθεί το Delphi, θα δούμε την οθόνη να χωρίζεται σε κάποια τμήματα. Στην κορυφή του παραθύρου υπάρχουν τα γνωστά μας πτυσσόμενα μενού (File, Edit, Search, View, Compile, Run, Options, Tools, Help) και στην άνω αριστερή γωνία υπάρχει μια χρήσιμη γραμμή εργαλείων με εικονίδια για το άνοιγμα και την αποθήκευση του έργου (project) και των αρχείων μιας εφαρμογής σε Delphi.

Στο μέσο της οθόνης εμφανίζεται ένα παράθυρο με τον τίτλο ***Form1***, που είναι η προκαθορισμένη φόρμα της εφαρμογής. Τις φόρμες τις χρησιμοποιούμε για να τοποθετούμε εκεί τα αντικείμενα γραφικών της εφαρμογής μας. Πάνω από τη φόρμα βρίσκεται μια παλέτα με πολλές καρτέλες (Standard, Additional, System, Win95, Dialogs, Data Access, Data Controls, Win 3.1, Internet, Samples κ.ά.). Οι καρτέλες αυτές περιέχουν τα αντικείμενα ή εξαρτήματα (components) που μπορούμε να προσθέσουμε στις φόρμες



Ο παρακάτω πίνακας παρουσιάζει τα εξαρτήματα που υπάρχουν στην καρτέλα Standard της παλέτας εξαρτημάτων του Delphi.

Εξάρτημα	Περιγραφή
Main Menu	Προσθέτει σε μια φόρμα μια γραμμή μενού
Popup Menu	Συσχετίζει ένα αναδυόμενο μενού με μια φόρμα
Label	Εμφανίζει μια ετικέτα
Edit	Εμφανίζει ένα πλαίσιο κειμένου
Memo	Εμφανίζει μια περιοχή σύνταξης πολλών γραμμών
Button	Προσθέτει ένα πλήκτρο εντολής
Checkbox	Προσθέτει ένα πλαίσιο ελέγχου
Radio Button	Προσθέτει ένα πλήκτρο επιλογής
List Box	Δημιουργεί ένα πλαίσιο λίστας
Combo Box	Δημιουργεί ένα σύνθετο πλαίσιο
Scrollbar	Δημιουργεί μια γραμμή κύλισης
Group Box	Ομαδοποιεί άλλα εξαρτήματα
Radio Group	Ομαδοποιεί πλήκτρα επιλογής
Panel	Δημιουργεί έναν πίνακα εξαρτημάτων

Στην καρτέλα ***Additional*** της γραμμής εργαλείων υπάρχει το εξάρτημα ***Bit Bin***, με το οποίο μπορούμε να δημιουργήσουμε ένα πλήκτρο εντολής που να περιέχει μια μικρή επεξηγηματική εικόνα μέσα του, της μορφής bitmap.

Ένα πλήκτρο Bit Bin εμφανίζεται αρχικά χωρίς γραφικό, αλλά από την ιδιότητά του ***Kind*** μπορούμε να επιλέξουμε μια από πολλές προκαθορισμένες μορφές, που εκτός από την εικόνα, ρυθμίζουν αυτόματα και άλλες λειτουργίες. Οι προκαθορισμένες αυτές μορφές είναι οι εξής : bkOk, bkYes, bkHelp, bkAbort, bkIgnore, bkCancel, bkNo, bkClose, bkRetry και bkAll.

Τέλος, υπάρχει και το παράθυρο του *Object Inspector*, που εμφανίζει τις ιδιότητες του αντικειμένου που έχουμε επιλέξει και έχει δύο καρτέλες : Properties και Events. Η καρτέλα *Properties* εμφανίζει τις ιδιότητες του επιλεγμένου αντικειμένου, όπως χρώμα, ύψος, πλάτος κ.ά. και η καρτέλα *Events* εμφανίζει τα συμβάντα που σχετίζονται με το επιλεγμένο αντικείμενο, όπως *On Click, OnDblClick, OnKeyDown, OnMouseMove* κ.ά. Οι επιλογές και για τις δύο αυτές καρτέλες διαφέρουν ανάλογα με τον τύπο του επιλεγμένου αντικειμένου.

3.6 ΔΗΜΙΟΥΡΓΙΑ ΜΙΑΣ ΑΠΛΗΣ ΕΦΑΡΜΟΓΗΣ

Κατά την δημιουργία ενός έργου, ή κατά το άνοιγμα ενός υπάρχοντος η παλέτα Componentη διατάσσεται, ώστε να εμφανίζει μόνο τα στοιχεία ελέγχου που σχετίζονται με την τρέχουσα βιβλιοθήκη.

Για να κάνουμε μια εφαρμογή στο Delphi, επιλέγουμε τη φόρμα πάνω στην οποία θα εργαστούμε, προσθέτουμε στη φόρμα τα αντικείμενα που θέλουμε, όπως πλήκτρα εντολών, πλαίσια κειμένου, ετικέτες κ.ά., ρυθμίζουμε τις ιδιότητες των εξαρτημάτων με τον Object Inspector και γράφουμε τον κατάλληλο κώδικα προγράμματος για τα διάφορα συμβάντα.

Για να προσθέσουμε ένα πλήκτρο εντολής (command button) σε μια φόρμα, μπορούμε να κάνουμε διπλό κλικ πάνω στο εικονίδιο της καρτέλας Standard που γράφει OK, οπότε θα προστεθεί ένα πλήκτρο εντολής στο κέντρο της φόρμας με προκαθορισμένες διαστάσεις ή να κάνουμε απλό κλικ στο ίδιο εικονίδιο και μετά να σχεδιάσουμε το πλήκτρο όπως το θέλουμε ή να κάνουμε απλό κλικ στο ίδιο εικονίδιο και μετά κλικ μέσα στη φόρμα, οπότε θα προστεθεί ένα πλήκτρο εντολής στο σημείο που κάναμε κλικ και με προκαθορισμένες διαστάσεις

Αφού επιλέξουμε το πλήκτρο εντολής, μπορούμε να αλλάξουμε την ιδιότητά του Caption ή την ιδιότητά του Name από τον Object Inspector. Η ιδιότητα ***Caption*** είναι η ετικέτα (τίτλος) που εμφανίζεται πάνω του, ενώ η ιδιότητα ***Name*** είναι το όνομά του, με το οποίο αναφερόμαστε στο πλήκτρο εντολής στα διάφορα συμβάντα που θα δημιουργήσουμε γι' αυτό. Εξ ορισμού και τα δύο έχουν αρχικά την τιμή Button1, Button2, κοκ.

Αφού προσθέσουμε ένα πλήκτρο εντολής στη φόρμα, το τοποθετήσουμε εκεί που θέλουμε, ρυθμίσουμε το πλάτος και το ύψος του και τις ιδιότητες Caption και Name, κάνουμε διπλό κλικ πάνω του για να ανοίξει το παράθυρο κώδικα όπου θα γράψουμε τις εντολές που θα εκτελεστούν όταν θα κάνουμε κλικ στο πλήκτρο αυτό.

Οι παρακάτω γραμμές κώδικα εμφανίζονται αυτόματα :

```
Procedure TForm1.Button1Click(Sender: TObject);
begin

end;
```

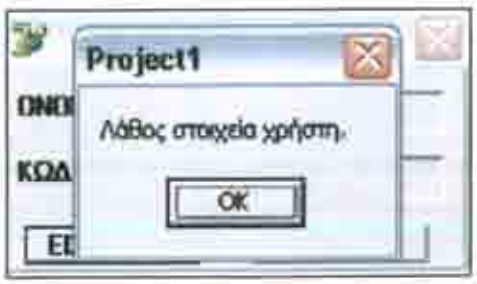
Βλέπουμε ότι το Delphi δημιουργεί μόνο του μια διαδικασία με όνομα TForm1.Button1Click, που σημαίνει ότι περιέχει τον κώδικα που θα εκτελεστεί όταν κάνουμε Click στο πλήκτρο εντολής που έχει το όνομα (Name) Button1 και ανήκει στην τρέχουσα φόρμα Form1.

Για να δημιουργήσουμε τώρα το πρώτο μας πρόγραμμα σε Delphi, ανάμεσα στις δεσμευμένες λέξεις ***begin*** και ***end***, γράφουμε τις εξής εντολές :

ShowMessage('Λάθος στοιχεία χρήστη');
Close;

Η εντολή (διαδικασία) ***ShowMessage()*** εμφανίζει ένα ***επιτακτικό*** *(modal)* παράθυρο, που περιέχει το μήνυμα *Λάθος στοιχεία χρήστη* και ένα πλήκτρο εντολής με τον τίτλο OK και στο οποίο πρέπει να κάνουμε κλικ για να μπορέσουμε να συνεχίσουμε. Προσέξτε τη χρήση του χαρακτήρα ; στο τέλος της κάθε εντολής και τη χρήση των μονών εισαγωγικών (' ') για τα μηνύματα.

Για να τρέξει τώρα αυτή η εφαρμογή, μπορούμε να επιλέξουμε την εντολή ***Run*** από το μενού ***Run*** ή να πατήσουμε το πλήκτρο ***F9*** ή να πατήσουμε το πλήκτρο με τον τίτλο (ToolTip) *Run* της γραμμής εργαλείων, που έχει το σχήμα4. Μόλις εκτελεστεί η εφαρμογή, μπορούμε να κάνουμε κλικ στο πλήκτρο εντολής για να εμφανιστεί ένα μικρό παράθυρο διαλόγου με το μήνυμα *Λάθος στοιχεία χρήστη*, με τον τίτλο PROJECT1.EXE και με ένα πλήκτρο εντολής με τον τίτλο *OK*, το οποίο θα περιμένει να κάνουμε κλικ στο πλήκτρο αυτό εντολής για να συνεχίσει η εφαρμογή μας, δηλ. στην ουσία να τελειώσει με την εντολή Close.



Εικόνα 3.1 Παράδειγμα Εντολής

3.7 Η ΠΡΟΣΘΗΚΗ ΣΧΟΛΙΩΝ

Για να γράψουμε σχόλια (comments) στο Delphi, που όλοι ξέρουμε πόσο σημαντικά είναι για ένα σωστό και ολοκληρωμένο πρόγραμμα, μπορούμε να επιλέξουμε έναν από τους εξής τρεις τρόπους :

{ με τη χρήση των αγκίστρων δεν πρέπει να ξεχάσουμε να κλείσουμε τα σχόλια }

(* τα σχόλια αυτά ανοίγουν με παρένθεση-αστερίσκο και κλείνουν με αστερίσκο-παρένθεση *)

// αυτά τα σχόλια ισχύουν μέχρι το τέλος μόνο της τρέχουσας γραμμής

3.8 ΤΑ ΠΛΑΙΣΙΑ ΛΙΣΤΑΣ

Το ***πλαίσιο λίστας*** *(listbox)* εμφανίζει μια λίστα από επιλογές και κατά την εκτέλεση μπορούμε να επιλέξουμε κάποιο απ' αυτά τα στοιχεία με το ποντίκι. Για να προσθέσουμε ένα στοιχείο σ' ένα πλαίσιο λίστας, μπορούμε να πάρουμε το κείμενο από ένα πλαίσιο κειμένου ή να χρησιμοποιήσουμε ένα σταθερό κείμενο. Για να γίνει αυτό, χρησιμοποιούμε τη μέθοδο ***Add*** της ιδιότητας ***Items*** ενός πλαισίου λίστας :

```
ListBox1.Items.Add (Edit1.Text);
ListBox1.Items.Add ('ΚΩΔΙΚΟΣ ΤΜΗΜΑΤΟΣ');
```

Πολλές φορές θα χρειαστεί να αρχικοποιήσουμε μια λίστα, δηλ. να της δώσουμε αρχικές τιμές κατά το ξεκίνημα μιας εφαρμογής. Προς το σκοπό αυτό κάνουμε διπλό κλικ στη φόρμα κατά τη σχεδίαση και καλείται το συμβάν ***FormCreate***, όπου και γράφουμε τον κώδικα αρχικοποιήσεων που θέλουμε να εκτελείται κατά το ξεκίνημα μιας εφαρμογής.

Αφού δημιουργηθεί μια λίστα, μπορούμε να επιλέξουμε ένα στοιχείο απ' αυτήν και για να ξέρουμε ποιο στοιχείο επιλέχθηκε, χρησιμοποιούμε την ιδιότητα ***ItemIndex*** του πλαισίου λίστας, που επιστρέφει ακέραια τιμή, ξεκινώντας από το 0 :

```
var
    CurrentItem : Integer;
begin
    CurrentItem := ListBox1.ItemIndex;
end;
```

Για να δούμε το κείμενο της επιλογής που κάναμε στη λίστα, μπορούμε να χρησιμοποιήσουμε την τιμή της ιδιότητας ItemIndex σαν δείκτη στην ιδιότητα Items :

```
ShowMessage(ListBox1.Items[ListBox1.ItemIndex]);
```

Άλλες χρήσιμες ιδιότητες ενός πλαισίου λίστας είναι η ***Columns***, στην οποία αν δώσουμε τιμή διάφορη από το 0, τα στοιχεία στο πλαίσιο λίστας θα εμφανίζονται σε πολλαπλές στήλες, η ***MultiSelect***, η οποία αν είναι *True* θα μπορούμε να κάνουμε πολλαπλή επιλογή στοιχείων στη λίστα με τα πλήκτρα *Shift* (συνεχόμενη πολλαπλή επιλογή) και *Control* (μη συνεχόμενη πολλαπλή επιλογή) και η ***Sorted***, η οποία αν είναι *True* η λίστα θα εμφανίζεται ταξινομημένη.

3.8.1 Τα σύνθετα πλαίσια

Το ***σύνθετο πλαίσιο*** ή ***πλαίσιο συνδυασμού*** *(combobox)* εμφανίζει μια λίστα επιλογών όπως και ένα πλαίσιο λίστας, αλλά έχει επιπλέον και ένα πλαίσιο κειμένου, όπου μπορούμε να κάνουμε καταχωρήσεις και είναι πτυσσόμενο, άρα εξοικονομεί χώρο στη φόρμα.

Τα σύνθετα πλαίσια έχουν κοινές με τα πλαίσια λίστας τις περισσότερες ιδιότητες, μεθόδους και συμβάντα και έχουν επιπλέον την ιδιότητα ***Text***, η οποία μας δίνει την τιμή που πληκτρολογήσαμε στο πλαίσιο κειμένου του σύνθετου πλαισίου ή την τιμή που επιλέξαμε από τη λίστα.

3.8.2 Τα πλαίσια ελέγχου

Το ***πλαίσιο ελέγχου*** *(checkbox)* είναι ένα τετράγωνο πλαίσιο, το οποίο όταν είναι επιλεγμένο εμφανίζει ένα σημάδι ελέγχου (Ö) μέσα του, ενώ όταν δεν είναι επιλεγμένο είναι κενό.

Σε μια ομάδα πλαισίων ελέγχου, μπορεί να είναι επιλεγμένα όλα, μερικά ή και κανένα. Το πλαίσιο ελέγχου έχει την ιδιότητα ***Caption***, που καθορίζει το μήνυμα (ετικέτα) που εμφανίζεται δίπλα του

Για να διαπιστώσουμε αν ένα πλαίσιο ελέγχου είναι επιλεγμένο, ελέγχουμε την ιδιότητά του ***Checked*** ως εξής :

```
If CheckBox1.Checked Then
        ShowMessage('Είναι επιλεγμένο')
Else
        ShowMessage('Δεν είναι επιλεγμένο');
```

3.9 ΤΑ ΠΛΗΚΤΡΑ ΕΠΙΛΟΓΗΣ

Το ***πλήκτρο επιλογής*** *(radiobutton* ή *optionbutton)* έχει παρόμοια λειτουργία με το πλαίσιο ελέγχου, αλλά με τις διαφορές ότι έχει κυκλικό σχήμα, όταν είναι επιλεγμένο εμφανίζει μια μαύρη βούλα (·) μέσα του και ακόμη ένα και μόνο ένα πλήκτρο επιλογής μπορεί να είναι επιλεγμένο από μια ομάδα πλήκτρων επιλογής. Τα πλήκτρα επιλογής είναι χρήσιμα για την εμφάνιση αμοιβαία αποκλειόμενων επιλογών στους χρήστες.

Το πλήκτρο επιλογής έχει την ιδιότητα ***Caption***, που καθορίζει το μήνυμα (ετικέτα) που εμφανίζεται δίπλα του και την ιδιότητα ***Checked***, που μπορεί να πάρει τιμή True ή False, όπως και στα πλαίσια ελέγχου, και καθορίζει το αν είναι επιλεγμένο ή όχι.

3.9.1 Η ομαδοποίηση των πλήκτρων επιλογής

Αν τοποθετήσουμε πλήκτρα επιλογής κατευθείαν σε μια φόρμα, αυτά θα είναι μεταξύ τους αμοιβαία αποκλειόμενα. Για να δημιουργήσουμε ομάδες πλήκτρων επιλογής, που να είναι μεταξύ

τους ανεξάρτητες, μπορούμε να χρησιμοποιήσουμε το πλήκτρο ***GroupBox*** ή το πλήκτρο ***Panel*** της καρτέλας Standard.

Σχεδιάζουμε πρώτα μέσα στη φόρμα ένα πλαίσιο ομάδας μ' ένα από τα δύο προηγούμενα πλήκτρα και μετά τοποθετούμε μέσα στο πλαίσιο αυτό όσα πλήκτρα επιλογής θέλουμε. Τα πλήκτρα επιλογής είναι «συνδεδεμένα» με το πλαίσιο, καθώς μετακινούνται μαζί του και είναι μεταξύ τους αμοιβαία αποκλειόμενα. Αν δημιουργήσουμε ένα άλλο πλαίσιο ομάδας, τα πλήκτρα επιλογής του δεν θα έχουν καμία σχέση με τα πλήκτρα επιλογής του πρώτου πλαισίου ομάδας.

Το GroupBox και το Panel μπορούν να χρησιμοποιηθούν και για την ομαδοποίηση πλαισίων ελέγχου ή και συνδυασμού πλαισίων ελέγχου με πλήκτρα επιλογής.

Υπάρχει, όμως, άλλο ένα εξάρτημα, αποκλειστικά για την ομαδοποίηση πλήκτρων επιλογής, το ***RadioGroup***. Για να εισάγουμε πλήκτρα επιλογής σ' ένα εξάρτημα RadioGroup, πρώτα σχεδιάζουμε ένα πλαίσιο μ' αυτό και μετά πάμε στην ιδιότητά του ***Items*** στον Object Inspector, όπου κάνουμε διπλό κλικ και στο παράθυρο *String list editor* που θα εμφανιστεί, γράφουμε τους τίτλους (ετικέτες) των πλήκτρων επιλογής που θέλουμε να δημιουργήσουμε μέσα στο RadioGroup.

3.10 ΕΝΕΡΓΟΠΟΙΗΣΗ ΚΑΙ ΑΠΟΚΡΥΨΗ ΕΞΑΡΤΗΜΑΤΩΝ

Τα εξαρτήματα είναι συνήθως διαθέσιμα (enabled) όταν τα τοποθετούμε σε μια φόρμα. Υπάρχουν, όμως, περιπτώσεις κατά την εκτέλεση μιας εφαρμογής, που θέλουμε ορισμένα εξαρτήματα να εμφανίζονται γκρίζα, ώστε να μην μπορούν να επιλεγούν.

Για να κάνουμε ένα εξάρτημα μη διαθέσιμο, θέτουμε στην ιδιότητά του ***Enabled*** την τιμή False :

Button1.Enabled := False;

και για να το κάνουμε διαθέσιμο, θέτουμε στην ιδιότητά του Enabled την τιμή True :

btnAdd.Enabled := True;

Παρόμοια, υπάρχουν περιπτώσεις κατά την εκτέλεση μιας εφαρμογής, που θέλουμε ορισμένα εξαρτήματα να μην εμφανίζονται καθόλου, δηλ. να αποκρύπτονται μέχρι να ικανοποιηθεί κάποια συνθήκη.

Για να κρύψουμε ένα εξάρτημα, θέτουμε στην ιδιότητά του ***Visible*** την τιμή False :

btnPrint.Visible := False;

και για να το εμφανίσουμε, θέτουμε στην ιδιότητά του Visible την τιμή True :

btnPrint.Visible := True;

3.11 ΕΝΑ ΠΡΟΓΡΑΜΜΑ ΠΡΟΒΟΛΗΣ ΕΙΚΟΝΩΝ

Θα κάνουμε μια εφαρμογή που θα εμφανίζει εικόνες σε μια φόρμα. Δημιουργούμε μια καινούργια εφαρμογή και ρυθμίζουμε την ιδιότητα ***Position*** της φόρμας σε *proScreenCenter* για να εμφανίζεται κεντραρισμένη κατά την εκτέλεση.

Από την καρτέλα Additional της γραμμής εργαλείων, προσθέτουμε δύο πλήκτρα εντολής BitBin και ένα πλήκτρο εικόνας (Image). Ρυθμίζουμε την ιδιότητα ***Stretch*** του στοιχείου Image σε *True*, ώστε η εικόνα που θα φορτωθεί να προσαρμοστεί στο μέγεθος του στοιχείου και για το ένα πλήκτρο BitBin επιλέγουμε την τιμή bkOK στην ιδιότητα Kind και για το άλλο την τιμή bkClose.

3.12 ΑΠΟΘΗΚΕΥΣΗ ΕΦΑΡΜΟΓΗΣ

Για κάθε έργο (*project*) η Delphi δημιουργεί διάφορα αρχεία. Το προεπιλεγμένο όνομα ενός έργου της Delphi είναι project1.dpr. Η προέκταση .dpr επιτρέπει στη Delphi να αναγνωρίζει ένα αρχείο ως έργο. Το όνομα πρέπει να είναι ενδεικτικό των περιεχομένων του αρχείου. Κάθε μονάδα (*unit*) ενός έργου περιέχει κώδικα Object Pascal το προεπιλεγμένο όνομα της πρώτης μονάδας είναι unit1.pas. Κάθε μονάδα πρέπει να αποθηκεύεται με μοναδικό όνομα. Σε κάθε φόρμα (*form*) αντιστοιχεί μια μονάδα. Το αρχείο φόρμας αποθηκεύεται με όνομα ίδιο με της αντίστοιχης μονάδας, αλλά με προέκταση .dfm (*Delphi Form – Φόρμα της Delphi*).

Στην άνω αριστερή γωνία στο μενού File υπάρχει η λειτουργία save (αποθήκευση). Με αυτόν τον τρόπο μπορούμε να επιλέξουμε τα εξής :*save, save as, save project as η save all*. Για να αποθηκεύσετε το αρχείο προσκήνιου χρησιμοποιήστε την εντολή *save*, για να αποθηκεύσετε το αρχείο προσκήνιου με καινούργιο όνομα χρησιμοποιήσετε την εντολή *save as*, για να αποθηκεύσετε το έργο κάτω από διαφορετικό όνομα χρησιμοποιήστε την εντολή *save project as* και τέλος για να αποθηκεύσετε οποιαδήποτε αρχείο που υπάρχει μέσα στο έργο χρησιμοποιήστε την εντολή *save all*.

4 Η ΥΛΟΠΟΙΗΣΗ ΤΗΣ ΕΡΓΑΣΙΑΣ ΜΕΣΩ ΤΗΣ ΓΛΩΣΣΑΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ DELPHI 7.0

4.1 ΑΝΤΙΚΕΙΜΕΝΟ ΤΟΥ ΚΕΦΑΛΑΙΟΥ

Στο συγκεκριμένο κεφάλαιο, θα αναλύσουμε την υλοποίηση της εφαρμογής μέσω του Delphi με την χρήση την Microsoft Access. Περιγράφετε, η δημιουργία ενός ψευδώνυμου ODBC με το οποίο θα συνδεθεί η βάση δεδομένων μας με το πρόγραμμα που έχουμε δημιουργήσει στο Delphi

Θα αναλύσουμε τις εικόνες και τα βασικά τμήματα των λιστών ,τα στοιχεία του κώδικα, τις λέξεις κλειδιά, τις ιδιότητες, τις κλάσεις, και τις συναρτήσεις.

Τέλος, θα παρουσιάσουμε τις φόρμες που δημιουργήσαμε στην εργασία μας, καθώς επίσης και τον κώδικα προγραμματισμού των φορμών για να μας δώσουν το τελικό πρόγραμμα στο σύνολο του.

4.2 OPEN DATABASE CONNECTIVITY (ODBC)

Από τα μέσα της δεκαετίας του '80, οι προγραμματιστές βάσεων δεδομένων, άρχισαν να ψάχνουν τη λύση ανεξαρτησίας των βάσεων δεδομένων. Η ιδέα είναι να χρησιμοποιούν ένα API που να μπορούν να χρησιμοποιούν οι εφαρμογές για να αλληλεπιδρούν με πολλές διαφορετικές προελεύσεις δεδομένων. Η χρήση ενός τέτοιου API θα απελευθέρωνε τους προγραμματιστές από την εξάρτηση από μια μόνο μηχανή βάσης δεδομένων και θα τους επέτρεπε να προσαρμόζονται στις διάφορες απαιτήσεις. Η πιο αξιοσημείωτη λύση, που έχουν παράγει η προμηθευτές ήταν *Open Database Connectivity (ODBC)*

Ξεκίνησε ως εσωτερικό προϊόν της Microsoft, αλλά η ευκολία στον προγραμματισμό και την εγκατάσταση των οδηγών έφερε τη γρήγορη αποδοχή του. Έτσι, οι αρχικές προδιαγραφές μετατράπηκαν σε πρότυπο, καθώς υπάρχουν σήμερα πάνω από 170 οδηγοί ODBC για κάθε σχεσιακή βάση δεδομένων.

Το *ODBC* αποτελεί μία καλύτερη συσκευασία και φυσικά διαθέτει όλες τις ρουτίνες και τη δυνατότητα πρόσβασης σε σχεσιακές βάσεις δεδομένων. Το ODBC ήταν η πρώτη ουσιαστικά προσπάθεια για έναν ενιαίο τρόπο πρόσβασης με τοπικές και απομακρυσμένες βάσεις δεδομένων. Το όνομά του προέρχεται από τα αρχικά των λέξεων Open DataBase Connectivity και αποτελείται από έναν κοινό τρόπο επικοινωνίας με τις σχεσιακές μόνο βάσεις δεδομένων. Τα περισσότερα περιβάλλοντα ανάπτυξης εφαρμογών και οι γλώσσες προγραμματισμού επιτρέπουν την πρόσβαση σε πίνακες της Access.

Έτσι, διατηρώντας τα στοιχεία σας σε ένα αρχείο .mdb της Access, μπορείτε αφενός να επεμβαίνετε πιο εύκολα αλλάζοντας κάποια από αυτά μέσα από το περιβάλλον της Access και αφετέρου να τα βλέπετε από οποιαδήποτε γλώσσα, όπως το Delph.

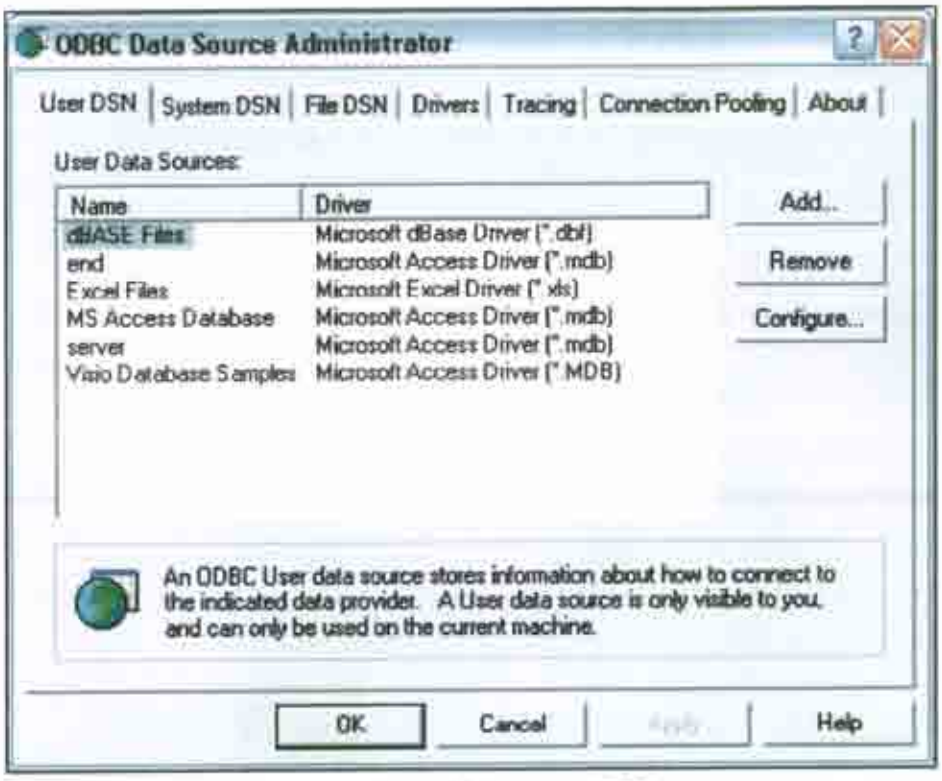
4.2.1 Περιγραφή και δημιουργία ενός ψευδωνύμου ODBC

Έχουμε σχεδιάσει μια Βάση δεδομένων , με την χρήση της Microsoft Access με όνομα αρχείου ***end.mdb***. Το παράδειγμα που θα ακολουθήσει θα δείξει την δημιουργία ενός ψευδωνύμου ODBC με το οποίο θα συνδεθεί η βάση δεδομένων μας με το πρόγραμμα που έχουμε δημιουργήσει στο Delphi.

Από το μενού Έναρξη κάνουμε κλικ στον *Πίνακα Ελέγχου* (Control Panel)

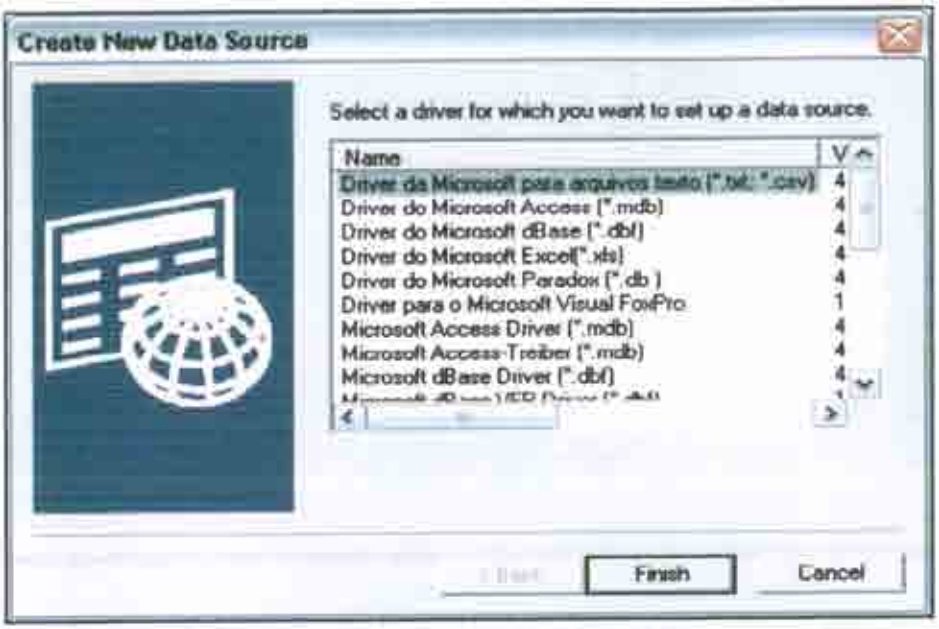
Επιλέγουμε το εικονίδιο *Administration Tools* και πατάμε στην επιλογή Data Sources (ODBC) οπού είναι ο κατάλογος στον οποίο έχουμε εγκαταστήσει τα Windows.

Στην οθόνη μας, εμφανίζεται το εξής εικονίδιο :



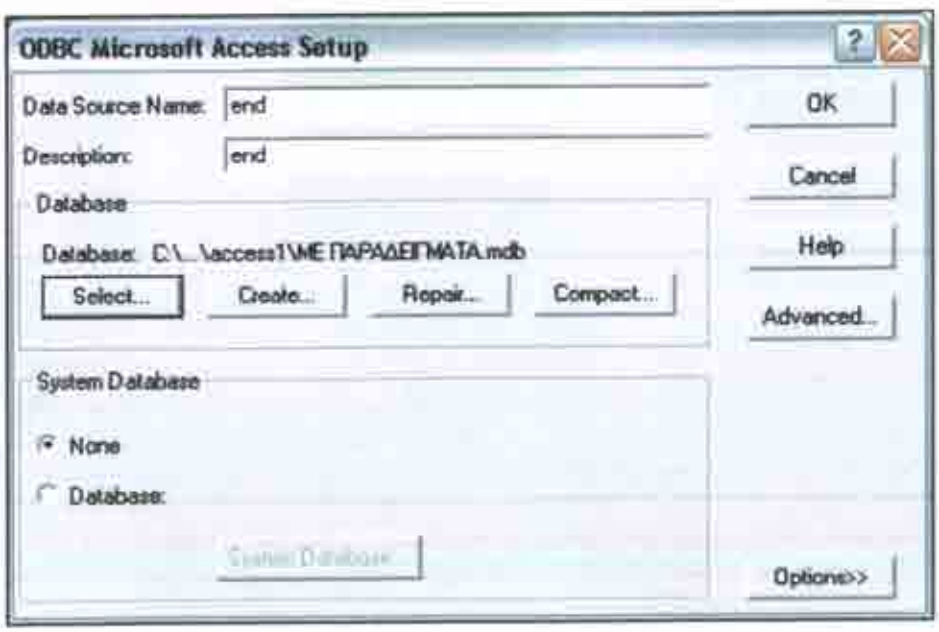
Εικόνα 3.2 Άνοιγμα του ODBC

Για να ξεκινήσουμε την δημιουργία ενός ψευδωνύμου κάνουμε κλικ στο κουμπί *Add* και εμφανίζετε το παράθυρο διάλογου Create New Data Source. Το παράθυρο διαλόγου Create New Data Source παραθέτει σε λίστα όλα τα προγράμματα οδήγησης (drivers) που έχουμε εγκαταστήσει. Βρίσκουμε και επιλέγουμε τη Microsoft Access Driver και κάνουμε κλικ στην εντολή *Finish*.



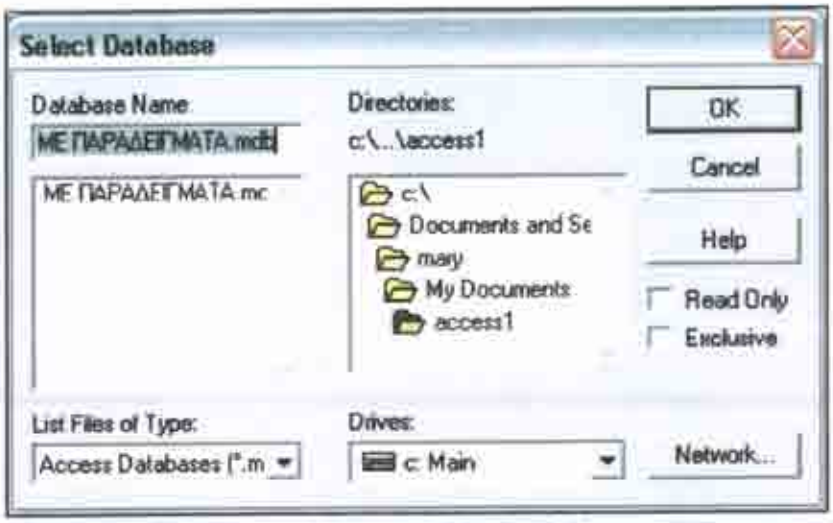
Εικόνα 3.3 Δημιουργία Νέας Πηγής Δεδομένων

Στην οθόνη μας εμφανίζεται το παράθυρο διαλόγου ODBC Microsoft Access Set Up. Δίνουμε όνομα και περιγραφή



Εικόνα 3.4 Επεξεργασία Πηγής Δεδομένων

και με την εντολή Select βρίσκουμε την Βάση Δεδομένων με την οποία θα δουλέψουμε. Τέλος, πατώντας την Εντολή Ok έχουμε δημιουργήσει ένα καινούριο User DSN με όνομα end.mdb.



Εικόνα 3.5 Δημιουργία Νέου User DSN

4.3 ΠΑΡΑΘΕΣΗ ΑΝΤΙΚΕΙΜΕΝΩΝ ΤΗΣ ΠΑΛΕΤΑΣ ΤΟΥ DELPHI ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΗΣΑΜΕ ΣΤΗΝ ΕΡΓΑΣΙΑ ΜΑΣ

Για να προσθέσουμε κάποιο από τα αντικείμενα της παλέτας του Delphi που θέλουμε να χρησιμοποιήσουμε επιλέγουμε το εικονίδιο και κάνοντας κλικ μέσα στην φόρμα μεταφέρεται αυτόματα.

Καρτέλα BDE :

Πίνακας *(Table)* : σύνολο δεδομένων (ίδιου τύπου) διατεταγμένο έτσι ώστε το κάθε στοιχείο του να μπορεί να προσδιοριστεί από το όνομα του συνόλου που το περιέχει και από τη θέση του η οποία καθορίζεται με την βοήθεια δεικτών, μέσα σε ένα σύνολο. Οι πίνακες μπορεί να είναι μιας ή περισσοτέρων διαστάσεων.

Καρτέλα DataAccess :

Πηγή Δεδομένων *(DataSourse)* : το συστατικό DataSource είναι μία κλάση. Η κλάση DataSource χρησιμοποιείται για την σύνδεσή στοιχείων ελέγχου (DataControls) στο DataSet. Το συστατικό DataSource δεν παρουσιάζει σημαντικό αριθμό δημοσίων ιδιοτήτων και μεθόδων αυτό που κάνει είναι να συγκρατεί αυτές τις δυνατότητες.

Καρτέλα Standard :

Κύριο Μενού *(MainMenu)* : σχεδιάζει μενού το οποίο εμφανίζεται αυτόματα στο πάνω μέρος της φόρμας το οποίο παρουσιάζει τις διάφορες λειτουργίες ή παραμέτρους που μπορούμε να επιλέξουμε καθώς και τον τρόπο επιλογής της κάθε μια από αυτές.

Ετικέτα *(Label)* : κάθε σειρά χαρακτήρων που προσδιορίζει μια εντολή στο οποίο γράφεται η ετικέτα χρησιμοποιείται από άλλα τμήματα του προγράμματος για την αναφορά τους στην συγκεκριμένη εντολή.

Κουμπί *(Button)* : δημιουργείτε ένα εικονίδιο στο οποίο μπορούμε να εφαρμόσουμε οποιαδήποτε εντολή – κώδικα επιθυμούμε.

Κουμπί ελέγχου *(DBCheckBox)* : χρησιμοποιείται για να εμφανίζει και να ενεργοποιεί – απενεργοποιεί μια επιλογή που αντιστοιχεί σε ένα λογικό πεδίο. Είναι μια περιορισμένη λίστα επειδή έχει δύο πιθανές τιμές, μαζί με την απροσδιόριστη κατάσταση για πεδία με κενές τιμές, προσδιορίζοντας ποιες είναι οι τιμές που θα σταλούν πίσω στην Βάση Δεδομένων.

Πλαίσιο *(Panel)* : αντικείμενο το οποίο μπορούμε να το χρησιμοποιήσουμε για την καλύτερη οπτική εικόνα της φόρμας.

Καρτέλα Additional :

Κουμπί *(BitBtn)* : δημιουργεί ένα κουμπί στην πρόσοψη του οποίου μπορεί να εμφανίζεται και ένα ψηφιογραφικό μαζί με τη λεζάντα του. Η Delphi διαθέτει μερικά έτοιμα τέτοια κουμπιά για την εκτέλεση συνηθισμένων ενεργειών όπως Cancel, Enter, Ok κ.τ.λ.

Καρτέλα DataControls :

Πεδίο *(DBEdit)* : Είναι ένα στοιχείο έλεγχου γενικής χρήσης για τις βάσεις δεδομένων. Επιτρέπει στον χρήστη να τροποποιήσει ένα πεδίο χρησιμοποιώντας ένα στοιχείο έλεγχου Edit.

Σύνολο κουμπιών *(DBNavigator)* : είναι ένα σύνολο από κουμπιά που χρησιμοποιείται για την πλοήγηση και την εκτέλεση ενεργειών στην βάση δεδομένων μπορούμε να απενεργοποιήσουμε μερικά από αυτά τα κουμπιά αφαιρώντας κάποια στοιχεία από το σύνολο VisibleButtons.

Πλέγμα Βάσης Δεδομένων *(DBGrid)* : είναι στοιχείο ελέγχου το οποίο συνδέεται με τον κλασικό πίνακα της Βάσης Δεδομένων.

Καρτέλα Win32 :

Ημερομηνία – Ώρα *(DateTimePicker)* : αντικείμενο το οποίο μπορούμε να χρησιμοποιήσουμε για την αυτόματη εμφάνιση ημερομηνίας και ώρας.

4.4 ΔΟΥΛΕΥΟΝΤΑΣ ΜΕ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

Για την προσπέλαση βάσεων δεδομένων, το Delphi χρησιμοποιεί το BDE (Borland Database Engine), που είναι ένας μηχανισμός που επιτρέπει στα προγράμματά μας να χρησιμοποιούν και να επεξεργάζονται δεδομένα διαφορετικών μορφών, όπως βάσεις δεδομένων σε dBase, Paradox, Access κ.ά.

Για να «συνδέσουμε» μια φόρμα μ' έναν πίνακα μιας βάσης δεδομένων, τοποθετούμε στη φόρμα τα εξαρτήματα *DataSource* και Table της καρτέλας Data Access, τα οποία φαίνονται κατά τη σχεδίαση της εφαρμογής, αλλά είναι αόρατα κατά την εκτέλεση. Στην ιδιότητα *DatabaseName* του εξαρτήματος ***Table*** επιλέγουμε τη βάση δεδομένων και στην ιδιότητα *TableName* του ίδιου εξαρτήματος επιλέγουμε τον πίνακα με τον οποίο θα δουλέψουμε.

Μετά, στην ιδιότητα *Dataset* του εξαρτήματος ***DataSource*** επιλέγουμε το όνομα του εξαρτήματος Table, δηλ. το Table1, ρυθμίζουμε την ιδιότητα *Active* του εξαρτήματος Table σε *True* και έτσι έχει επιτευχθεί η σύνδεση της φόρμας με τον πίνακα της βάσης δεδομένων.

Από την καρτέλα DataControls μπορούμε μετά να προσθέσουμε διάφορα εξαρτήματα στη φόρμα, όπως DBText, DBEdit, DBMemo, DBImage, DBListBox, DBComboBox και DBCheckBox. Για κάθε εξάρτημα, πρέπει να ρυθμίσουμε την ιδιότητά του *DataSource* στο όνομα του εξαρτήματος DataSource1 και την ιδιότητά του *DataField* στο όνομα του πεδίου του πίνακα με το οποίο θα συνδεθεί.

Έτσι, όταν θα εκτελεστεί η εφαρμογή, το κάθε εξάρτημα θα εμφανίζει τα περιεχόμενα κάποιου πεδίου του πίνακα και ότι αλλαγές κάνουμε στα περιεχόμενα των εξαρτημάτων αυτών, θα αποθηκευθούν και στα αντίστοιχα πεδία του πίνακα.

Δύο άλλα χρήσιμα εξαρτήματα είναι το ***DBGrid*** και το ***DBNavigator***. Σ' αυτά χρειάζεται να ρυθμίσουμε μόνο την ιδιότητά τους DataSource καθώς δεν έχουν ιδιότητα DataField. Το πρώτο εμφανίζει ένα πλέγμα με τα περιεχόμενα του πίνακα σε στήλες (πεδία) και γραμμές (εγγραφές) και το δεύτερο περιέχει διάφορα χρήσιμα πλήκτρα για να μπορούμε να μετακινούμαστε στις εγγραφές του πίνακα.

4.5 ΠΕΡΙΓΡΑΦΗ ΚΑΙ ΑΝΑΛΥΣΗ ΤΟΥ ΠΡΟΓΡΑΜΜΑΤΟΣ

Ήδη έχουμε δημιουργήσει μια Βάση Δεδομένων με όνομα αρχείου Εταιρεία.mdb. Έπειτα σχεδιάσαμε ένα ψευδώνυμο με το όνομα "end" για την σύνδεση της βάσης δεδομένων μας μέσω του ODBC driver, και του προγράμματος που έχει δημιουργηθεί στην γλώσσα προγραμματισμού DELPHI 7.0 με όνομα αρχείο ΕΤΑΙΡΕΙΑ.

Με βάση τα παραπάνω και έχοντας συνδέσει την Microsoft Access με το ODBC θα σχεδιάσουμε την διαδικασία δημιουργίας φορμών και θα καταλήξουμε στην υλοποίηση του προγράμματος με την χρήση του Delphi 7.0 Enterprise.

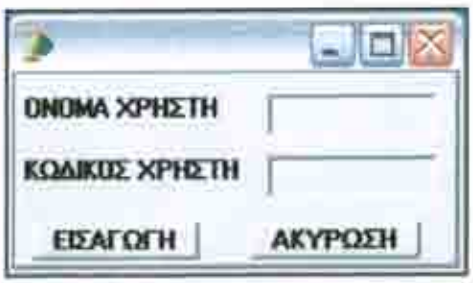
Ακολουθεί αναλυτικά η διαδικασία της χρήσης του προγράμματος :

Ανοίγουμε το πρόγραμμα Delphi και για να δημιουργήσουμε μια καινούργια φόρμα κάνουμε τα εξής :

Στην γραμμή μενού επιλέγουμε το αρχείο File ⟹ New ⟹ Form

Στην οθόνη μας εμφανίζεται μια κενή φόρμα με όνομα Form 1 στην οποία θα σχεδιάσουμε την φόρμα με την οποία οι υπάλληλοι θα μπορούν να έχουν πρόσβαση στο σύστημα.

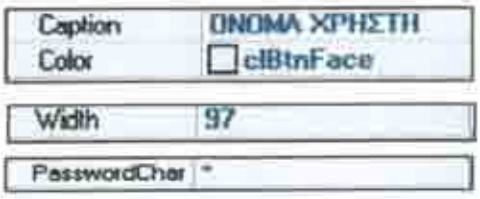
Έτσι λοιπόν η φόρμα μας είναι η εξής :



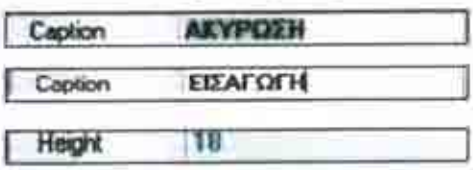
Για την δημιουργία της συγκεκριμένης φόρμας τα αντικείμενα τα οποία έχουμε χρησιμοποιήσει είναι τα εξής : Ετικέτα *(Label)*, Πλαίσιο Κειμένου *(Edit)*, Κουμπί *(Button)*, Πλαίσιο *(Panel)*.

Έχουμε εισάγει δύο Ετικέτες *(Label)* τα οποία και έχουμε ονομάσει Ετικέτα 1*(Label 1)* **Όνομα Χρήστη** και Ετικέτα 2 *(Label 2)* **Κωδικός Χρήστη**. Κάθε Label έχει ένα Πλαίσιο Κειμένου *Edit* που εισάγουμε το όνομα και τον κωδικό χρήστη αντίστοιχα.

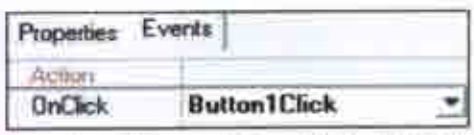
Παραθέτουμε τις ιδιότητες οι οποίες είναι ίδιες και για τις Ετικέτες *Label* και για τα Πλαίσια Κειμένου *Edit* εκτός από το Πλαίσιο Κειμένου 2 *Edit2* που εισάγαμε στην εντολή PasswordChar το σήμα * για να μην είναι ορατός ο κωδικός του κάθε χρήστη.



Για την πρόσβαση στο σύστημα έχουμε χρησιμοποιήσει δύο Κουμπιά (*Button*) όπου Button1 είναι η εισαγωγή και Button2 είναι η ακύρωση.



Στα Events έχουμε δημιουργήσει κάποιους κώδικές οι οποίοι λειτουργούν πατώντας κλικ με το ποντίκι.




```
procedure TPasswordForm.Button1Click(Sender: TObject);
var
i: Integer;
userfound: Boolean;
begin
userfound := False;
MainForm.Users.First;
//showmessage(IntToStr(MainForm.Users.RecordCount));

for i:=1 to MainForm.Users.RecordCount do
begin
//showmessage(inttostr(i));
if ((MainForm.Users.FieldByName('ONOMA ΧΡΗΣΤΗ').Value = Edit1.Text) and
(MainForm.Users.FieldByName('ΚΩΔΙΚΟΣ ΧΡΗΣΤΗ').Value = Edit2.Text)) then
begin
userfound := True;
break;
end;
MainForm.Users.Next;
end;

if userfound = True then
MainForm.Show
else
```

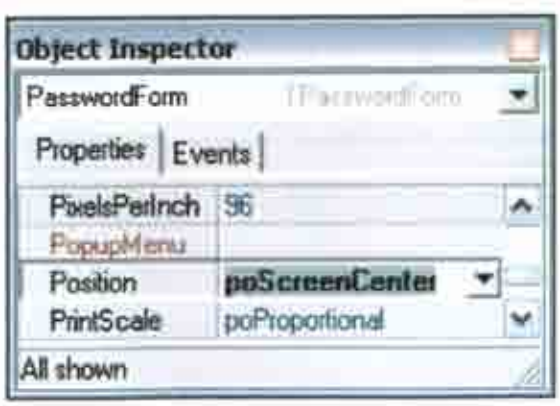
```
ShowMessage('ΛΑΘΟΣ ΣΤΟΙΧΕΙΑ ΧΡΗΣΤΗ');
if userfound = True then
PasswordForm.hide;
end;
procedure TPasswordForm.Button2Click(Sender: TObject);
begin
close;
end;
end.
```

Όλα τα παραπάνω αντικείμενα είναι επικολλημένα πάνω σε ένα Πλαίσιο *(Panel)* όπου τροποποιώντας την εντολή Align σε alClient επικαλύπτει όλη την φόρμα

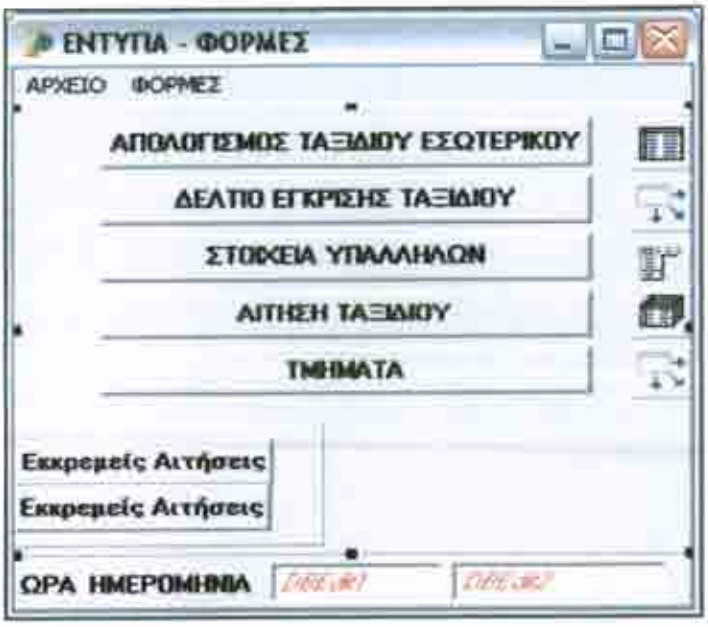
Με αυτή την ιδιότητα οι Ετικέτες *(Labels)* αποκτούν το χρώμα του Πλαισίου *(Panel)*

Επίσης για την καλύτερη εμφάνιση της φόρμας προσαρμόσαμε τις παρακάτω ιδιότητες έτσι ώστε η φόρμα μας να μπορεί να ελαχιστοποιείται και όχι να μεγιστοποιείται.

και στην εντολή Position επιλέξαμε την ιδιότητα proScreenCenter για να εμφανίζεται η φόρμα στο κέντρο.

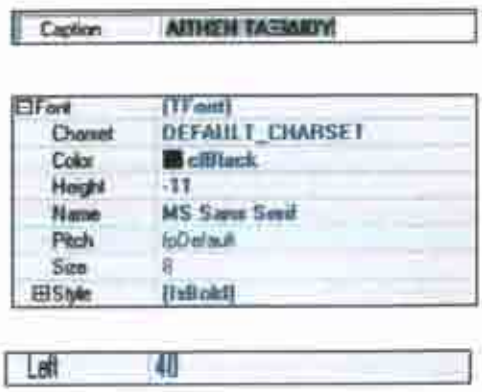


Μόλις κάνει εισαγωγή ο χρήστης στο σύστημα εμφανίζεται η κύρια φόρμα (MainForm).



Για την δημιουργία της, χρησιμοποιήσαμε τα εξής αντικείμενα : Κουμπί *(Button)*, Πλαίσιο *(Panel)*, Ετικέτα *(Label)*,Πηγή Δεδομένων *(DataSourse)*,Πίνακας *(Table)*, Στοιχείο Ελέγχου Κειμένου *(DBEdit)*,Κύριο Μενού *(MainMenu)*.

Όλα τα Κουμπιά *(Buttons)* τα οποία είναι 7, έχουν τις ίδιες ιδιότητες μεταξύ τους για αυτό αναφέρουμε ένα από αυτά.



Στα Events έχουμε δημιουργήσει κάποιους κώδικές οι οποίοι λειτουργούν πατώντας κλικ με το ποντίκι.


```
procedure TMainForm.Button1Click(Sender: TObject);
begin
AccountForm.show;
end;
procedure TMainForm.Button2Click(Sender: TObject);
begin
ApprovalForm.show;
End
procedure TMainForm.Button3Click(Sender: TObject);
begin
EmployeesForm.show;
end;
procedure TMainForm.Button4Click(Sender: TObject);
end;
begin
procedure TMainForm.Button5Click(Sender: TObject);
begin
DepartmentForm.show;
end;
procedure TMainForm.Button6Click(Sender: TObject);
begin
PendingApplicationForManager.Show;
end;
procedure TMainForm.Button7Click(Sender: TObject);
begin
PendingApplicationForAccount.show;
end;
```

Το  Πίνακας (table) έχει τις εξής ιδιότητες :

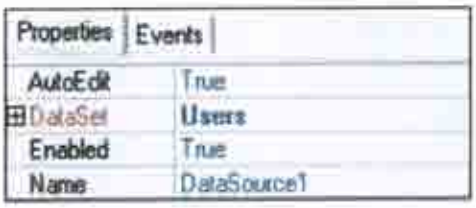
Συνδέουμε με την ιδιότητα database την βάση που έχουμε δημιουργήσει στο ODBC

Άμα ανοίξουμε το πτυσσόμενο κατάλογο θα εμφανιστούν όλοι οι πίνακες που έχουμε δημιουργήσει στην βάση δεδομένων μας. Εμείς επιλεγούμε το στοιχεία υπαλλήλων

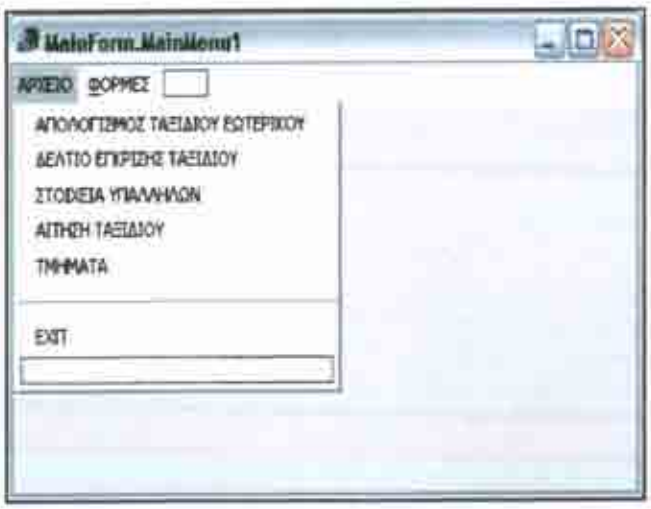
Δίνουμε όνομα

Και κάνουμε το Active σε true για να γίνει η σύνδεση

Η Πηγή Δεδομένων (DataSource)
έχει τις εξής ιδιότητες : όπου DataSet βάζουμε το όνομα του πίνακα Users



Το  Κύριο Μενού (MainMenu) το οποίο είναι το εξής σχεδιάστηκε στο πάνω μέρος της φόρμας. Αποτελείται από αυτόματα κουμπιά τα οποία αποκαλούνται N* και δώσαμε σε αυτά ξεχωριστές ονομασίες (Captions).

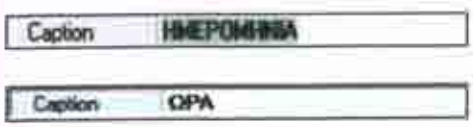


Στα Events έχουμε δημιουργήσει κάποιους κώδικες οι οποίοι λειτουργούν πατώντας κλικ με το ποντίκι.

```
procedure TMainForm.N2Click(Sender: TObject);

begin
Applicate.show;
end;

procedure TMainForm.N3Click(Sender: TObject);
begin
```

```
Account.Show;
end;

procedure TMainForm.N4Click(Sender: TObject);
begin
Approval.show;
end;

procedure TMainForm.N7Click(Sender: TObject);
begin
AccountForm.Show;
end;



procedure TMainForm.N8Click(Sender: TObject);
begin
ApprovalForm.Show;
end;

procedure TMainForm.N9Click(Sender: TObject);
begin
EmployeesForm.Show;
end;

procedure TMainForm.N10Click(Sender: TObject);
begin
ApplicationForm.Show;
end;

procedure TMainForm.N11Click(Sender: TObject);
begin
DepartmentForm.Show;
end;

procedure TMainForm.EXIT1Click(Sender: TObject);
begin
close;
end;
```

Χρησιμοποιήσαμε δύο Ετικέτες (Labels) τα οποία ονομάσαμε Ώρα και Ημερομηνία, δώσαμε στην εντολή Caption τις αντίστοιχες ονομασίες :



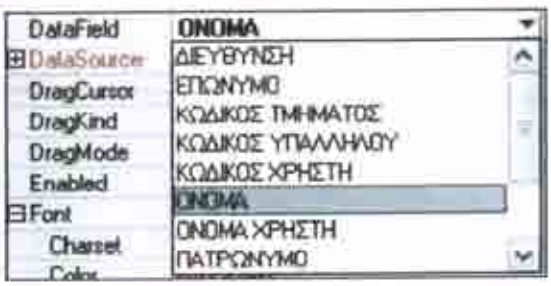
Στα Events έχουμε δημιουργήσει κάποιους κώδικες οι οποίοι λειτουργούν πατώντας κλικ με το ποντίκι.

```
procedure TMainForm.Label3Click(Sender: TObject);
begin
Label3.Caption := DateToStr(Date);
end;

procedure TMainForm.Label4Click(Sender: TObject);
```

```
begin
  Label4.Caption := TimeToStr(Time);
end;
```

Παραθέσαμε δύο  Πεδία (DBEdit) τα οποία η χρησιμότητα των οποίων είναι να εμφανίζει ποιός χρήστης έχει κάνει εισαγωγή στο σύστημα.

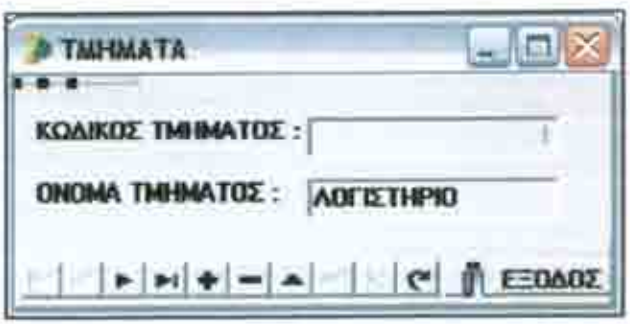
Επιλέγουμε σαν Πηγή δεδομένων την DataSourse1.



Στο πεδίο DataField εμφανίζονται όλα τα πεδία του πίνακα Users της βάσης δεδομένων που έχουμε δημιουργήσει στην Access όπου επιλέγουμε μόνο το όνομα και το επώνυμο.

Όλα τα παραπάνω αντικείμενα είναι επικολλημένα πάνω σε ένα Πλαίσιο (Panel) όπου τροποποιώντας την εντολή Align σε alClient επικαλύπτει όλη την φόρμα

Όπως κάθε εταιρεία αποτελείται από διάφορα τμήματα, έτσι και στην δικιά μας δημιουργήσαμε τα πέντε πιο απαραίτητα τμήματα για την καλύτερη υλοποίηση του έργου. Η φόρμα αυτή έχει ουσιαστικό ρόλο γιατί ανάλογα με τον κωδικό τμήματος που ανήκει στον κάθε υπάλληλο, έχει πρόσβαση και σε συγκεκριμένες φόρμες.



Για την σχεδίαση της παραπάνω φόρμας την οποία έχουμε ονομάσει DepartmentForm – Τμήματα χρησιμοποιήσαμε τα εξής αντικείμενα : Κουμπί *(Bitbin)*, Σύνολο κουμπιών *(DBNavigator)*, Κουμπί *(Button)*, Πλαίσιο *(Panel)*, Ετικέτα *(Label)*,Πηγή Δεδομένων *(DataSource)*,Πίνακας *(Table)*, Στοιχείο Ελέγχου Κειμένου *(DBEdit)*,Κύριο Μενού *(MainMenu)*.

Αρχικά πήραμε το απαραίτητο αντικείμενο Table το οποίο συνοδεύεται αυτόματα με ένα Datasource. Οι ιδιότητές που του προσαρμόσαμε είναι :

→ Συνδέσαμε την βάση μας

→ Δώσαμε όνομα

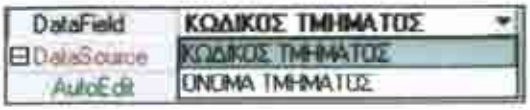
→ Συνδέσαμε τον πίνακα της Access Τμήματα με την ιδιότητα του Table TableName

→ Και με την ιδιότητα Active από False σε True έγινε η σύνδεση.

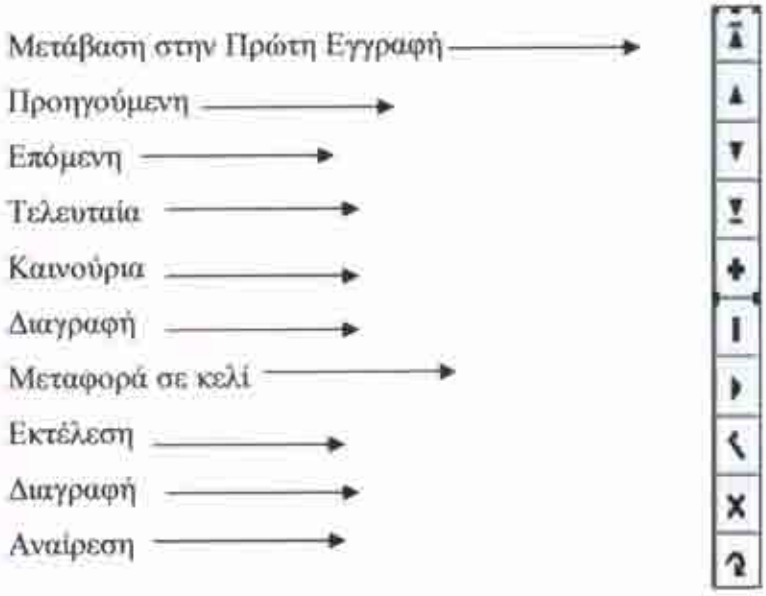
Και αυτόματα η ιδιότητα Dataset του Datasource συνδέθηκε με τις ιδιότητες του Table και έγινε "Departments"

Επίσης χρησιμοποιήσαμε δύο Ετικέτες (Label) τα οποία έχουν τις κλασικές ιδιότητες εκτός από την ιδιότητα Caption που έχουν ως εξής:

Η χρησιμότητα των Ετικετών (Labels) είναι να επεξηγούν το περιεχόμενο των DBEdit και σαν πηγή δεδομένων για τη συγκεκριμένη φόρμα στην ιδιότητα DataSource επιλέξαμε το DataSource1. Στην ιδιότητα DataField περάστηκαν αυτόματα τα στοιχεία των πεδίων της Access και τα επιλέγουμε για να γεμίσουν το περιεχόμενο του DBEdit.

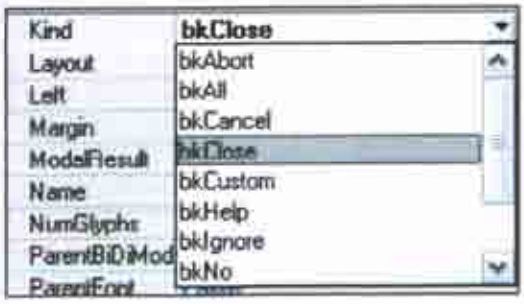


Τοποθετήσαμε στην φόρμα μας, το αντικείμενο Σύνολο Κουμπιών (DBNavigator), το οποίο είναι ένα σύνολο κουμπιών που βοηθάει στην καλύτερη πλοήγηση και την εκτέλεση ενεργειών.



Το συνδέσαμε με την πηγή δεδομένων μας DataSousce1 και τον πίνακα της φόρμας Departments έτσι ώστε να λειτουργεί η πλοήγηση των εγγραφών.

Τέλος τοποθετήσαμε το αντικείμενο Κουμπί (BitBin) το οποίο διαθέτει μερικά έτοιμα κουμπιά για την εκτέλεση συνηθισμένων ενεργειών. Εμείς επιλέξαμε το  κουμπί Close έτσι ώστε πατώντας κλικ επάνω του, να κλείνει η φόρμα.



Σε αυτή τη φόρμα όπως και στις επόμενες που ακολουθούν τοποθετούμε κάθε φορά δύο Πλαίσια (Panel). Όπου το ένα Πλαίσιο (Panel) περιέχει μόνο τα αντικείμενα Κουμπί (BitBin) και Σύνολο Κουμπιών DBNavigator και το άλλο περιέχει όλα τα υπόλοιπα.

Με τα στοιχεία της παραπάνω φόρμας δημιουργήσαμε τον εξής κώδικα στην MainForm έτσι ώστε ανάλογα με τον κωδικό τμήματος ο υπάλληλος να έχει πρόσβαση σε συγκεκριμένες φόρμες. Με βάση λοιπόν τις εντολές που δώσαμε τα κουμπιά στην κύρια φόρμα είναι διαθέσιμα ή μη ανάλογα με τον χρήστη που κάνει εισαγωγή στο σύστημα.

```
procedure TMainForm.FormShow(Sender: TObject);
begin
// Xristes logistiriou
if Users.FieldByName('ΚΩΔΙΚΟΣ ΤΜΗΜΑΤΟΣ').Value = 1 then
begin
Button1.Enabled := True;
Button2.Enabled := False;
Button3.Enabled := False;
Button4.Enabled := True;
Button5.Enabled := False;
Button7.Enabled := True;
Button6.Enabled := False;
end;

// Dieuthintis
If Users.FieldByName('ΚΩΔΙΚΟΣ ΤΜΗΜΑΤΟΣ').Value = 2 then
begin
Button1.Enabled := True;
Button2.Enabled := True;
Button3.Enabled := True;
Button4.Enabled := True;
Button5.Enabled := True;
Button7.Enabled := False;
Button6.Enabled := True;
end;
//Ypallhlos
if Users.FieldByName('ΚΩΔΙΚΟΣ ΤΜΗΜΑΤΟΣ).Value = 3 then
begin
Button1.Enabled := True;
Button2.Enabled := False;
Button3.Enabled := False;
Button4.Enabled := True;
Button5.Enabled := False;
Button7.Enabled := False;
Button6.Enabled := False;
end;

// Texniko proswpiko
if Users.FieldByName('ΚΩΔΙΚΟΣ ΤΜΗΜΑΤΟΣ').Value = 4 then
begin
Button1.Enabled := True;
Button2.Enabled := True;
Button3.Enabled := True;
Button4.Enabled := True;
Button5.Enabled := True;
Button7.Enabled := True;
Button6.Enabled := False;
end;

//logistirio
if Users.FieldByName('ΚΩΔΙΚΟΣ ΤΜΗΜΑΤΟΣ').Value = 1 then
```

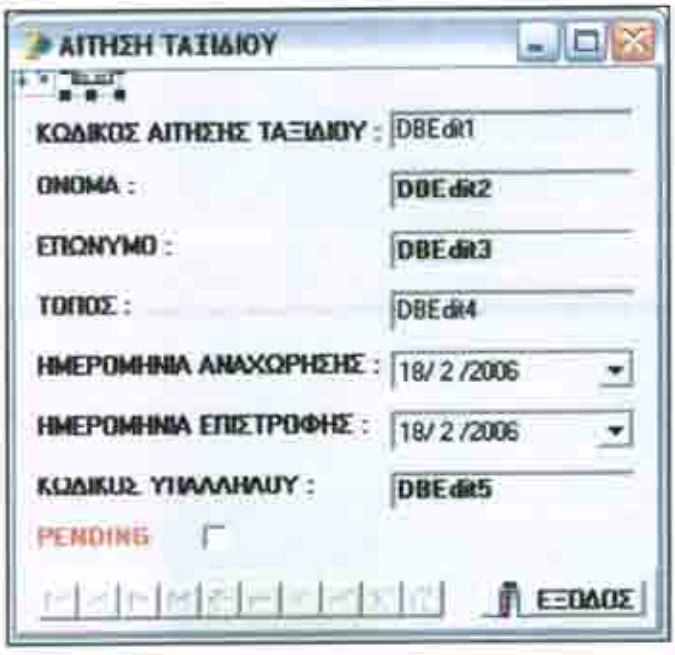
```
begin
N7.Enabled := True;
N8.Enabled := False;
N9.Enabled := False;
N10.Enabled := True;
N11.Enabled := False;
end;

//Dieythintis
if Users.FieldByName('ΚΩΔΙΚΟΣ ΤΜΗΜΑΤΟΣ').Value = 2 then
begin
N7.Enabled := True;
N8.Enabled := True;
N9.Enabled := True;
N10.Enabled := True;
N11.Enabled := True;
end;
//Upallhlos
if Users.FieldByName('ΚΩΔΙΚΟΣ ΤΜΗΜΑΤΟΣ').Value = 3 then
begin
N7.Enabled := True;
N8.Enabled := False;
N9.Enabled := False;
N10.Enabled := True;
N11.Enabled := False;
end;

//Texniko proswpiko
if Users.FieldByName('ΚΩΔΙΚΟΣ ΤΜΗΜΑΤΟΣ').Value = 4 then
begin
N7.Enabled := True;
N8.Enabled := True;
N9.Enabled := True;
N10.Enabled :=True;
N11.Enabled := True;
end;
end;
```

Συνεχίζουμε με την σχεδίαση της φόρμας που περιέχει τα στοιχεία των υπαλλήλων και την οποία έχουμε ονομάσει EmployeesForm.

Τα αντικείμενα τα οποία έχουμε χρησιμοποιήσει για την δημιουργία της είναι : Κουμπί *(Bitbin)*, Σύνολο κουμπιών *(DBNavigator)*, Κουμπί *(Button)*, Πλαίσιο *(Panel)*, Ετικέτα *(Label)*.Πηγή Δεδομένων *(DataSource)*,Πίνακας *(Table)*, Στοιχείο Ελέγχου Κειμένου *(DBEdit)*

Έτσι λοιπόν οι ιδιότητες που θα χρησιμοποιήσουμε για την φόρμα μας με όνομα αρχείου Στοιχεία Υπαλλήλων από τον object insector είναι ίδιες με αυτές που έχουμε αναλύσει παραπάνω.

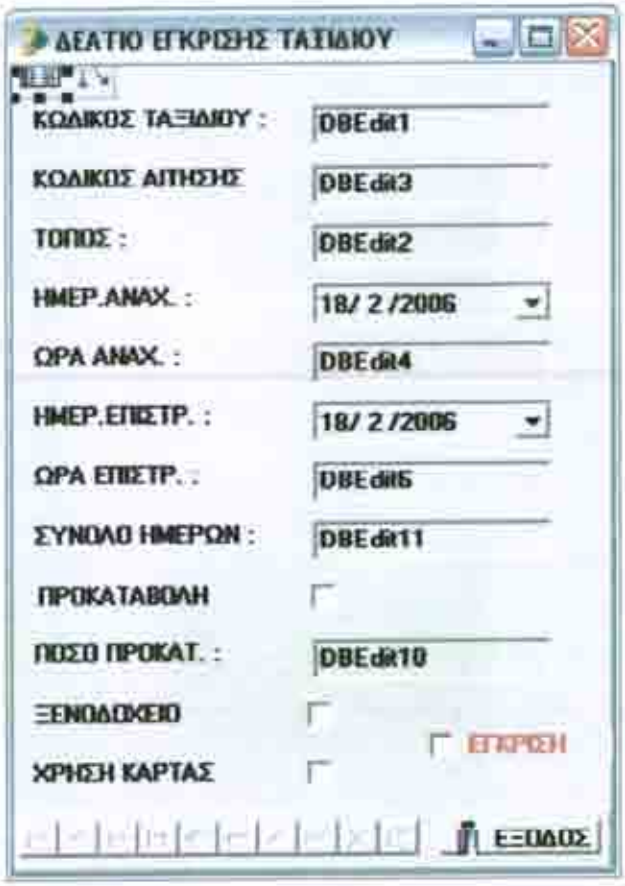
Η πιο σημαντική φόρμα στην εργασία μας είναι η φόρμα ApplicationForm (Αίτηση Ταξιδίου). Αυτή η φόρμα συμπληρώνεται από τον κάθε υπάλληλο όταν πρόκειται να πραγματοποιήσει κάποιο ταξίδι και είναι η ακόλουθη.



Τα αντικείμενα τα οποία έχουμε χρησιμοποιήσει για την δημιουργία της είναι : Κουμπί *(Bitbin)*, Σύνολο κουμπιών *(DBNavigator)*, Κουμπί *(Button)*, Πλαίσιο *(Panel)*, Ετικέτα *(Label)*,Πηγή Δεδομένων *(DataSource)*,Πίνακας *(Table)*, Στοιχείο Ελέγχου Κειμένου *(DBEdit)* Κουμπί Ελέγχου *(DBCheckBox)*, Ημερομηνία – Ώρα *(DateTimePicker)*.

Έτσι λοιπόν οι ιδιότητες που θα χρησιμοποιήσουμε για την φόρμα μας με όνομα αρχείου Αίτηση Ταξιδίου από τον object insector είναι ίδιες με αυτές που έχουμε αναλύσει παραπάνω. Οι ιδιότητες του αντικειμένου Κουμπί Ελέγχου (DBCheckBox) είναι ίδιες με τις ιδιότητες του αντικειμένου Στοιχείο Ελέγχου Κειμένου (DBEdit). Την Ημερομηνία – Ώρα (DateTimePicker) απλώς το τοποθετήσαμε στην φόρμα μας και δεχτήκαμε τις προκαθορισμένες ιδιότητες.

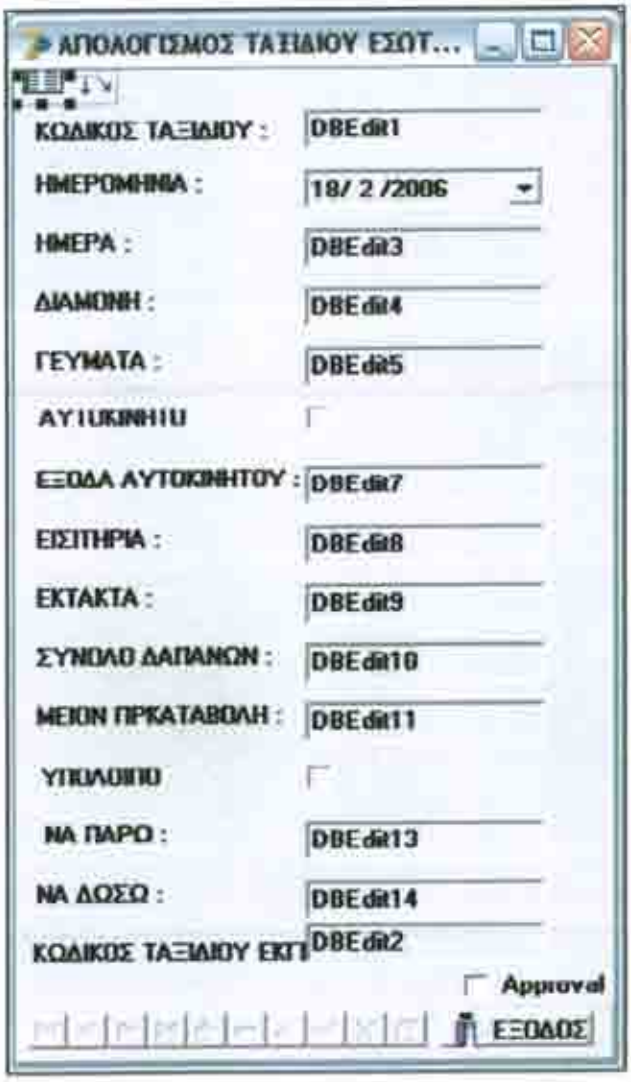
Η αμέσως επόμενη φόρμα είναι η ακόλουθη και την ονομάσαμε ApprovalForm (Δελτίο Έγκρισης Ταξιδίου). Στην συγκεκριμένη φόρμα έχει πρόσβαση μόνο ο διευθύντης και η χρησιμότητα της είναι να δηλώνει στον υπάλληλο αν το ταξίδι του έχει εγκριθεί ή όχι.



Τα αντικείμενα τα οποία έχουμε χρησιμοποιήσει για την δημιουργία της είναι : Κουμπί *(Bitbin)*, Σύνολο κουμπιών *(DBNavigator)*, Κουμπί *(Button)*, Πλαίσιο *(Panel)*, Ετικέτα *(Label)*,Πηγή Δεδομένων *(DataSource)*,Πίνακας *(Table)*, Στοιχείο Ελέγχου Κειμένου *(DBEdit)* Κουμπί Ελέγχου *(DBCheckBox)*.

Έτσι λοιπόν οι ιδιότητες που θα χρησιμοποιήσουμε για την φόρμα μας με όνομα αρχείου Δελτίο Έγκρισης Ταξιδίου από τον object insector είναι ίδιες με αυτές που έχουμε αναλύσει παραπάνω.

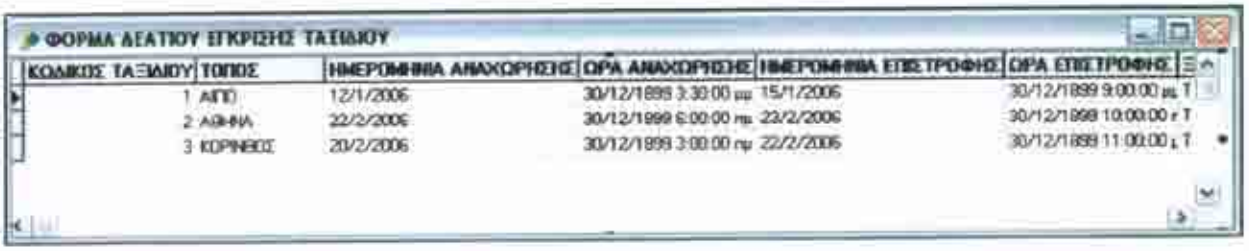
Η φόρμα που ακολουθεί ονομάζεται AccountForm (Απολογισμός Ταξιδίου Εσωτερικού) και περιέχει τα απαραίτητα στοιχεία που ο κάθε υπάλληλος είναι υποχρεωμένος να συμπληρώσει για να δηλώσει τα έξοδα εφόσον έχει πραγματοποιήσει ένα ταξίδι.



Τα αντικείμενα τα οποία έχουμε χρησιμοποιήσει για την δημιουργία της είναι : : Κουμπί *(Bitbin)*, Σύνολο κουμπιών *(DBNavigator)*, Κουμπί *(Button)*, Πλαίσιο *(Panel)*, Ετικέτα *(Label)*,Πηγή Δεδομένων *(DataSourse)*,Πίνακας *(Table)*, Στοιχείο Ελέγχου Κειμένου *(DBEdit)* Κουμπί Ελέγχου *(DBCheckBox)*.

Έτσι λοιπόν οι ιδιότητες που θα χρησιμοποιήσουμε για την φόρμα μας με όνομα αρχείου Απολογισμός Ταξιδίου από τον object insector είναι ίδιες με αυτές που έχουμε αναλύσει παραπάνω.

Έχουμε δημιουργήσει δύο επιπλέον φόρμες, μια για τον *Απολογισμό Ταξιδίου Εσωτερικού* και μια για το *Δελτίο Έγκρισης Ταξιδίου*, οι οποίες μας παρέχουν τα ίδια ακριβώς δεδομένα με τις αντίστοιχες φόρμες αλλά με μια διαφορετική εμφάνιση – δομή, δηλαδή τα στοιχεία των συγκεκριμένων φορμών εμφανίζονται σε πλέγμα βάσης δεδομένων.



Τα αντικείμενα, τα οποία έχουμε εισάγει για την σχεδίασή τους είναι : Πηγή δεδομένων (DataSource), Πινάκας (Table) και Πλέγμα Βάσης Δεδομένων (DBGrid).

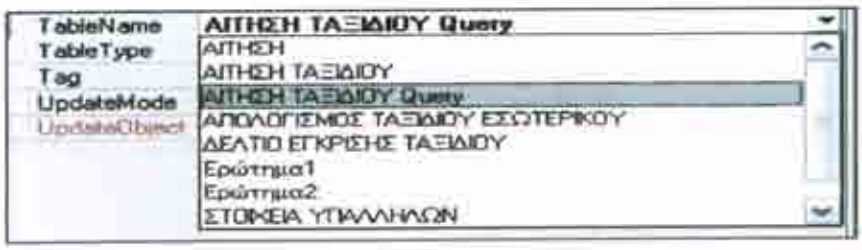
Οι ιδιότητες που προσαρμόσαμε στις φόρμες, έχουν ήδη αναλυθεί.

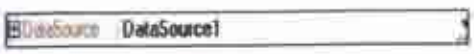
Μετά την δημιουργία των κλασικών φορμών (σύνδεση Delphi με πίνακα Access), σχεδιάσαμε ένα ερώτημα στην Access για το οποίο έχει γίνει αναφορά στο προηγούμενο κεφάλαιο. Όπως γνωρίζουμε κάθε ερώτημα έχει μία ιδιότητα SQL και λαμβάνει έναν έγκυρο κώδικα SQL. Η ιδιότητα και ο κώδικας που περάσαμε είναι να μας εμφανίζει τυχόν εκκρεμείς αιτήσεις οι οποίες δεν έχουν ελεγχθεί – εγκριθεί από το λογιστήριο και τον διευθυντή ξεχωριστά.

Αυτή η διαδικασία διεκπεραιώνεται όταν το λογιστήριο ή ο διευθυντής κάνει εισαγωγή στο σύστημα και ενημερώνεται εάν υπάρχουν εκκρεμείς αιτήσεις.

Η φόρμα που σχεδιάσαμε για να μας ενημερώνει για τις αιτήσεις που δεν έχουν εγκριθεί είναι οι εξής :

Η διαδικασία σύνδεσης με το ερώτημα που έχουμε δημιουργήσει στην Access είναι η ίδια όπως συνδέαμε τον πίνακα με το Delphi, αλλά στην ιδιότητα TableName του Table επιλέγουμε το ερώτημα που έχουμε δημιουργήσει στην Access όπως απεικονίζεται παρακάτω.





Στην ιδιότητα DataSource του DBGrid επιλέγουμε το στοιχείο DataSource1 για να γεμίσει το περιεχόμενό του με τα δεδομένα του ερωτήματος.

Στα Events του DBgrid περάσαμε τις παρακάτω εντολές έτσι ώστε πατώντας διπλό κλικ με το ποντίκι πάνω στο DBgrid να εμφανίζονται αναλυτικά τα στοιχεία των εκκρεμών αιτήσεων.


```
procedure TPendingApplicationForAccount.DBGrid1DblClick(Sender: TObject);
begin
AccountApplications.show;
end;
end.

procedure TPendingApplicationForManager.DBGrid1DblClick(Sender: TObject);
begin
ManagerApplications.show;
end;
end.
```

Τα αντικείμενα τα οποία έχουμε χρησιμοποιήσει για την δημιουργία των παραπάνω φορμών είναι : Σύνολο κουμπιών *(DBNavigator)*, Κουμπί *(Button)*, Πλαίσιο *(Panel)*, Ετικέτα *(Label)*,Πηγή Δεδομένων *(DataSource)*,Πίνακας *(Table)*, Στοιχείο Ελέγχου Κειμένου *(DBEdit)* Κουμπί Ελέγχου *(DBCheckBox)*.

Οι υπόλοιπες ιδιότητες που έχουμε δημιουργήσει για τις φόρμες μας με όνομα αρχείου Αιτήσεις προς διεκπεραίωση, Manager Application και Account Applications από τον object insector είναι ίδιες με αυτές που έχουμε αναλύσει.

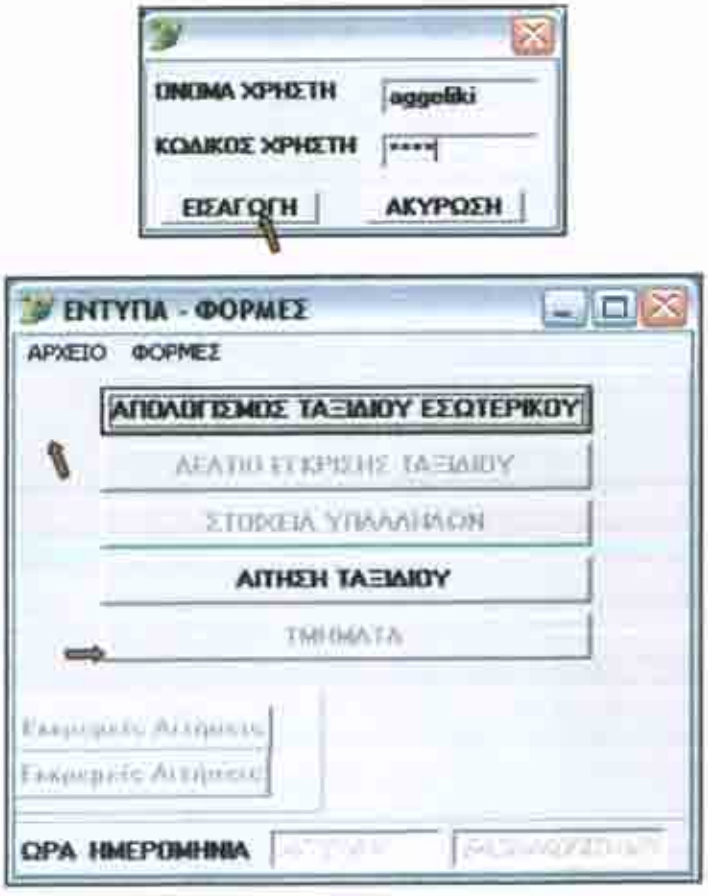
4.6 ΠΑΡΑΔΕΙΓΜΑ ΕΦΑΡΜΟΓΗΣ

Για να γίνει πιο κατανοητό το αντικείμενο της πτυχιακής μας εργασίας θα αναφέρουμε όσο πιο αναλυτικά γίνετε την διαδικασία ροής του προγράμματος.

Όπως έχουμε προαναφέρει η εταιρία αποτελείται από τέσσερα τμήματα : *Διευθυντής, Λογιστήριο, Υπάλληλοι, Τεχνικό Προσωπικό – Διαχειριστής.*

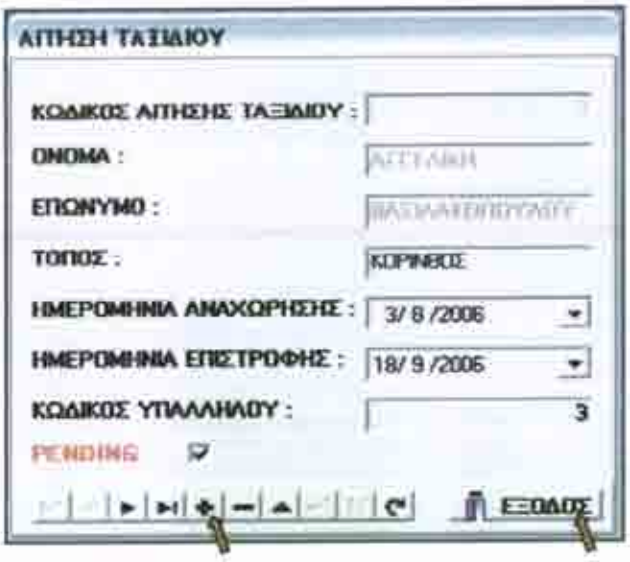
Στα πλαίσια της εταιρείας οι υπάλληλοι για διάφορους λόγους πραγματοποιούν κατά καιρούς διάφορα ταξίδια (είτε για να πραγματοποιήσουν κάποια σεμινάρια είτε για να επισκεφτούν κάποια από τα υποκαταστήματά τους). Τα έξοδα των ταξιδιών καλύπτονται από την εταιρεία εφόσον ο κάθε υπάλληλος προσκομίσει τα απαραίτητα δικαιολογητικά που δηλώνουν την ύπαρξη εξόδων π.χ. αποδείξεις διαμονής, οδοιπορικά κ.τ.λ.

Η διαδικασία που θα πρέπει να ακολουθήσει ένας υπάλληλος της εταιρείας η εξής :
Αρχικά ο υπάλληλος κάνει εισαγωγή στο σύστημα. Συμπληρώνει όνομα χρήστη και κωδικό χρήστη και πατώντας το κουμπί ΕΙΣΑΓΩΓΗ εμφανίζεται η Κεντρική Φόρμα (MainForm).



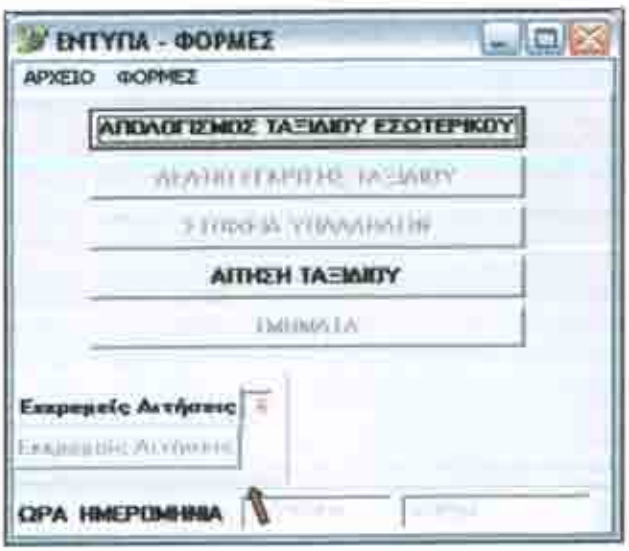
Όπως βλέπουμε ο υπάλληλος έχει πρόσβαση στη Αίτηση Ταξιδίου και στον Απολογισμό Ταξιδίου Εσωτερικού. Για να συμπληρώσει την αίτησή του έχει δύο επιλογές : είτε κάνει κλικ πάνω στο κουμπί με όνομα ΑΙΤΗΣΗ ΤΑΞΙΔΙΟΥ είτε από το μενού ΑΡΧΕΙΟ.

Μπαίνοντας λοιπόν στην συγκεκριμένη φόρμα συμπληρώνει πεδία της. Ο κωδικός αίτησης ταξιδίου το όνομα και το επώνυμο συμπληρώνονται αυτόματα από το σύστημα.

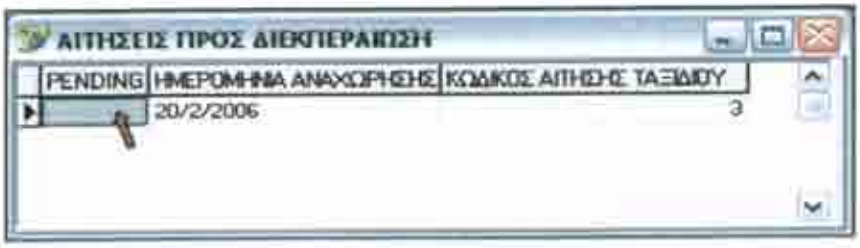


Πατώντας το πλήκτρο + αποθηκεύει την αίτησή του και με την επιλογή ΕΞΟΔΟΣ βγαίνει από το σύστημα.

Στην συνέχεια το λογιστήριο μόλις κάνει εισαγωγή στο σύστημα ακριβώς με τον ίδιο τρόπο που έκανε ο υπάλληλος, χρησιμοποιώντας το δικό του όνομα και κωδικό χρήστη, είναι υποχρεωμένος να ελέγχει τυχόν αιτήσεις που βρίσκονται σε εκκρεμότητα.



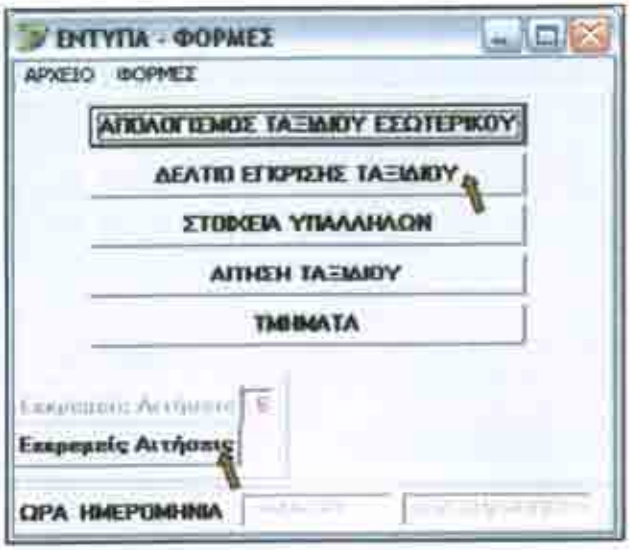
Πατώντας στο κουμπί με όνομα ΕΚΚΡΕΜΕΙΣ ΑΙΤΗΣΕΙΣ εμφανίζεται η φόρμα Αιτήσεις προς Διεκπεραίωση όπου απεικονίζονται οι αιτήσεις πάνω σε ένα πλέγμα Βάσεις Δεδομένων με τα απολύτως απαραίτητα στοιχεία.



Για να δει αναλυτικά τα στοιχεία της αίτησης κάνει διπλό κλικ πάνω στο πλέγμα και έτσι η φόρμα που εμφανίζεται είναι η εξής :

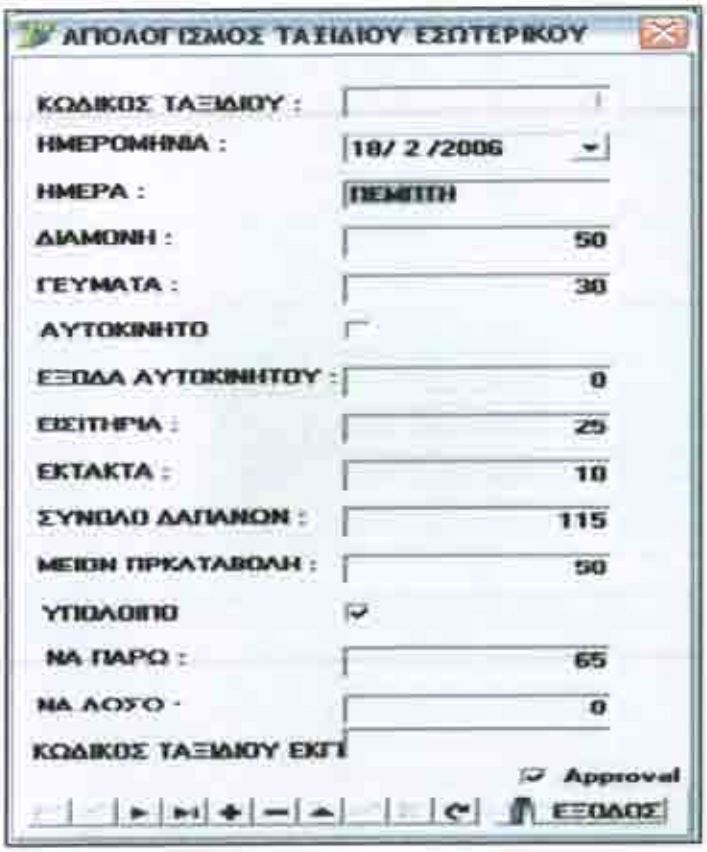
Εφόσον την ελέγξει με την επιλογή PENDING από True που δηλώνει ότι είναι εκκρεμείς την κάνει False και έτσι διαγράφεται από την λίστα των εκκρεμών αιτήσεων.

Ο διευθυντής με την σειρά του κάνει εισαγωγή στο σύστημα και με τον ίδιο τρόπο με το λογιστήριο ακολουθεί την διαδικασία ελέγχου των εκκρεμών αιτήσεων. Η μόνη διαφορά είναι ότι ο διευθυντής έχει πρόσβαση σε όλες τις φόρμες.



Επιλέγει την φόρμα ΔΕΛΤΙΟ ΕΓΚΡΙΣΗΣ ΤΑΞΙΔΙΟΥ και συμπληρώνει τα πεδία με τα στοιχεία του υπαλλήλου. Για να εγκρίνει την αίτηση κάνει την επιλογή ΕΓΚΡΙΣΗ από False σε True.

Όταν ο υπάλληλος επιστρέψει από το ταξίδι του είναι υποχρεωμένος να κάνει εισαγωγή στο σύστημα με τον τρόπο που έχουμε αναφέρει και να συμπληρώσει την φόρμα ΑΠΟΛΟΓΙΣΜΟΣ ΤΑΞΙΔΙΟΥ ΕΣΩΤΕΡΙΚΟΥ που περιέχει αναλυτικά τα έξοδα ταξιδίου. Ο υπάλληλος συμπληρώνει τα πεδία ανάλογα με τις δαπάνες που πραγματοποίησε προσκομίζοντας τα απαραίτητα δικαιολογητικά στο λογιστήριο για να του επιστραφούν τα χρήματα αφού εγκριθεί από τον διευθυντή.



5 ΕΠΙΛΟΓΟΣ

Στο σημείο αυτό ολοκληρώνεται η Σχεδίαση και Υλοποίηση Τυποποιημένης Διαδικασίας μιας Εταιρείας σε γλώσσα προγραμματισμού DELPHI 7.0 μέσω της βάσης δεδομένων ACCESS.

Με την εργασία αυτή καταφέραμε να δημιουργήσουμε με μία αναλυτική περιγραφή των φορμών που αναπτύχθηκαν μέσω της γλώσσας προγραμματισμού DELPHI 7.0 ένα κατανοητό πρόγραμμα Αιτήσεων Ταξιδίου για οποιαδήποτε εταιρεία έτσι ώστε να είναι προσιτό και σε έναν απλό χρήστη. Το συγκεκριμένο πρόγραμμα, έχει δημιουργηθεί και αναλυθεί πλήρως σε αυτό το βιβλίο.

Τελικά, η συγκεκριμένη πτυχιακή εργασία μπορεί να βρει εφαρμογή σε οποιαδήποτε Εταιρεία και προσαρμόζεται ανάλογα με τις ανάγκες της, αφού καταφέραμε και υλοποιήσαμε ένα αξιόπιστοπρόγραμμα.

ΒΙΒΛΙΟΓΡΑΦΙΑ

Θεωρία & πράξη
Συγγραφέας: Χρήστος Γουλτίδης
Εκδότης : "ΚΛΕΙΔΑΡΙΘΜΟΣ"

ΕΛΛΗΝΙΚΗ ACCESS 2000 ΣΤΗΝ ΕΚΠΑΙΔΕΥΣΗ
Εκδότης: ΓΚΙΟΥΡΔΑΣ Β.
Έτος Έκδοσης: 2001

Microsoft Access 2000 βήμα βήμα
Εκδότης: ΚΛΕΙΔΑΡΙΘΜΟΣ

ΟΔΗΓΟΣ ΤΗΣ ACCESS 2000
Συγγραφέας: CASSEL
Εκδότης: ΓΚΙΟΥΡΔΑΣ Μ.
Έτος Έκδοσης: 2000

Πλήρες Εγχειρίδιο Delphi 7
Συγγραφέας: Marco Cantu
Εκδότης : "Μ. Γκιούρδας"

DELPHI 7
Συγγραφέας: Cantu Marco
Εκδότης: Anaya Multimedia
Έτος έκδοσης: 30/09/2005

BORLAND DELPHI 6 DEVELOPER'S GUIDE
Συγγραφέας: Teixeira, Steve
Εκδότης: Pearson Professional Education
Έτος έκδοσης: 03/01/2002

Sites
www.borland.com/delphi
www.delphi.com
www.borland.it
www.prestwood.com/community/delphi
www.delphi.about.com/od/productreviews/a/bld7ann.htm –
www.borland.hu/delphi/pdf/del7_feaben.pdf
www.amazon.com/exec/obidos/tg/detail/-/0782143423?v=glance

ΚΩΔΙΚΕΣ ΤΩΝ ΦΟΡΜΩΝ ΠΟΥ ΕΧΟΥΝ ΣΧΕΔΙΑΣΤΕΙ ΣΕ DELPHI 7.0

Παρακάτω παραθέτουμε τους κώδικες και τα κουμπιά που χρησιμοποιήσαμε στο Borland Delphi 7.0 Enterprice σε κάθε unit και φόρμα ξεχωριστά.

Unit 7 PasswordForm

```
interface
uses
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, StdCtrls, DB, DBTables, Buttons, ExtCtrls;
type
TPasswordForm = class(TForm)
Panel1: TPanel;
Label1: TLabel;
Label2: TLabel;
Edit1: TEdit;
Edit2: TEdit;
Button1: TButton;
Button2: TButton;
procedure Button1Click(Sender: TObject);
procedure Button2Click(Sender: TObject);
private
{ Private declarations }
public
{ Public declarations-}
end;
var
PasswordForm: TPasswordForm;

implementation
```

```
uses Unit2, Unit1;

{$R *.dfm}

procedure TPasswordForm.Button1Click(Sender: TObject);
var
i: Integer;
userfound: Boolean;
begin
userfound := False;
MainForm.Users.First;
//showmessage(IntToStr(MainForm.Users.RecordCount));

for i:=1 to MainForm.Users.RecordCount do
begin
//showmessage(inttostr(i));
if ((MainForm.Users.FieldByName('ONOMA XPHΣTH').Value = Edit1.Text) and
(MainForm.Users.FieldByName('ΚΩΔΙΚΟΣ XPHΣTH').Value = Edit2.Text)) then
begin
userfound := True;
break;
end;
MainForm.Users.Next;
end;

if userfound = True then
MainForm.Show
else
ShowMessage('ΛΑΘΟΣ ΣΤΟΙΧΕΙΑ XPHΣTH');
if userfound = True then
PasswordForm.hide;
end;
procedure TPasswordForm.Button2Click(Sender: TObject);
```

```
begin
close;
end;
end.
```

Unti 1 MainForm

```
unit Unit1;
interface
uses
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, StdCtrls, DB, DBTables, Mask, DBCtrls, Grids, DBGrids, Menus,
Buttons, ExtCtrls;
type
TMainForm = class(TForm)
Button1: TButton;
Button2: TButton;
Button3: TButton;
Button4: TButton;
Button5: TButton;
Users: TTable;
UsersBDEDesigner: TIntegerField;
UsersBDEDesigner2: TStringField;
UsersBDEDesigner3: TStringField;
UsersBDEDesigner4: TStringField;
UsersBDEDesigner5: TStringField;
UsersBDEDesigner7: TStringField;
UsersBDEDesigner8: TIntegerField;
UsersBDEDesigner6: TFloatField;
DataSource1: TDataSource;
Panel1: TPanel;
DBEdit1: TDBEdit;
```

```
DBEdit2: TDBEdit;
MainMenu1: TMainMenu;
N1: TMenuItem;
N2: TMenuItem;
N3: TMenuItem;
N4: TMenuItem;
N5: TMenuItem;
N6: TMenuItem;
nnnn1: TMenuItem;
N7: TMenuItem;
N8: TMenuItem;
N9: TMenuItem;
N10: TMenuItem;
N11: TMenuItem;
N12: TMenuItem;
EXIT1: TMenuItem;
Database1: TDatabase;
UsersBDEDesigner9: TIntegerField;
UsersBDEDesigner10: TStringField;
Users_: TIntegerField;
UsersBDEDesigner11: TStringField;
Label3: TLabel;
Label4: TLabel;
Panel2: TPanel;
Panel3: TPanel;
Button7: TButton;
Button6: TButton;
procedure Button1Click(Sender: TObject);
procedure Button2Click(Sender: TObject);
procedure Button3Click(Sender: TObject);
procedure Button4Click(Sender: TObject);
procedure Button5Click(Sender: TObject);
procedure FormShow(Sender: TObject);
procedure Button6Click(Sender: TObject);
```

```
procedure N2Click(Sender: TObject);
procedure N3Click(Sender: TObject);
procedure N4Click(Sender: TObject);
procedure N7Click(Sender: TObject);
procedure N8Click(Sender: TObject);
procedure N9Click(Sender: TObject);
procedure N10Click(Sender: TObject);
procedure N11Click(Sender: TObject);
procedure EXIT1Click(Sender: TObject);
procedure Label3Click(Sender: TObject);
procedure Label4Click(Sender: TObject);
procedure Button7Click(Sender: TObject);

private
{ Private declarations }
public
{ Public declarations }
end;

var
MainForm: TMainForm;

implementation

uses Unit2, Unit3, Unit4, Unit5, Unit6, Unit7, Unit8, Unit9, Unit10,
Unit11, Unit13;

{$R *.dfm}

procedure TMainForm.Button1Click(Sender: TObject);
begin
AccountForm.show;
end;
```

```
procedure TMainForm.Button2Click(Sender: TObject);
begin
ApprovalForm.show;
end;

procedure TMainForm.Button3Click(Sender: TObject);
begin
EmployeesForm.show;
end;

procedure TMainForm.Button4Click(Sender: TObject);
begin;
end;

procedure TMainForm.Button5Click(Sender: TObject);
begin
DepartmentForm.show;
end;

procedure TMainForm.FormShow(Sender: TObject);
begin
// Xristes logistiriou
if Users.FieldByName('ΚΩΔΙΚΟΣ ΤΜΗΜΑΤΟΣ').Value = 1 then
begin
Button1.Enabled := True;
Button2.Enabled := False;
Button3.Enabled := False;
Button4.Enabled := True;
Button5.Enabled := False;
Button7.Enabled := True;
Button6.Enabled := False;
end;
```

```
// Dieuthintis
if Users.FieldByName('ΚΩΔΙΚΟΣ ΤΜΗΜΑΤΟΣ').Value = 2 then
begin
Button1.Enabled := True;
Button2.Enabled := True;
Button3.Enabled := True;
Button4.Enabled := True;
Button5.Enabled := True;
Button7.Enabled := False;

Button6.Enabled := True;
end;
//Ypallhlos
if Users.FieldByName('ΚΩΔΙΚΟΣ ΤΜΗΜΑΤΟΣ).Value = 3 then
begin
Button1.Enabled := True;
Button2.Enabled := False;
Button3.Enabled := False;
Button4.Enabled := True;
Button5.Enabled := False;
Button7.Enabled := False;

Button6.Enabled := False;
end;
// Texniko proswpiko
if Users.FieldByName('ΚΩΔΙΚΟΣ ΤΜΗΜΑΤΟΣ').Value = 4 then
begin
Button1.Enabled := True;
Button2.Enabled := True;
Button3.Enabled := True;
Button4.Enabled := True;
Button5.Enabled := True;
Button7.Enabled := True;
```

```
Button6.Enabled := False;
end;

//logistirio
if Users.FieldByName('ΚΩΔΙΚΟΣ ΤΜΗΜΑΤΟΣ').Value = 1 then
begin

N7.Enabled := True;
N8.Enabled := False;
N9.Enabled := False;
N10.Enabled := True;
N11.Enabled := False;
end;
//Dieythintis
if Users.FieldByName('ΚΩΔΙΚΟΣ ΤΜΗΜΑΤΟΣ').Value = 2 then
begin
N7.Enabled := True;
N8.Enabled := True;
N9.Enabled := True;
N10.Enabled := True;
N11.Enabled := True;
end;
//Upallhlos
if Users.FieldByName('ΚΩΔΙΚΟΣ ΤΜΗΜΑΤΟΣ').Value = 3 then
begin
N7.Enabled := True;
N8.Enabled := False;
N9.Enabled := False;
N10.Enabled := True;
N11.Enabled := False;
end;
//Texniko proswpiko
if Users.FieldByName('ΚΩΔΙΚΟΣ ΤΜΗΜΑΤΟΣ').Value = 4 then
begin
```

```
N7.Enabled := True;
N8.Enabled := True;
N9.Enabled := True;
N10.Enabled :=True;
N11.Enabled := True;
end;
end;

procedure TMainForm.Button6Click(Sender: TObject);
begin
PendingApplicationForManager.Show;
end;

procedure TMainForm.N2Click(Sender: TObject);
begin
Applicate.show;
end;

procedure TMainForm.N3Click(Sender: TObject);
begin
Account.Show;
end;

procedure TMainForm.N4Click(Sender: TObject);
begin
Approval.show;
end;

procedure TMainForm.N7Click(Sender: TObject);
begin
AccountForm.Show;
end;

procedure TMainForm.N8Click(Sender: TObject);
```

```
begin
ApprovalForm.Show;
end;

procedure TMainForm.N9Click(Sender: TObject);
begin
EmployeesForm.Show;
end;

procedure TMainForm.N10Click(Sender: TObject);
begin
ApplicationForm.Show;
end;

procedure TMainForm.N11Click(Sender: TObject);
begin
DepartmentForm.Show;
end;

procedure TMainForm.EXIT1Click(Sender: TObject);
begin
close;
end;

procedure TMainForm.Label3Click(Sender: TObject);
begin
Label3.Caption := DateToStr(Date);
end;

procedure TMainForm.Label4Click(Sender: TObject);
begin
Label4.Caption := TimeToStr(Time);
end;
```

```
procedure TMainForm.Button7Click(Sender: TObject);
begin
PendingApplicationForAccount.show;
end;

end.
```

Unit2 AccountForm

```
interface

uses
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, ExtCtrls, DBCtrls, StdCtrls, Mask, Grids, DBGrids, DB, DBTables,
Buttons, ComCtrls;

type
TAccountForm = class(TForm)
Label1: TLabel;
Panel1: TPanel;
Label3: TLabel;
Label4: TLabel;
Label5: TLabel;
Label6: TLabel;
Label8: TLabel;
Label9: TLabel;
Label10: TLabel;
Label11: TLabel;
Label12: TLabel;
Label14: TLabel;
Label15: TLabel;
DBEdit3: TDBEdit;
DBEdit4: TDBEdit;
```

DBEdit5: TDBEdit;

DBEdit7: TDBEdit;

DBEdit8: TDBEdit;

DBEdit9: TDBEdit;

DBEdit10: TDBEdit;

DBEdit11: TDBEdit;

DBEdit13: TDBEdit;

DBEdit14: TDBEdit;

Account: TTable;

DataSource1: TDataSource;

Panel2: TPanel;

DBNavigator1: TDBNavigator;

BitBtn1: TBitBtn;

DateTimePicker1: TDateTimePicker;

AccountBDEDesigner2: TDateTimeField;

AccountBDEDesigner3: TStringField;

AccountBDEDesigner4: TFloatField;

AccountBDEDesigner5: TFloatField;

AccountBDEDesigner6: TBooleanField;

AccountBDEDesigner7: TFloatField;

AccountBDEDesigner8: TFloatField;

AccountBDEDesigner9: TFloatField;

AccountBDEDesigner10: TFloatField;

AccountBDEDesigner11: TFloatField;

AccountBDEDesigner12: TBooleanField;

AccountBDEDesigner13: TFloatField;

AccountBDEDesigner14: TFloatField;

AccountBDEDesigner: TIntegerField;

Label2: TLabel;

DBEdit1: TDBEdit;

DBCheckBox1: TDBCheckBox;

DBCheckBox2: TDBCheckBox;

AccountBDEDesigner15: TIntegerField;

AccountApproval: TBooleanField;

```
Label7: TLabel;
DBEdit2: TDBEdit;
DBCheckBox3: TDBCheckBox;
private
{ Private declarations }
public
{ Public declarations }
end;

var
AccountForm: TAccountForm;

implementation

{$R *.dfm}

end.
```

Unit3 ApprovalForm

```
interface
uses
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, ExtCtrls, DBCtrls, StdCtrls, Mask, Grids, DBGrids, DB, DBTables,
Buttons, ComCtrls;
type
TApprovalForm = class(TForm)
Panel1: TPanel;
Label1: TLabel;
Label2: TLabel;
Label3: TLabel;
Label4: TLabel;
Label5: TLabel;
Label6: TLabel;
Label10: TLabel;
```

```
Label11: TLabel;
DBEdit1: TDBEdit;
DBEdit2: TDBEdit;
DBEdit4: TDBEdit;
DBEdit6: TDBEdit;
DBEdit10: TDBEdit;
DBEdit11: TDBEdit;
Approvals: TTable;
DataSource1: TDataSource;
Panel2: TPanel;
DBNavigator1: TDBNavigator;
BitBtn1: TBitBtn;
DateTimePicker1: TDateTimePicker;
DateTimePicker2: TDateTimePicker;
ApprovalsBDEDesigner: TIntegerField;
ApprovalsBDEDesigner2: TStringField;
ApprovalsBDEDesigner3: TDateTimeField;
ApprovalsBDEDesigner4: TDateTimeField;
ApprovalsBDEDesigner5: TDateTimeField;
ApprovalsBDEDesigner6: TDateTimeField;
ApprovalsBDEDesigner7: TBooleanField;
ApprovalsBDEDesigner8: TBooleanField;
ApprovalsBDEDesigner9: TBooleanField;
ApprovalsBDEDesigner10: TFloatField;
ApprovalsBDEDesigner11: TIntegerField;
ApprovalsBDEDesigner12: TIntegerField;
ApprovalsBDEDesigner13: TBooleanField;
DBCheckBox1: TDBCheckBox;
Label12: TLabel;
DBEdit3: TDBEdit;
DBCheckBox2: TDBCheckBox;
DBCheckBox3: TDBCheckBox;
DBCheckBox4: TDBCheckBox;
procedure DBNavigator1Click(Sender: TObject; Button: TNavigateBtn);
```

```
private
{ Private declarations }
public
{ Public declarations }
end;

var
ApprovalForm: TApprovalForm;

implementation

{$R *.dfm}

procedure TApprovalForm.DBNavigator1Click(Sender: TObject; Button: TNavigateBtn);
begin
ShowMessage('form3');
end;
end.
```

Unit4 EmployeesForm

```
interface
uses
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, StdCtrls, Mask, DBCtrls, Grids, DBGrids, DB, DBTables, ExtCtrls,
Buttons;
type
TEmployeesForm = class(TForm)
Employees: TTable;
DataSource1: TDataSource;
Panel1: TPanel;
```

Label1: TLabel;

DBEdit1: TDBEdit;

Label2: TLabel;

DBEdit2: TDBEdit;

Label3: TLabel;

DBEdit3: TDBEdit;

Label4: TLabel;

DBEdit4: TDBEdit;

Label5: TLabel;

DBEdit5: TDBEdit;

Label6: TLabel;

DBEdit6: TDBEdit;

Panel2: TPanel;

DBNavigator1: TDBNavigator;

BitBtn1: TBitBtn;

EmployeesBDEDesigner: TStringField;

Employees_: TIntegerField;

EmployeesBDEDesigner2: TStringField;

EmployeesBDEDesigner3: TIntegerField;

EmployeesBDEDesigner4: TStringField;

Label7: TLabel;

DBEdit7: TDBEdit;

Label8: TLabel;

DBEdit8: TDBEdit;

Label9: TLabel;

DBEdit9: TDBEdit;

Label10: TLabel;

DBEdit10: TDBEdit;

Label11: TLabel;

DBEdit11: TDBEdit;

EmployeesBDEDesigner6: TIntegerField;

EmployeesBDEDesigner7: TStringField;

EmployeesBDEDesigner8: TStringField;

EmployeesBDEDesigner9: TStringField;

```
EmployeesBDEDesigner10: TStringField;
EmployeesBDEDesigner11: TFloatField;
EmployeesBDEDesigner5: TIntegerField;
Label12: TLabel;
DBEdit12: TDBEdit;
private
{ Private declarations }
public
{ Public declarations }
end;
var
EmployeesForm: TEmployeesForm;

implementation

{$R *.dfm}
end.
```

Unit5 ApplicationForm

```
interface
uses
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, ExtCtrls, DBCtrls, StdCtrls, Mask, Grids, DBGrids, DB, DBTables,
Buttons, ComCtrls;
type
TApplicationForm = class(TForm)
Panel1: TPanel;
Label2: TLabel;
Label3: TLabel;
Label4: TLabel;
Label5: TLabel;
Label6: TLabel;
DBEdit4: TDBEdit;
Applications: TTable;
```

```
DataSource1: TDataSource;
Panel2: TPanel;
DBNavigator1: TDBNavigator;
BitBtn1: TBitBtn;
ApplicationsStringField: TStringField;
ApplicationsBDEDesigner2: TIntegerField;
ApplicationsBDEDesigner3: TStringField;
ApplicationsBDEDesigner4: TDateTimeField;
ApplicationsBDEDesigner5: TDateTimeField;
ApplicationsField: TStringField;
DateTimePicker1: TDateTimePicker;
DateTimePicker2: TDateTimePicker;
ApplicationsPENDING: TBooleanField;
DBCheckBox1: TDBCheckBox;
Label7: TLabel;
DBEdit5: TDBEdit;
ApplicationsBDEDesigner: TIntegerField;
Label1: TLabel;
DBEdit1: TDBEdit;
DBEdit2: TDBEdit;
DBEdit3: TDBEdit;
procedure ApplicationsAfterInsert(DataSet: TDataSet);
procedure ApplicationsBeforePost(DataSet: TDataSet);
private
{ Private declarations }
public
{ Public declarations }
end;
var
ApplicationForm: TApplicationForm;

implementation

uses Unit1;
```

```
{$R *.dfm}

procedure TApplicationForm.ApplicationsAfterInsert(DataSet: TDataSet);
begin
Applications.FieldByName('ÊÙÄÉÊÏÓ ÁÉÔÇÓÇÓ ÔÁÎÉÄÉÏÕ').Value := 30;
Applications.FieldByName('ÊÙÄÉÊÏÓ ÕÐÁËËÇËÏÕ').Value :=
MainForm.Users.FieldByName('ÊÙÄÉÊÏÓ ÕÐÁËËÇËÏÕ').Value;
end;
procedure TApplicationForm.ApplicationsBeforePost(DataSet: TDataSet);
begin
Applications.FieldByName('ÇÌÅÑÌÌÇÍÉÁ ÁÍÁ×ÙÑÇÓÇÓ').Value := DateTimePicker1.DateTime;
end;
end.
```

Unit6 DepartmentForm

```
interface
uses
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, ExtCtrls, DBCtrls, StdCtrls, Mask, Grids, DBGrids, DB, DBTables,
Buttons, Menus;
type
TDepartmentForm = class(TForm)
Departments: TTable;
DataSource1: TDataSource;
Label2: TLabel;
DBEdit2: TDBEdit;
Panel2: TPanel;
DBNavigator1: TDBNavigator;
BitBtn1: TBitBtn;
DepartmentsBDEDesigner: TStringField;
DepartmentsBDEDesigner2: TAutoIncField;
Label1: TLabel;
DBEdit1: TDBEdit;
private
```

```
{ Private declarations }
public
{ Public declarations }
end;

var
DepartmentForm: TDepartmentForm;

implementation

{$R *.dfm}
end.
```

Unit8

```
interface
uses
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, Grids, DBGrids, DB, DBTables, ExtCtrls;
type
TApplicate = class(TForm)
Panel1: TPanel;
Applications: TTable;
DataSource1: TDataSource;
DBGrid1: TDBGrid;
ApplicationsBDEDesigner: TIntegerField;
ApplicationsBDEDesigner2: TIntegerField;
ApplicationsBDEDesigner3: TStringField;
ApplicationsBDEDesigner4: TDateTimeField;
ApplicationsBDEDesigner5: TDateTimeField;
ApplicationsPENDING: TBooleanField;
private
{ Private declarations }
public
{ Public declarations }
```

```
end;

var
Applicate: TApplicate;

implementation

{$R *.dfm}
end.
```

Unit9

```
interface
uses
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, DB, DBTables, Grids, DBGrids, ExtCtrls;

type
TAccount = class(TForm)
Panel1: TPanel;
DBGrid1: TDBGrid;
DataSource1: TDataSource;
Account: TTable;
private
{ Private declarations }
public
{ Public declarations }
end;

var
Account: TAccount;

implementation

{$R *.dfm}
```

```
end.
```

Unit10 Approval

```
interface

uses
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, DB, DBTables, Grids, DBGrids, ExtCtrls;
type
TApproval = class(TForm)
Panel1: TPanel;
DBGrid1: TDBGrid;
DataSource1: TDataSource;
Approvals: TTable;
private
{ Private declarations }
public
{ Public declarations }
end;

var
Approval: TApproval;

implementation

{$R *.dfm}

end.
```

Unit11 PendingApplicationForAccount

```
interface

uses
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, DB, DBTables, Grids, DBGrids, StdCtrls, ExtCtrls;

type
TPendingApplicationForAccount = class(TForm)
Panel1: TPanel;
Table1: TTable;
DataSource1: TDataSource;
DBGrid1: TDBGrid;
procedure DBGrid1DblClick(Sender: TObject);
private
{ Private declarations }
public
{ Public declarations }
end;

var
PendingApplicationForAccount: TPendingApplicationForAccount;

implementation

uses Unit2, Unit5, Unit12;

{$R *.dfm}

procedure TPendingApplicationForAccount.DBGrid1DblClick(Sender: TObject);
begin
AccountApplications.show;
```

```
end;
end.
```

Unit12 AccountApplications

```
interface

uses
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, DBCtrls, StdCtrls, Mask, DB, DBTables, ExtCtrls;

type
TAccountApplications = class(TForm)
Panel1: TPanel;
Table1: TTable;
DataSource1: TDataSource;
Table1PENDING: TBooleanField;
Table1BDEDesigner: TStringField;
Table1BDEDesigner2: TStringField;
Table1BDEDesigner3: TStringField;
Table1BDEDesigner4: TDateTimeField;
Table1BDEDesigner5: TDateTimeField;
DBCheckBox1: TDBCheckBox;
DataSource2: TDataSource;
Label1: TLabel;
DBEdit1: TDBEdit;
Label2: TLabel;
DBEdit2: TDBEdit;
Label3: TLabel;
DBEdit3: TDBEdit;
Label4: TLabel;
DBEdit4: TDBEdit;
Label5: TLabel;
```

```
DBEdit5: TDBEdit;
DBNavigator1: TDBNavigator;
private
{ Private declarations }
public
{ Public declarations }
end;

var
AccountApplications: TAccountApplications;

implementation

{$R *.dfm}
end.
```

Unit13 PendingApplicationForManager

```
interface
uses
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, Grids, DBGrids, DB, DBTables, ExtCtrls;
type
TPendingApplicationForManager = class(TForm)
Panel1: TPanel;
Table1: TTable;
DataSource1: TDataSource;
DBGrid1: TDBGrid;
procedure DBGrid1DblClick(Sender: TObject);
private
{ Private declarations }
public
{ Public declarations }
```

```
end;

var
PendingApplicationForManager: TPendingApplicationForManager;

implementation
```

Unit14 PendingApplicationForManager

```
{$R *.dfm}

procedure TPendingApplicationForManager.DBGrid1DblClick(Sender: TObject);
begin
ManagerApplications.show;
end;

end.
unit Unit14;
interface
uses
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, DBCtrls, StdCtrls, Mask, DB, DBTables, ExtCtrls;
type
TManagerApplications = class(TForm)
Panel1: TPanel;
Table1: TTable;
DataSource1: TDataSource;
Table1BDEDesigner: TStringField;
Table1BDEDesigner2: TDateTimeField;
Table1BDEDesigner3: TDateTimeField;
Table1PENDING: TBooleanField;
Table1BDEDesigner4: TStringField;
Table1BDEDesigner5: TStringField;
Table1BDEDesigner6: TAutoIncField;
```

```
Label1: TLabel;
DBEdit1: TDBEdit;
Label2: TLabel;
DBEdit2: TDBEdit;
Label3: TLabel;
DBEdit3: TDBEdit;
DBCheckBox1: TDBCheckBox;
Label4: TLabel;
DBEdit4: TDBEdit;
Label5: TLabel;
DBEdit5: TDBEdit;
DBNavigator1: TDBNavigator;
private
{ Private declarations }
public
{ Public declarations }
end;

var
ManagerApplications: TManagerApplications;

implementation
{$R *.dfm}

end.
```