Κεφάλαιο 14

Λειτουργικές Μονάδες

Στο κεφάλαιο αυτό παρουσιάζεται η δημιουργία και χρήση λειτουργικών μονάδων δια της χρήσης των οποίων είναι δυνατή η πραγματοποίηση πολύπλοκων διαδικασιών που δεν μπορούν να υλοποιηθούν με τη βοήθεια των υπόλοιπων εργαλείων της Microsoft Access.

Μετά την αναλυτική περιγραφή των βασικών αντικειμένων μιας βάσης δεδομένων έτσι όπως αυτή υλοποιείται μέσα από το περιβάλλον της **Microsoft Access**, ας περάσουμε τώρα σε πιο προχωρημένες τεχνικές ανάπτυξης εφαρμογών οι οποίες εκτός των άλλων περιλαμβάνουν **τη συγγραφή κώδικα**, δια της χρήσης του οποίου είναι δυνατή η εκτέλεση πολύπλοκων διαδικασιών. Σε όλες τις εκδόσεις της **Microsoft Access**, η διαθέσιμη γλώσσα προγραμματισμού για την συγγραφή αυτού του κώδικα, είναι μια περιορισμένη έκδοση της **Visual Basic** που είναι γνωστή ως **VBA (Visual Basic for Applications)**. Δεν είναι λίγες όμως και οι περιπτώσεις ανάπτυξης εφαρμογών, στις οποίες η βάση δεδομένων έχει υλοποιηθεί χρησιμοποιώντας τη **Microsoft Access**, ενώ το βασικό περιβάλλον αλληλεπίδρασης με το χρήση έχει δημιουργηθεί χρησιμοποιώντας την πλήρη έκδοση της γλώσσας **Visual Basic**.

Αν και στις πιο πολλές περιπτώσεις, η υλοποίηση των διαδικασιών της αναπτυσσόμενης εφαρμογής μπορεί να γίνει και με πολλούς άλλους εναλλακτικούς τρόπους όπως είναι για παράδειγμα **δια της χρήσης μακροεντολών**, εν τούτοις η υλοποίηση αυτών των διαδικασιών δια μέσου μιας γλώσσας προγραμματισμού κρίνεται πιο αποδοτική, καθώς **διευκολύνει περισσότερο τη σωστή σχεδίαση και υλοποίηση της εφαρμογής**. Αυτό ισχύει ιδιαίτερα **στις περιπτώσεις σχεδίασης φορμών και αναφορών**, η προσπέλαση και χρήση των οποίων διευκολύνεται πάρα πολύ όταν στηρίζεται στη χρήση ειδικών προγραμμάτων που είναι γνωστά με το όνομα **λειτουργικές μονάδες (modules)**. Σύμφωνα με τα αρχεία τεκμηρίωσης της Microsoft Access, **μια λειτουργική μονάδα ορίζεται ως μια συλλογή δηλώσεων, προτάσεων και διαδικασιών, οι οποίες βρίσκονται αποθηκευμένες ως μία ενιαία οντότητα, και κάτω από ένα κοινό όνομα**. Στην περίπτωση κατά την οποία μία φόρμα ή αναφορά χαρακτηρίζεται από την ύπαρξη λειτουργικών μονάδων οι οποίες επιτρέπουν την προσπέλασή της από τον τελικό χρήστη, αυτές οι μονάδες αποτελούν τμήμα του αντικειμέ-

νου της φόρμας ή της αναφοράς, και επομένως, εάν το αντικείμενο αυτό διαγραφεί ή μετακινηθεί σε άλλη βάση, οι συσχετιζόμενες λειτουργικές μονάδες θα διαγραφούν ή θα μετακινηθούν ανάλογα. Αυτό όμως δεν συμβαίνει όταν η αλληλεπίδραση του χρήστη με τη φόρμα ή την αναφορά γίνεται δια μέσου κατάλληλα σχεδιασμένων μακροεντολών. Πράγματι, όπως έχει αναφερθεί στο προηγούμενο κεφάλαιο, **οι μακροεντολές αποθηκεύονται ως ξεχωριστά αντικείμενα της βάσης**, και επομένως, η διαγραφή ή μετακίνηση κάποιας φόρμας ή αναφοράς, θα απαιτεί ενδεχομένως και τη διαγραφή ή μετακίνηση της αντίστοιχης μακροεντολής. Επομένως η χρήση των λειτουργικών μονάδων στη θέση των μακροεντολών, **διευκολύνει τη συντήρηση της βάσης και καθιστά απλούστερη τη σχεδίαση της εφαρμογής**.

Η δεύτερη χαρακτηριστική περίπτωση στην οποία η χρήση λειτουργικών μονάδων προσφέρει περισσότερα πλεονεκτήματα σε σχέση με τη χρήση των μακροεντολών, αναφέρεται στη δημιουργία **προσαρμοσμένων συναρτήσεων (customized functions)** οι οποίες χρησιμοποιούνται για την πραγματοποίηση των κατάλληλων σε κάθε περίπτωση διαδικασιών. Υλοποιώντας τέτοιου είδους συναρτήσεις, **μπορούμε να βελτιώσουμε και να προσαρμόσουμε τη λειτουργία των ενσωματωμένων συναρτήσεων της Microsoft Access, έτσι ώστε να καλύψουμε τις ανάγκες μας**. Η χρήση των λειτουργικών μονάδων μας επιτρέπει ακόμη **να διαχειρισθούμε με πιο αποτελεσματικό τρόπο τα σφάλματα που λαμβάνουν χώρα κατά τη διάρκεια εκτέλεσης της εφαρμογής, εμφανίζοντας τα κατάλληλα σε κάθε περίπτωση μηνύματα σφάλματος**.

Στην τρέχουσα έκδοση της Microsoft Access, υπάρχουν δύο κατηγορίες λειτουργικών μονάδων. Η πρώτη από τις κατηγορίες αυτές που φέρει το όνομα **λειτουργικές μονάδες κλάσης (class modules)** χρησιμοποιείται για τον ορισμό **νέων τύπων αντικειμένων**, και είναι εντελώς ανάλογη με τις **κλάσεις (classes)** των αντικειμενοστραφών γλωσσών προγραμματισμού. Χαρακτηριστικό παράδειγμα αυτής της κατηγορίας λειτουργικών μονάδων, είναι **οι λειτουργικές μονάδες φόρμας και αναφοράς**, οι οποίες στη γενική περίπτωση περιλαμβάνουν διαδικασίες συμβάντος που καθορίζουν την απόκριση της φόρμας ή της αναφοράς σε συμβάντα συγκεκριμένου τύπου. Ας σημειωθεί πως σε αντίθεση με τις προηγούμενες εκδόσεις της **Microsoft Access** όπου οι λειτουργικές μονάδες κλάσης έπρεπε υποχρεωτικά να συσχετιστούνε με κάποια φόρμα ή αναφορά, στην τρέχουσα έκδοση μπορούμε να δημιουργήσουμε λειτουργικές μονάδες κλάσης χωρίς να εφαρμόσουμε υποχρεωτικά ένα τέτοιο συσχετισμό.

Από την άλλη πλευρά, οι **βασικές λειτουργικές μονάδες (standard modules)** περιλαμβάνουν **δηλώσεις, προτάσεις και διαδικασίες**, οι οποίες δεν συσχετίζονται με κάποιο αντικείμενο της βάσης, και μπορούν να κληθούν και να εκτελεστούν από οποιοδήποτε σημείο της εφαρμογής. Αυτού του είδους οι λειτουργικές μονάδες εμφανίζονται στην ομώνυμη σελίδα του κεντρικού παραθύρου διαχείρισης της βάσης δεδομένων, και μπορούν να χρησιμοποιηθούν είτε από μόνες τους, είτε μέσα από κάποια λειτουργική μονάδα κλάσης προκειμένου να υλοποιήσουν το είδος της αλληλεπίδρασης του χρήστη με κάποιο από τα αντικείμενα της εφαρμογής.

ΤΑ ΒΑΣΙΚΑ ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ ΤΗΣ ΓΛΩΣΣΑΣ VISUAL BASIC

Η γλώσσα **Visual Basic** ως μια τυπική γλώσσα προγραμματισμού υψηλού επιπέδου, χαρακτηρίζεται από όλες εκείνες τις δομικές μονάδες που περιγράφουν τις

γλώσσες προγραμματισμού αυτού του είδους όπως είναι **οι σταθερές (constants), οι μεταβλητές (variables), οι προτάσεις (statements) και οι μέθοδοι (methods) των αντικειμένων της βάσης**. Αν και η σημασία και χρήση όλων αυτών των δομικών μονάδων είναι παρόμοια με εκείνη των τυπικών γλωσσών προγραμματισμού, είναι χρήσιμο για λόγους πληρότητας να προχωρήσουμε σε μια συνοπτική περιγραφή του τρόπου με τον οποίο δηλώνονται και χρησιμοποιούνται μέσα από μία λειτουργική μονάδα της **Microsoft Access**. Σύμφωνα με τα αρχεία τεκμηρίωσης της εφαρμογής, τα πιο σημαντικά από τα χαρακτηριστικά της γλώσσας προγραμματισμού που χρησιμοποιούνται για την ανάπτυξη των λειτουργικών μονάδων, είναι τα ακόλουθα:

Σταθερές (Constants): οι σταθερές ορίζονται ως **δομικές μονάδες** οι οποίες περιγράφονται από κάποιο **όνομα** και φέρουν κάποια **τιμή** που παραμένει **αμετάβλητη** κατά τη διάρκεια εκτέλεσης του προγράμματος. Σε μία τυπική εφαρμογή αυτές οι σταθερές μπορεί να είναι τόσο **αριθμητικές σταθερές (ακέραιες ή πραγματικές)**, όσο και **σταθερές συμβολοσειρών (δηλαδή σταθερές χαρακτήρων)**. Εναλλακτικά η τιμή μιας σταθεράς μπορεί να μην είναι ένας απλός αριθμός αλλά **η τιμή μιας ολόκληρης αριθμητικής παράστασης ορισμένη από το χρήστη**. Η δήλωση μιας σταθεράς σε ένα πρόγραμμα της γλώσσας Visual Basic, γίνεται χρησιμοποιώντας τη δεσμευμένη λέξη **Const**. Για παράδειγμα, προκειμένου να ορίσουμε μία ακέραια σταθερά που να φέρει το όνομα **Age** και τιμή ίση με **18**, θα χρησιμοποιήσουμε τη δήλωση **Const Age As Integer = 18**.

Μεταβλητές (Variables): μια μεταβλητή ορίζεται ως μια **δομική μονάδα** η οποία περιγράφεται από κάποιο **όνομα** και φέρει κάποια **τιμή** που μπορεί να μεταβληθεί κατά τη διάρκεια της εκτέλεσης του προγράμματος. Επειδή η δήλωση μιας μεταβλητής προκαλεί **τη δέσμευση της κατάλληλης σε κάθε περίπτωση ποσότητας μνήμης** η διεύθυνση της οποίας ταυτοποιείται μονοσήμαντα από το όνομα αυτής της μεταβλητής, είναι προφανές πως **μέσα στην ίδια διαδικασία δεν είναι επιτρεπτή η δήλωση δύο μεταβλητών με το ίδιο όνομα**. Όσον αφορά τα ονόματα που μπορούμε να χρησιμοποιήσουμε για αυτές τις μεταβλητές, αυτά είναι **συμβολοσειρές** που θα πρέπει υποχρεωτικά να ξεκινάνε από **αλφαβητικό** (και όχι αριθμητικό) χαρακτήρα, και να έχουν μέγιστο μήκος ίσο με **255 χαρακτήρες**.

Μια μεταβλητή μπορεί να ανήκει σε πολλούς διαφορετικούς τύπους δεδομένων το πλήθος και το είδος των οποίων είναι συνάρτηση της γλώσσας προγραμματισμού που χρησιμοποιείται σε κάθε περίπτωση. Στην ειδική περίπτωση της γλώσσας **Visual Basic**, μια μεταβλητή μπορεί να ανήκει σε κάποιον από τους παρακάτω τύπους δεδομένων:

Όνομα Τύπου Δεδομένων	Μέγεθος σε Bytes	Εύρος Τιμών
Byte	**01**	0 έως 255
Boolean	**02**	TRUE ή FALSE
Integer	**02**	-32768 έως 32767
Long	**04**	2147483648 έως 2147483647
Single	**04**	-3.402823E38 έως 1.401298E-45 για αρνητικές τιμές και 1.401298E-45 έως 3.402823E38 για θετικές τιμές
Double	**08**	-1.79769313486231E308 έως 4.94065645841247E-324 για αρνητικές τιμές και 4.94065645841247E-324 έως 1.79769313486232E308 για θετικές τιμές.
Currency	**08**	922337203685477,5808 έως 922337203685477,5807
Decimal	**14**	+/-79228162514264337593543950335 χωρίς τη χρήση δεκαδικών ψηφίων, ή +/-7.9228162514264337593543950335 με 28 θέσεις για δεκαδικά ψηφία. Ο μικρότερος μη μηδενικός αριθμός έχει τιμή +/-0.0000000000000000000000000001
Date	**08**	January 1, 100 έως December 31, 9999
Object	**04**	Οποιαδήποτε αναφορά σε αντικείμενο
String (μεταβλητού μήκους)	**10 + μήκος συμβολοσειράς**	0 έως 2000000000 περίπου
String (σταθερού μήκους)	**Μήκος συμβολοσειράς**	1 έως 65400 περίπου
Variant (με αριθμούς)	**16**	Ίδιο εύρος τιμών με εκείνο του τύπου δεδομένων Double
Variant (με χαρακτήρες)	**22 + μήκος συμβολοσειράς**	Ίδιο εύρος τιμών με εκείνο των συμβολοσειρών μεταβλητού μήκους (Variable Length Strings)
User Defined Data Type	**Μεταβλητό (εξαρτάται από το μέγεθος των απλών τύπων δεδομένων που περιέχει)**	Το εύρος δεδομένων του κάθε ατομικού στοιχείου (element) εξαρτάται από τον τύπο δεδομένων αυτού του στοιχείου

Είναι σημαντικό να αναφερθεί στο σημείο αυτό, πως στην περίπτωση δήλωσης **πινάκων** (για κάθε τύπο δεδομένων), οι απαιτήσεις σε μνήμη **δεν περιορίζονται μόνο στον αποθηκευτικό χώρο που θα καταλάβουν τα δεδομένα αυτών των πινάκων, αλλά αντίθετα απαιτούνται 20 επιπλέον bytes μνήμης για το αντικείμενο του πίνακα και 4 bytes μνήμης για κάθε μια από τις διαστάσεις που τον περιγρά-**

φουν. Το μέγεθος του αποθηκευτικού χώρου που απαιτείται για την αποθήκευση των δεδομένων του πίνακα, είναι προφανώς ίσο με το γινόμενο **του πλήθους των στοιχείων του πίνακα** επί το **μέγεθος του τύπου δεδομένων αυτών των στοιχείων**. Για παράδειγμα **η αποθήκευση ενός συνόλου 20 ακεραίων αριθμών, απαιτεί χώρο αποθήκευσης με μέγεθος 40 bytes**, διότι οι απαιτήσεις σε μνήμη του τύπου δεδομένων ακέραιων αριθμών (integer data type) είναι **2 bytes**.

Προτάσεις (Statements): μια πρόταση ορίζεται ως μια **συντακτικώς ορθή ομάδα λεκτικών μονάδων** η οποία χρησιμοποιείται **για την πραγματοποίηση κάποιας ενέργειας** καθώς επίσης **για τον ορισμό και την απόδοση τιμής σε κάποια μεταβλητή ή σταθερά**. Στις πιο συνηθισμένες περιπτώσεις η κάθε πρόταση καταλαμβάνει **μία και μόνο γραμμή στο πηγαίο αρχείο**, αν και υπάρχει η δυνατότητα να επεκταθεί σε περισσότερες από μία γραμμές, χρησιμοποιώντας τον ειδικό χαρακτήρα «_». Στην περίπτωση κατά την οποία μία πρόταση είναι αρκετά μικρή, μπορούμε να την τοποθετήσουμε στην ίδια γραμμή με την προηγούμενη πρόταση, τοποθετώντας ανάμεσά τους το διαχωριστικό χαρακτήρα **«:»**.

Μέθοδος (method): σύμφωνα με τις βασικές αρχές του αντικειμενοστραφούς προγραμματισμού, το κάθε **στιγμιότυπο** μιας κλάσης (που είναι γνωστό ως **αντικείμενο (object)**), χαρακτηρίζεται από την ύπαρξη κάποιων **ιδιοτήτων (attributes)** που περιγράφουν τη φύση του και τα χαρακτηριστικά του, καθώς και κάποιων **συναρτήσεων (member functions) ή μεθόδων (methods)** οι οποίες επιτρέπουν τη χρήση αυτού του αντικειμένου μέσα από τη συνάρτηση που το καλεί. Στην ειδική περίπτωση της **Visual Basic**, αυτές οι μέθοδοι συσχετίζονται με μια πληθώρα αντικειμένων διαφορετικού τύπου, τα πιο χαρακτηριστικά εκ των οποίων είναι οι **φόρμες** και οι **αναφορές** που χρησιμοποιούνται για τη διαχείριση και την προεπισκόπηση των δεδομένων της βάσης.

Διαδικασίες συνάρτησης και διαδικασίες ρουτίνας

Χρησιμοποιούμε τον όρο **διαδικασία (procedure)** για να χαρακτηρίσουμε μία **μονάδα κώδικα της γλώσσας Visual Basic**, δηλαδή **ένα σύνολο εντολών, δηλώσεων και μεθόδων που επιτελούν κάποια συγκεκριμένη λειτουργία**. Ανάλογα με τη φύση της και τον τρόπο με τον οποίο έχει υλοποιηθεί, μια διαδικασία καλείται **με ορίσματα ή χωρίς ορίσματα (arguments)**. Αυτά τα ορίσματα δεν είναι τίποτε άλλο από **σταθερές, μεταβλητές ή ακόμη και ολόκληρες εκφράσεις (expressions)** πάνω στις οποίες εφαρμόζεται ο αλγόριθμος που υλοποιείται από τη διαδικασία στην οποία αποστέλλονται. Μια διαδικασία μπορεί να επιστρέφει ή όχι κάποια τιμή στο κυρίως πρόγραμμα ή στη διαδικασία από την οποία καλείται, γεγονός που μας επιτρέπει να ομαδοποιήσουμε τις διαδικασίες στις δύο επόμενες κατηγορίες:

Διαδικασίες ρουτίνας (sub procedures): οι διαδικασίες αυτού του τύπου επιτελούν μία ή περισσότερες λειτουργίες χωρίς ωστόσο να επιστρέφουν κάποια τιμή. Για το λόγο αυτό δεν μπορούν να κληθούν μέσα από άλλες διαδικασίες ρουτίνας (nested procedures). Αυτού του είδους οι διαδικασίες χρησιμοποιούνται κατά κύριο λόγο **σε λειτουργικές μονάδες κλάσης (class modules)** οι οποίες επιτρέπουν την αλληλεπίδραση του χρήστη με τις φόρμες και τις αναφορές της εφαρμογής που διαχειρίζεται τη βάση δεδομένων μας. Ο κώδικας που περιλαμβάνεται στις διαδικασίες αυτού

του είδους, αναγράφεται ανάμεσα στο ζεύγος των δεσμευμένων προτάσεων **Begin Sub** και **End Sub**.

Διαδικασίες συνάρτησης (function procedures): σε αντίθεση με τις διαδικασίες ρουτίνας που δεν επιστρέφουν κάποια τιμή στη διαδικασία μέσα από την οποία καλούνται, **οι διαδικασίες συνάρτησης (που συχνά αποκαλούνται και απλά συναρτήσεις) επιστρέφουν μία τέτοια τιμή**, και επομένως μπορούν να χρησιμοποιηθούν σε πιο σύνθετες εκφράσεις και παραστάσεις. Ο κώδικας που περιλαμβάνεται στις διαδικασίες αυτού του είδους αναγράφεται ανάμεσα στο ζεύγος των δεσμευμένων προτάσεων **Begin Function** και End Function.

Η **Visual Basic** ως **γλώσσα προγραμματισμού υψηλού επιπέδου (high level programming language)**, παρέχει τη δυνατότητα διαμόρφωσης του περιεχομένου μιας λειτουργικής μονάδας **ως ένα σύνολο από διαδικασίες συνάρτησης ή ρουτίνας**. Η κάθε μία από αυτές τις διαδικασίες χρησιμοποιεί γενικά **το δικό της σύνολο μεταβλητών** – οι οποίες ονομάζονται **τοπικές μεταβλητές (local variables)** – αν και υπάρχει η δυνατότητα χρήσης **καθολικών μεταβλητών (global variables)**, οι οποίες δύνανται να χρησιμοποιηθούν **από το σύνολο των διαδικασιών που περιλαμβάνονται σε μία λειτουργική μονάδα**. Σύμφωνα με τις βασικές αρχές του **δομημένου προγραμματισμού (structured programming)**, το τμήμα του προγράμματος στο οποίο είναι ορατή κάποια από τις μεταβλητές που περιλαμβάνονται σε αυτό, ορίζεται ως η **εμβέλεια (scope)** αυτής της μεταβλητής. Στενά συνδεδεμένη με την έννοια της **εμβέλειας** είναι η έννοια του **χρόνου ζωής της μεταβλητής (variable lifetime)**, που ορίζεται ως **το χρονικό διάστημα κατά τη διάρκεια του οποίου η εν λόγω μεταβλητή διατηρεί την τιμή της**. Στην περίπτωση της γλώσσας Visual Basic, η εμβέλεια και ο χρόνος ζωής μιας μεταβλητής χαρακτηρίζονται από τους ακόλουθους κανόνες:

Εάν μια μεταβλητή ορίζεται μέσα σε μια συγκεκριμένη διαδικασία (συνάρτησης ή ρουτίνας), μπορεί να χρησιμοποιηθεί **μόνο από τις εντολές που περιλαμβάνονται σε αυτή, και όχι από τις υπόλοιπες διαδικασίες της λειτουργικής μονάδας**. Όσον αφορά το χρονικό διάστημα για το οποίο αυτή η μεταβλητή διατηρεί την τιμή της, αυτό είναι ίσο **με το χρόνο εκτέλεσης της διαδικασίας που την περιέχει**. Σε αυτό το χρόνο εκτέλεσης περιλαμβάνονται και τα χρονικά διαστήματα εκτέλεσης των διαδικασιών εκείνων που ενδεχομένως καλούνται μέσα από την τρέχουσα διαδικασία.

Από την άλλη πλευρά, εάν η μεταβλητή δεν ορίζεται μέσα σε κάποια διαδικασία αλλά **στο δηλωτικό τμήμα (declaration section) μιας λειτουργικής μονάδας** και δια της χρήσης της δεσμευμένης λέξης **Dim** (που θα μελετηθεί στη συνέχεια), μπορεί να χρησιμοποιηθεί **από όλες τις διαδικασίες που περιλαμβάνονται σε αυτή**, όχι όμως και από τις διαδικασίες των υπόλοιπων λειτουργικών μονάδων. Στην ορολογία του δομημένου προγραμματισμού, οι μεταβλητές αυτές χαρακτηρίζονται ως **ιδιωτικές (private)**. Υπάρχει όμως η δυνατότητα προσπέλασης μιας μεταβλητής από όλες τις λειτουργικές μονάδες της εφαρμογής, εάν δηλωθεί ως **κοινόχρηστη**, κάτι που γίνεται αναγράφοντας τη δεσμευμένη λέξη **Public** πριν από το όνομα αυτής της μεταβλητής.

Η έννοια της **ιδιωτικής** και της **κοινόχρηστης** μεταβλητής μπορεί χωρίς καμία τροποποίηση να επεκταθεί και στην περίπτωση των **διαδικασιών** που περιέχονται σε μία λειτουργική μονάδα. Μιλώντας γενικά, αυτού του είδους οι διαδικασίες θεωρούνται **κοινόχρηστες**, και επομένως μπορούν να χρησιμοποιηθούν **από όλες τις λειτουργικές μονάδες μιας εφαρμογής**. Εξαίρεση στον κανόνα αυτό, αποτελούν οι **διαδικασίες συμβάντος (event procedures)** οι οποίες πάντοτε θεωρούνται **ιδιωτικές**. Για το λόγο αυτό, κάθε φορά που δηλώνουμε μια διαδικασία συμβάντος, η **Visual Basic** τοποθετεί πάντοτε τη δεσμευμένη λέξη **Private** πριν από το όνομα αυτής της διαδικασίας. Για όλες τις υπόλοιπες διαδικασίες, εφ' όσον επιθυμούμε να τις διαπραγματευτούμε ως ιδιωτικές, θα πρέπει να καταχωρήσουμε από μόνοι μας τη λέξη **Private** αμέσως πριν από το όνομα που τις χαρακτηρίζει.

Δηλώσεις σταθερών, μεταβλητών και πινάκων.

Όπως έχει ήδη αναφερθεί σε προηγούμενη παράγραφο, η δήλωση μιας σταθεράς και η απόδοση τιμής σε αυτή, λαμβάνει χώρα δια της χρήσης της δεσμευμένης λέξης **Const**. Μετά τη δήλωσή της, **η σταθερά διατηρεί πάντα την τιμή που της έχει αποδοθεί**, και η οποία δεν δύναται να μεταβληθεί κατά τη διάρκεια εκτέλεσης του προγράμματος. Εφόσον η σταθερά δηλωθεί μέσα σε μια διαδικασία ή στο δηλωτικό τμήμα μιας λειτουργικής μονάδας, θεωρείται ως **ιδιωτική**, ενώ για να χρησιμοποιηθεί ως **κοινόχρηστη**, θα πρέπει πριν από το όνομά της, να αναγραφεί η δεσμευμένη λέξη **Public**. Είναι προφανές, πως η χρήση της δεσμευμένης λέξης **Private** επιτρέπει τη χρήση της σταθεράς ως **ιδιωτική**, κάτι που άλλωστε αποτελεί και την προεπιλεγμένη συμπεριφορά της εφαρμογής.

Η πρόταση δήλωσης μιας σταθεράς, στη γενική περίπτωση έχει τη μορφή

[Private | Public] Const [Constant Name] As [Data Type] = [Constant Value]

όπου η χρήση του χαρακτήρα «|» υποδηλώνει τον τελεστή της **λογικής διάζευξης (Boolean OR)** ενώ το όρισμα **[Data Type]** είναι το όνομα του τύπου δεδομένων της σταθεράς που θέλουμε να δηλώσουμε. Ως τύπο δεδομένων για τη νέα σταθερά μπορούμε να χρησιμοποιήσουμε οποιοδήποτε έγκυρο τύπο δεδομένων της γλώσσας **Visual Basic** (δηλαδή κάποιον από τους **Boolean, Byte, Integer, Long, Currency, Single, Double, Date, String** και **Variant**). Στην περίπτωση κατά την οποία επιθυμούμε να δηλώσουμε **δύο σταθερές μέσα από την ίδια πρόταση**, μπορούμε να το κάνουμε χρησιμοποιώντας το κόμμα «,» ως το χαρακτήρα διαχωρισμού. Έτσι η πρόταση

Const Age As Integer = 18, Salary as Currency = 30000

επιτρέπει την ταυτόχρονη δήλωση δύο σταθερών, εκ των οποίων η πρώτη είναι **ακέραιος αριθμός** και φέρει την τιμή **18**, ενώ η δεύτερη είναι **νομισματική μονάδα** και φέρει την τιμή **30000**.

Από την άλλη πλευρά, η δήλωση μιας **μεταβλητής (variable)** γίνεται με παρόμοιο τρόπο, και χρησιμοποιώντας τη δεσμευμένη λέξη **Dim**. Σε πλήρη αναλογία με τη δήλωση μιας σταθεράς, η δήλωση μιας μεταβλητής μπορεί να λάβει χώρα **είτε μέσα σε κάποια από τις διαδικασίες της λειτουργικής μονάδας, είτε στο δηλωτικό**

της τμήμα. Και στις δύο αυτές περιπτώσεις η εν λόγω μεταβλητή χρησιμοποιείται ως **ιδιωτική**, ενώ για να χαρακτηριστεί ως **κοινόχρηστη**, θα πρέπει στη δηλωτική της πρόταση και πριν από το όνομά της, να αναγραφεί η δεσμευμένη λέξη **Public**. Η χρήση της λέξης **Private** καθιστά τη μεταβλητή **ιδιωτική**, ενώ μια άλλη ενδιαφέρουσα περίπτωση είναι η χρήση της λέξης **Static**, η οποία επιτρέπει στην εν λόγω μεταβλητή **να διατηρεί την τιμή της ανάμεσα σε διαδοχικές κλήσεις της από τις διαδικασίες της λειτουργικής μονάδας**. Οι μεταβλητές αυτού του είδους ονομάζονται **στατικές μεταβλητές (static variables)**.

Η πρόταση δήλωσης μιας μεταβλητής, στη γενική περίπτωση έχει τη μορφή

[Private | Public | Dim] [Variable Name] As [Data Type]

όπου ο τύπος δεδομένων για τη νέα μεταβλητή μπορεί να είναι κάποιος από τους **Boolean, Byte, Integer, Long, Currency, Single, Double, Date, String (σταθερού ή μεταβλητού μήκους), Object ή Variant**. Εάν δεν καταχωρηθεί κάποιος τύπος δεδομένων, χρησιμοποιείται ο προεπιλεγμένος τύπος **Variant**. Η εφαρμογή παρέχει επίσης τη δυνατότητα να χρησιμοποιήσουμε ως τύπο δεδομένων κάποιο τύπο που έχει ορισθεί από το χρήστη σε προηγούμενο στάδιο της διαδικασίας **(user defined data type)** - η δήλωση νέων τύπων δεδομένων, γίνεται χρησιμοποιώντας τη δεσμευμένη λέξη **Type**. Τέλος, σε πλήρη αναλογία με τη δήλωση μιας σταθεράς, υπάρχει η δυνατότητα να ορίσουμε ταυτόχρονα πολλές μεταβλητές μέσα από την ίδια πρόταση, και χρησιμοποιώντας το κόμμα «,» ως διαχωριστικό ανάμεσα σε δύο διαδοχικές δηλώσεις μεταβλητών. Έτσι η δήλωση

Dim Color As Integer, Mean, Fract As Double

ορίζει τρεις μεταβλητές, τις **Color**, **Mean** και **Fract** εκ των οποίων η πρώτη είναι τύπου δεδομένων **Integer**, η δεύτερη τύπου δεδομένων **Variant** (αφού δεν έχει καθορισθεί κάποιος τύπος δεδομένων), και η τρίτη τύπου δεδομένων **Double**.

Η τελευταία ενδιαφέρουσα περίπτωση που θα εξεταστεί σε αυτή την ενότητα, αφορά τη δήλωση **πινάκων (arrays)**. Ένας πίνακας ορίζεται ως μία **συλλογή αντικειμένων που βρίσκονται αποθηκευμένα σε διαδοχικές θέσεις μνήμης συγκεκριμένου μεγέθους**. Η δήλωση ενός πίνακα γίνεται χρησιμοποιώντας τις δεσμευμένες λέξεις **Dim, Static, Private** και **Public** και ακολουθεί **τον ίδιο τρόπο σύνταξης με αυτόν που χρησιμοποιείται για τη δήλωση των μεταβλητών**. Άλλωστε η βασική διαφορά που υφίσταται ανάμεσα στη δήλωση μιας **μεταβλητής** και στη δήλωση ενός **πίνακα**, είναι πως στη δεύτερη περίπτωση θα πρέπει **εκτός από τον τύπο δεδομένων των στοιχείων του πίνακα να δηλώσουμε και το πλήθος αυτών των στοιχείων**. Ένας πίνακας λέγεται **στατικός (fixed array)** όταν το μέγεθός του **δεν μεταβάλλεται κατά τη διάρκεια της εκτέλεσης του προγράμματος**, αν και υπάρχει η δυνατότητα δήλωσης και χρήσης **δυναμικών πινάκων (dynamic arrays)** το μέγεθος των οποίων μεταβάλλεται κατά βούληση έτσι ώστε να καλύπτονται οι ανάγκες που υφίστανται σε κάθε περίπτωση. Η δήλωση ενός δυναμικού πίνακα ακολουθεί την ίδια σύνταξη με τη δήλωση ενός στατικού πίνακα, με τη διαφορά πως δεν καθορίζουμε το πλήθος των στοιχείων του, αλλά αφήνουμε κενές τις παρενθέσεις που ακολουθούν αμέσως μετά το όνομά του. Έτσι η δήλωση **Dim Samples () As Integer**, ορίζει ένα δυναμικό πίνακα ακεραίων στοιχείων. Στην περίπτωση κατά την οποία στη δήλωση ενός στατικού ή

ενός δυναμικού πίνακα δεν καθορίσουμε τον τύπο δεδομένων των στοιχείων που περιλαμβάνονται σε αυτόν, χρησιμοποιείται ο προεπιλεγμένος τύπος **Variant**.

Η γλώσσα προγραμματισμού **Visual Basic** παρέχει τη δυνατότητα χρήσης τόσο **μονοδιάστατων** όσο και **πολυδιάστατων** πινάκων. Η διάσταση αυτών των πινάκων καθορίζεται μονοσήμαντα από τον τρόπο με τον οποίο λαμβάνει χώρα η δήλωσή τους. Έτσι η δήλωση **Dim points (50) As Integer** ορίζει ένα μονοδιάστατο πίνακα ακεραίων **51 θέσεων** (η προεπιλεγμένη αρίθμηση των θέσεων των πινάκων ξεκινάει από τιμή ίση με το **0**), ενώ η δήλωση **Dim coordinates (19, 19) As Double** ορίζει ένα δισδιάστατο πίνακα πραγματικών αριθμών **διπλής ακρίβειας (double precision)** με **20 γραμμές και 20 στήλες**. Από τα δύο ορίσματα που χρησιμοποιούνται στον ορισμό του πλήθους των κελιών ενός πίνακα δύο διαστάσεων, το πρώτο αναφέρεται **στον αριθμό των γραμμών και το δεύτερο στον αριθμό των στηλών του πίνακα**. Στην περίπτωση κατά την οποία επιθυμούμε η αρίθμηση των κελιών να ξεκινά από το **1** και όχι από το **0**, θα πρέπει πριν από τις δηλώσεις των πινάκων της λειτουργικής μονάδας, να καταχωρήσουμε την πρόταση **Option Base 1**.

Οι τελεστές της γλώσσας Visual Basic

Σε πλήρη αναλογία με τις υπόλοιπες γλώσσες προγραμματισμού, η γλώσσα **Visual Basic** χαρακτηρίζεται από την ύπαρξη ενός συνόλου **τελεστών (operators)** που χρησιμοποιούνται κατά τη διαδικασία **δόμησης εκφράσεων**. Τυπικό παράδειγμα ενός τέτοιου τελεστή, είναι **ο τελεστής καταχώρησης (assignment operator) «=»**, ο οποίος χρησιμοποιείται **για την απόδοση τιμών στις διάφορες σταθερές και μεταβλητές μιας διαδικασίας**. Οι τελεστές της γλώσσας **Visual Basic**, ομαδοποιούνται όπως συνήθως σε **αριθμητικούς (arithmetic), συγκριτικούς (comparison)** και **λογικούς (logical) τελεστές**, ενώ υπάρχει και άλλος ένας τελεστής που χρησιμοποιείται για τη **συνένωση (concatenation) μεταβλητών συμβολοσειράς (concatenation operators)**. Σε μια πιο αναλυτική περιγραφή, οι τελεστές που μπορούν να χρησιμοποιηθούνε για τη δόμηση εκφράσεων μέσα από τη γλώσσα προγραμματισμού **Visual Basic**, είναι οι ακόλουθοι:

- **Αριθμητικοί Τελεστές (Arithmetic Operators):** στην κατηγορία αυτή των τελεστών, **ανήκουν ο τελεστής ύψωσης σε δύναμη (^), οι τελεστές πρόσθεσης (+), αφαίρεσης (-), πολλαπλασιασμού (*) και διαίρεσης (/), ο τελεστής υπολοίπου (module) (mod)**, και **ο τελεστής (\) που διαιρεί δύο αριθμούς αλλά επιστρέφει μόνο το ακέραιο μέρος του πηλίκου της διαίρεσης**.

- **Συγκριτικοί Τελεστές (Comparison Operators):** στην κατηγορία αυτή ανήκουν οι τελεστές **«μικρότερο από» (<), «μικρότερο ή ίσο με » (<=), «μεγαλύτερο από» (>), «μεγαλύτερο ή ίσο με» (>=), «ίσο με» (=) και «όχι ίσο με» (<>).**

- **Τελεστής Συνένωσης Συμβολοσειράς (Concatenation Operator):** στην περίπτωση κατά την οποία επιθυμούμε να συνενώσουμε τα περιεχόμενα δύο συμβολοσειρών προκειμένου να σχηματίσουμε μια μεγαλύτερη συμβολοσειρά, μπορούμε να το κάνουμε χρησιμοποιώντας τον τελεστή **«&»**. Εάν για παράδειγμα ορίσουμε μια μεταβλητή συμβολοσειράς με το όνομα **Message**, η δήλωση **Message**

= **"Hello" & "World"** θα έχει ως αποτέλεσμα την καταχώρηση στη μεταβλητή **Message**, της σταθερής συμβολοσειράς **«Hello World»**.

- **Λογικοί Τελεστές (Logical Operators):** τέλος στην κατηγορία των λογικών τελεστών, ανήκουν οι γνωστοί τελεστές **AND, OR, NOT, XOR, EQV** και **IMP** που χρησιμοποιούνται στη διαδικασία δόμησης λογικών εκφράσεων, οι οποίες μπορούν να λάβουν μόνο τις τιμές **TRUE** ή **FALSE**.

ΠΡΑΓΜΑΤΟΠΟΙΗΣΗ ΕΝΕΡΓΕΙΩΝ ΔΙΑ ΤΗΣ ΧΡΗΣΗΣ ΛΕΙΤΟΥΡΓΙΚΩΝ ΜΟΝΑΔΩΝ

Το πιο χαρακτηριστικό παράδειγμα χρήσης των λειτουργικών μονάδων μέσα από την τρέχουσα βάση δεδομένων, είναι **η πραγματοποίηση των ενεργειών (actions) που περιγράψαμε στο κεφάλαιο των μακροεντολών**. Αυτό σημαίνει πως αυτές οι ενέργειες μπορούν να εκτελεστούν **τόσο μέσα από μια μακροεντολή όσο και μέσα από μια λειτουργική μονάδα**, και ο τρόπος κλήσεις τους που θα επιλέξουμε, εξαρτάται από τη φύση και τα χαρακτηριστικά της εφαρμογής. Είναι προφανές πως όποιον από τους δύο τρόπους και αν διαλέξουμε για να εκτελέσουμε κάποια ενέργεια, θα πρέπει να καθορίσουμε τα ορίσματα που τη συνοδεύουν έτσι ώστε να περιγράψουμε πλήρως τη λειτουργία της. Στην ορολογία της **Microsoft Access**, οι ενέργειες που καλούνται μέσα από μία λειτουργική μονάδα, είναι γνωστές ως **μέθοδοι (methods)**.

Μιλώντας γενικά, η έννοια της **μεθόδου** είναι άρρηκτα συνδεδεμένη με την έννοια του **αντικειμενοστραφούς προγραμματισμού (object oriented programming)**. Μια **μέθοδος** ορίζεται ως μια **διαδικασία** η οποία εφαρμόζεται πάνω σε κάποιο **αντικείμενο**. Το κάθε αντικείμενο χαρακτηρίζεται από την παρουσία πολλών μεθόδων δια της χρήσης των οποίων **προσπελαύνουμε τα χαρακτηριστικά του συγκεκριμένου αντικειμένου, και καθορίζουμε τον τρόπο με τον οποίο αυτό χρησιμοποιείται**. Η **Microsoft Access** διαθέτει μια πληθώρα αντικειμένων αυτού του τύπου που χρησιμοποιούνται ανάλογα με τις περιστάσεις και όπου αυτό είναι αναγκαίο. Ωστόσο στις σελίδες που ακολουθούν θα χρησιμοποιήσουμε μόνο δύο από αυτά τα αντικείμενα, που φέρουν τα ονόματα **Application** και **DoCmd**.

Το αντικείμενο της εφαρμογής **(Application Object)** αναφέρεται **στην τρέχουσα εφαρμογή της Microsoft Access** και περιέχει όλα τα επιμέρους αντικείμενα που περιλαμβάνονται σε αυτή (όπως είναι για παράδειγμα τα αντικείμενα **φόρμας** και **αναφοράς**). Αυτό το αντικείμενο χρησιμοποιείται προκειμένου να καθορίσουμε **τις παραμέτρους λειτουργίας της εφαρμογής**, στο σύνολό της, μέσα από τη γλώσσα **Visual Basic**. Ως παράδειγμα χρήσης αυτού του αντικειμένου αναφέρουμε **τη διαδικασία εμφάνισης της γραμμής κατάστασης (status bar) της εφαρμογής**, η οποία γίνεται με μία εντολή της μορφής **Application.SetOption "Show Status Bar", True**. Το αποτέλεσμα αυτής της εντολής είναι **η εμφάνιση της γραμμής εργαλείων στο κάτω μέρος της οθόνης**, και είναι το ίδιο με εκείνο που θα προέκυπτε εάν ανοίγαμε το πλαίσιο διαλόγου **«Επιλογές»** – για να το κάνουμε αυτό μεταφερόμαστε στο κεντρικό menu επιλογών της **Access** και ακολουθούμε τη διαδρομή **Εργαλεία→Επιλογές** – και ενεργοποιούσαμε το **check box** με τίτλο **«Γραμμή Κατάστα-**

σης». Περισσότερες λεπτομέρειες για τις μεθόδους του αντικειμένου της εφαρμογής μπορούν να βρεθούν στα αρχεία τεκμηρίωσης της **Microsoft Access**.

Το δεύτερο σημαντικό αντικείμενο που θα μας απασχολήσει σε αυτή την ενότητα, είναι το αντικείμενο **DoCmd** που χρησιμοποιείται **για την πραγματοποίηση των ενεργειών των μακροεντολών μέσα από τη γλώσσα Visual Basic**. Αυτό το αντικείμενο περιέχεται μέσα στο αντικείμενο της εφαρμογής, και οι μέθοδοι που περιλαμβάνονται σε αυτό, επιτρέπουν την πραγματοποίηση των πιο χαρακτηριστικών λειτουργιών της εφαρμογής, όπως είναι **η χρήση φορμών και αναφορών, η εκτέλεση ερωτημάτων, και η μετακίνηση του χρήστη ανάμεσα στις εγγραφές των πινάκων της βάσης**. Υπάρχουν ωστόσο και ενέργειες οι οποίες δεν μπορούν να εκτελεσθούν δια της χρήσης των μεθόδων αυτού του αντικειμένου, όπως είναι για παράδειγμα **η εμφάνιση κάποιου μηνύματος, η προσθήκη νέων menus επιλογών και ο τερματισμός της εκτέλεσης μιας μακροεντολής**. Αυτού του είδους οι διαδικασίες πραγματοποιούνται δια της χρήσης εναλλακτικών μεθόδων – για παράδειγμα η εμφάνιση ενός μηνύματος λαμβάνει χώρα χρησιμοποιώντας τη συνάρτηση **MsgBox** – ενώ εκείνες που υποστηρίζονται από το αντικείμενο **DoCmd**, καλούνται μέσω μιας πρότασης της μορφής

[application].DoCmd.MethodName [argument list]

Στην παραπάνω πρόταση το όρισμα **application** αναφέρεται στο αντικείμενο της εφαρμογής (η χρήση αυτού του ορίσματος είναι προαιρετική καθώς εάν δεν καθορισθεί υπονοείται πως χρησιμοποιείται το αντικείμενο της τρέχουσας εφαρμογής), ενώ **MethodName** είναι το όνομα της μεθόδου του αντικειμένου **DoCmd** που επιθυμούμε να χρησιμοποιήσουμε. Δεν είναι δύσκολο να διαπιστώσει κανείς, πως **η κλήση αυτής της μεθόδου ακολουθεί τους συντακτικούς κανόνες που χρησιμοποιούνται σε όλες τις αντικειμενοστραφείς γλώσσες προγραμματισμού για την προσπέλαση μεθόδων αντικειμένων**, και οι οποίοι χρησιμοποιούν την τελεία «**.**» ως το **χαρακτήρα διαχωρισμού** του ονόματος της μεθόδου από το όνομα του αντικειμένου. Η κάθε μία από αυτές τις μεθόδους του αντικειμένου **DoCmd** καλείται με τον κατάλληλο σε κάθε περίπτωση αριθμό ορισμάτων, ενώ στην περίπτωση κατά την οποία κάποια ορίσματα δεν καθορισθούν, χρησιμοποιούνται για αυτά **προεπιλεγμένες τιμές (default values)**. Χαρακτηριστικό παράδειγμα αυτής της περίπτωσης είναι η μέθοδος **OpenForm** που χρησιμοποιούμε για να ανοίξουμε κάποια από τις φόρμες της εφαρμογής. Αν και αυτή η μέθοδος απαιτεί τον καθορισμό πολλών ορισμάτων (που περιλαμβάνουν μεταξύ των άλλων **τον καθορισμό της προβολής της φόρμας (Normal or Design View) και του φίλτρου που ενδεχομένως επιθυμούμε να εφαρμόσουμε πάνω στα δεδομένα που περιλαμβάνονται σε αυτή**), εν τούτοις το μόνο όρισμα που θα πρέπει να καθορίσομε υποχρεωτικά είναι το όνομά της, ενώ για τα υπόλοιπα ορίσματα θα χρησιμοποιηθούν προεπιλεγμένες τιμές – η προεπιλεγμένη τιμή για την προβολή της φόρμας είναι η **Normal View**. Η αναλυτική παρουσίαση των μεθόδων του αντικειμένου **DoCmd** παρουσιάζεται στις επόμενες ενότητες.

ΟΙ ΜΕΘΟΔΟΙ ΤΟΥ ΑΝΤΙΚΕΙΜΕΝΟΥ DOCMD

Ακύρωση συμβάντος (CancelEvent Method): χρησιμοποιούμε τη μέθοδο **CancelEvent** για να ακυρώσουμε το συμβάν που προκάλεσε την εκτέλεση της τρέχουσας διαδικασίας. Η μέθοδος αυτή καλείται με τη μορφή **DoCmd.CancelEvent** και δεν περιλαμβάνει τη χρήση ορισμάτων.

Άνοιγμα αποθηκευμένης διαδικασίας (OpenStoredProcedure Method): σε πλήρη αναλογία με την ομώνυμη ενέργεια που περιγράψαμε στο κεφάλαιο των μακροεντολών, αυτή η μέθοδος επιτρέπει **τη χρήση μιας αποθηκευμένης διαδικασίας σε προβολή σχεδίασης (design view), προβολή εκτέλεσης (open view) ή σε προεπισκόπηση εκτύπωσης (print preview)**. Η κλήση αυτής της μεθόδου ακολουθεί τη σύνταξη

DoCmd.OpenStoredProcedure procedureName, viewMode, dataMode

όπου **procedureName** είναι **το όνομα της διαδικασίας** που επιθυμούμε να χρησιμοποιήσουμε, **viewMode** είναι η **προβολή** στην οποία θα ανοίξουμε την εν λόγω διαδικασία και **dataMode** είναι **το είδος της προσπέλασης** που μπορούμε να εφαρμόσουμε πάνω στα δεδομένα που συσχετίζονται με αυτή. Από τα τρία αυτά ορίσματα το όνομα της διαδικασίας είναι **υποχρεωτικό**, ενώ τα άλλα δύο ορίσματα είναι προαιρετικά και χρησιμοποιούνται ανάλογα με τις περιστάσεις. Όσον αφορά τις τιμές που μπορούν να λάβουν αυτά τα ορίσματα, αυτές περιλαμβάνονται στον ακόλουθο πίνακα:

Όρισμα viewMode	**acViewNormal**	Επιτρέπει τη χρήση της αποθηκευμένης διαδικασίας σε προβολή εκτέλεσης
	acViewDesign	Επιτρέπει τη χρήση της αποθηκευμένης διαδικασίας σε προβολή σχεδίασης
	acViewPreview	Μεταφέρει την αποθηκευμένη διαδικασία σε προεπισκόπηση εκτύπωσης
Όρισμα dataMode	**acAdd**	Επιτρέπει την προσθήκη δεδομένων μέσα από την τρέχουσα διαδικασία
	acEdit	Επιτρέπει τη μεταβολή των δεδομένων μέσα από την τρέχουσα διαδικασία
	acReadOnly	Επιτρέπει μόνο την ανάγνωση των δεδομένων μέσα από την τρέχουσα διαδικασία, ενώ απαγορεύει την τροποποίησή τους

Στην περίπτωση κατά την οποία δεν καθορίσουμε κάποιες τιμές για τα ορίσματα **viewMode** και **dataMode**, χρησιμοποιούνται οι προεπιλεγμένες τιμές **acViewNormal** και **acEdit** αντίστοιχα.

Παράδειγμα χρήσης της μεθόδου **OpenStoredProcedure** είναι η κλήση της με τη μορφή

DoCmd.OpenStoredProcedure "Employees", acViewNormal, acReadOnly

η οποία επιτρέπει τη χρήση της αποθηκευμένης διαδικασίας «**Employees**» σε **προβολή εκτέλεσης** αλλά χωρίς τη δυνατότητα τροποποίησης των δεδομένων που συσχετίζονται με αυτή.

Άνοιγμα διαγράμματος (OpenDiagram Method): χρησιμοποιούμε τη μέθοδο **OpenDiagram** για να ανοίξουμε ένα **διάγραμμα** της βάσης δεδομένων μέσα από την τρέχουσα διαδικασία. Η κλήση της μεθόδου ακολουθεί τη σύνταξη

DoCmd.OpenDiagram diagramName

όπου **diagramName** είναι το όνομα του διαγράμματος που επιθυμούμε να χρησιμοποιήσουμε. Η χρήση αυτής της μεθόδου, δεν απαιτεί τον καθορισμό επιπλέον ορισμάτων.

Άνοιγμα έκθεσης (OpenReport Method): χρησιμοποιούμε τη μέθοδο **OpenReport** για να εμφανίσουμε ή να εκτυπώσουμε **τα περιεχόμενα μιας αναφοράς της βάσης δεδομένων** μέσα από την τρέχουσα διαδικασία. Η κλήση αυτής της μεθόδου ακολουθεί τη σύνταξη

DoCmd.OpenReport reportName, reportView, filterName, whereCondition

όπου το όρισμα **reportName** είναι **το όνομα της αναφοράς** που θέλουμε να χρησιμοποιήσουμε, το όρισμα **reportView** καθορίζει **το είδος της προβολής** στην οποία θα χρησιμοποιήσουμε την εν λόγω αναφορά, ενώ τα ορίσματα **filterName** και **whereCondition** επιτρέπουν τη χρήση όχι όλων των εγγραφών της αναφοράς, αλλά μόνο εκείνων που πληρούνε κάποια κριτήρια.

Από τα τέσσερα παραπάνω ορίσματα, το μόνο που θα πρέπει να καταχωρηθεί υποχρεωτικά είναι το **reportName**, που καθορίζει **το όνομα της τρέχουσας αναφοράς**. Από την άλλη πλευρά, οι τιμές που μπορεί να λάβει το όρισμα **reportView** είναι η **acViewDesign** που ανοίγει την αναφορά **σε προβολή σχεδίασης**, η **acViewNormal** που επιτρέπει **την άμεση εκτύπωση της αναφοράς** (η τιμή αυτή είναι και η προεπιλεγμένη), και η **acViewPreview** που ανοίγει την αναφορά σε προεπισκόπηση εκτύπωσης. Τέλος, το όρισμα **filterName** δέχεται ως τιμή το όνομα κάποιου **ερωτήματος (query)** της βάσης δεδομένων που επιθυμούμε να εφαρμόσουμε πάνω στα δεδομένα της αναφοράς, ενώ η τιμή που χρησιμοποιούμε για το όρισμα **whereCondition** είναι μία έγκυρη πρόταση **WHERE** της εντολής **SELECT** της γλώσσας **SQL**, αλλά χωρίς την ίδια τη λέξη **WHERE** (η οποία υπονοείται). Για παράδειγμα προκειμένου να εμφανίσουμε μόνο τις εγγραφές της αναφοράς «**Κατάλογος Υπαλλήλων**» για τις οποίες το πεδίο **SALARY** έχει τιμές μεγαλύτερες από **30000**, θα καταχωρήσουμε στο όρισμα αυτό τη συμβολοσειρά **SALARY>30000**.

Παράδειγμα χρήσης αυτής της μεθόδου, είναι η κλήση της με τη μορφή

DoCmd.OpenReport "Project Report", acViewPreview

η οποία ανοίγει την αναφορά «**Project Report**» σε προεπισκόπηση εκτύπωσης.

Άνοιγμα ερωτήματος (OpenQuery Method): χρησιμοποιούμε τη μέθοδο **OpenQuery** για να ανοίξουμε ένα ερώτημα **σε προβολή σχεδίασης, προβολή φύλλου δεδομένων ή προεπισκόπησης εκτύπωσης**. Η κλήση αυτής της μεθόδου ακολουθεί τη σύνταξη

DoCmd.OpenQuery queryName, queryView, dataMode

όπου το όρισμα **queryName** είναι **το όνομα του ερωτήματος** που επιθυμούμε να χρησιμοποιήσουμε, το όρισμα **queryView** καθορίζει την προβολή στην οποία θα ανοίξει το ερώτημα, ενώ το όνομα **dataView** επιτρέπει τον καθορισμό του είδους της πρόσβασης πάνω στα δεδομένα που επιστρέφονται από το ερώτημα.

Από τα παραπάνω ορίσματα, εκείνο που θα πρέπει να χρησιμοποιήσουμε υποχρεωτικά είναι το όρισμα **queryName**, ενώ τα υπόλοιπα χρησιμοποιούνται κατά βούληση και όπου αυτό είναι αναγκαίο. Σε πλήρη αναλογία με προηγούμενες μεθόδους, το όρισμα **queryView** δέχεται μία από τις τιμές **acViewDesign**, **acViewNormal** και **acViewPreview**. Από τις τρεις αυτές τιμές, η πρώτη προκαλεί τη χρήση του ερωτήματος **σε προβολή σχεδίασης (design view)**, η δεύτερη (που είναι και η προεπιλεγμένη) προκαλεί **την εκτέλεση του ερωτήματος** (δηλαδή την προβολή του σε μορφή **φύλλου δεδομένων** (datasheet view)), ενώ η τιμή **acViewPreview** εμφανίζει το ερώτημα σε προβολή **προεπισκόπησης εκτύπωσης**. Από την άλλη πλευρά, το όρισμα **dataMode** παίρνει μία από τις τιμές **acAdd**, **acEdit** (η τιμή αυτή είναι και η προεπιλεγμένη) και **acReadOnly** που επιτρέπουν **την προσθήκη εγγραφών, την τροποποίηση των δεδομένων του ερωτήματος, και την εργασία με αυτά τα δεδομένα σε κατάσταση μόνο ανάγνωσης (read only)** αντίστοιχα.

Παράδειγμα χρήσης της μεθόδου **OpenQuery** είναι η κλήση της με τη μορφή

DoCmd.OpenQuery "Employee List", acReadOnly

που έχει ως αποτέλεσμα **την προεπισκόπηση του ερωτήματος "Employee List" σε κατάσταση μόνο ανάγνωσης**.

Άνοιγμα Λειτουργικής Μονάδας (OpenModule Method): χρησιμοποιούμε τη μέθοδο **OpenModule** για να προσπελάσουμε τις διαδικασίες μιας λειτουργικής μονάδας μέσα από την τρέχουσα διαδικασία. Αυτή η μέθοδος είναι εντελώς ανάλογη με την ενέργεια **OpenModule** που περιγράψαμε στο κεφάλαιο διαχείρισης μακροεντολών.

Η κλήση της μεθόδου OpenModule χαρακτηρίζεται από μία σύνταξη της μορφής

DoCmd.OpenModule moduleName, procedureName

όπου **procedureName** είναι το όνομα της διαδικασίας που θέλουμε να χρησιμοποιήσουμε, και **moduleName** το όνομα της λειτουργικής μονάδας στην οποία ανήκει αυτή η διαδικασία.

Στην περίπτωση κατά την οποία καθορίσουμε όνομα διαδικασίας αλλά όχι όνομα λειτουργικής μονάδας, η **Microsoft Access** θα αναζητήσει την καθορισμένη διαδικασία **στις βασικές λειτουργικές μονάδες της βάσης (standard modules)** και θα ανοίξει εκείνη τη λειτουργική μονάδα στην οποία θα εντοπίσει αυτή τη διαδικασία. Από την άλλη πλευρά, εάν καθορίσουμε όνομα λειτουργικής μονάδας αλλά όχι και όνομα διαδικασίας, η **Microsoft Access** θα ανοίξει τη λειτουργική μονάδα που έχει καθορισθεί, και θα μεταφέρει τον έλεγχο της εφαρμογής **στο δηλωτικό τμήμα (declaration section) αυτής της μονάδας**. Σε κάθε περίπτωση, η χρήση της μεθόδου **OpenModule** απαιτεί τον καθορισμό **τουλάχιστον ενός από τα ορίσματα που την περιγράφουν**. Είναι προφανές, πως εάν καθορισθούν και τα δύο αυτά ορίσματα, η **Microsoft Access** θα ανοίξει τη λειτουργική μονάδα που έχει καθορισθεί, και θα εκτελέσει τη διαδικασία της οποίας το όνομα έχει περαστεί ως τιμή στο όρισμα **procedureName**.

Χαρακτηριστικό παράδειγμα χρήσης της μεθόδου **OpenModule** είναι η κλήση της με τη μορφή

DoCmd.OpenModule “Utility Functions”, “IsLoaded”

που προκαλεί την εκτέλεση της διαδικασίας **IsLoaded** που ανήκει στη λειτουργική μονάδα **Utility Functions**. Το παράδειγμα αυτό έχει ληφθεί από τα αρχεία τεκμηρίωσης της **Microsoft Access** που αναφέρονται στη μέθοδο **OpenModule**.

Άνοιγμα πίνακα (OpenTable Method): χρησιμοποιούμε τη μέθοδο **OpenTable** για να ανοίξουμε ένα πίνακα της βάσης δεδομένων **σε προβολή σχεδίασης, προβολή φύλλου δεδομένων ή προεπισκόπησης εκτύπωσης**. Αυτή η μέθοδος είναι ανάλογη με την ομώνυμη ενέργεια που περιγράψαμε στο κεφάλαιο διαχείρισης μακροεντολών, και η κλήση της χαρακτηρίζεται από μία σύνταξη της μορφής

DoCmd.OpenTable tableName, tableView, dataMode

Στη σύνταξη αυτή το όρισμα **tableName** καθορίζει **το όνομα του πίνακα** που επιθυμούμε να χρησιμοποιήσουμε. Η χρήση του ορίσματος αυτού είναι υποχρεωτική, ενώ τα υπόλοιπα δύο ορίσματα χρησιμοποιούνται κατά βούληση και ανάλογα με τις περιστάσεις. Όσον αφορά τις τιμές που μπορούν να λάβουν τα ορίσματα αυτά, ισχύουν τα όσα έχουν αναφερθεί στις προηγούμενες παραγράφους. Έτσι το όρισμα **tableView** μπορεί να λάβει μία από τις τιμές **acViewDesign**, **acViewNormal** και **acViewPreview** ενώ το όρισμα **dataMode** μπορεί να λάβει κάποια από τις τιμές **acAdd**, **acEdit** και **acReadOnly**. Εάν δεν αποδώσουμε τιμή στα δύο αυτά ορίσματα, χρησιμοποιούνται οι προεπιλεγμένες τιμές **acViewNormal** και **acEdit** αντίστοιχα.

Χαρακτηριστικό παράδειγμα χρήσης της μεθόδου **OpenTable** είναι η κλήση της με τη μορφή

DoCmd.OpenTable “EMPLOYEE”, acViewDesign

η οποία ανοίγει τον πίνακα **EMPLOYEE** σε προβολή σχεδίασης.

Άνοιγμα προβολής (OpenView Method): χρησιμοποιούμε τη μέθοδο **OpenView**, για να ανοίξουμε μία **προβολή (view)** σε προβολή σχεδίασης, προβολή φύλλου δεδομένων ή προεπισκόπηση εκτύπωσης. Η κλήση αυτής της μεθόδου ακολουθεί τη σύνταξη

DoCmd.OpenView viewName, viewMode, dataMode

Στην παραπάνω πρόταση το όρισμα **viewName** καθορίζει **το όνομα της προβολής** που θέλουμε να χρησιμοποιήσουμε ενώ τα ορίσματα **viewMode** και **dataMode**, σε πλήρη αναλογία με την προηγούμενη περίπτωση, καθορίζουν **το είδος της προβολής που θέλουμε να χρησιμοποιήσουμε, και το είδος της πρόσβασης επί των δεδομένων που συσχετίζονται με την καθορισμένη προβολή**. Το σύνολο τιμών και τρόπος χρήσης αυτών των ορισμάτων είναι παρόμοια με εκείνα που παρουσιάσαμε στην προηγούμενη μέθοδο **(OpenTable)** και δεν κρίνεται σκόπιμο να παρουσιαστούν εκ νέου. Παράδειγμα χρήσης της μεθόδου **OpenView** είναι η κλήση της με τη μορφή

DoCmd.OpenView "Departments", acViewNormal, acReadOnly

που επιτρέπει την προεπισκόπηση της προβολής **Departments** σε προβολή φύλλου δεδομένων και σε κατάσταση μόνο ανάγνωσης.

Άνοιγμα σελίδας πρόσβασης δεδομένων (OpenDataAccessPage Method): χρησιμοποιούμε τη μέθοδο **OpenDataAccessPage** για να προσπελάσουμε μία **σελίδα πρόσβαση δεδομένων (data access page)** μέσα από την τρέχουσα διαδικασία. Η μέθοδος αυτή είναι παρόμοια με την ομώνυμη ενέργεια που παρουσιάσαμε στο κεφάλαιο διαχείρισης των μακροεντολών, και η κλήση της χαρακτηρίζεται από μία σύνταξη της μορφής

DoCmd.OpenDataAccessPage dataAccessPageName, dataAccessPageView

όπου το όρισμα **dataAccessPageName** καθορίζει το όνομα της σελίδας πρόσβασης δεδομένων που θέλουμε να χρησιμοποιήσουμε, ενώ το όρισμα **dataAccessPageView** καθορίζει την προβολή στην οποία επιθυμούμε να ανοίξουμε τη συγκεκριμένη σελίδα. Υπάρχουν δύο τιμές που μπορεί να λάβει αυτή η ιδιότητα, εκ των οποίων η **acDataPageBrowse** (που είναι και η προεπιλεγμένη τιμή) επιτρέπει την προεπισκόπηση της σελίδας πρόσβασης δεδομένων σε **προβολή περιήγησης**, ενώ η τιμή **acDataPageDesign** επιτρέπει την προεπισκόπηση της σελίδας πρόσβασης δεδομένων σε **προβολή σχεδίασης**.

Χαρακτηριστικό παράδειγμα χρήσης της μεθόδου **OpenDataAccessPage** είναι η κλήση της με τη μορφή

DoCmd.OpenDataAccessPage "Employees", acDataPageBrowse

η οποία ανοίγει τη σελίδα πρόσβασης δεδομένων που φέρει το όνομα **Employees** σε **προβολή περιήγησης**.

Άνοιγμα φόρμας (OpenForm Method): χρησιμοποιούμε τη μέθοδο **OpenForm** για να ανοίξουμε κάποια από τις φόρμες της τρέχουσας βάσης δεδομένων. Η μέθοδος αυτή είναι ανάλογη της ενέργειας **OpenForm** που περιγράψαμε στο κεφάλαιο διαχείρισης μακροεντολών, και η κλήση της χαρακτηρίζεται από μια σύνταξη της μορφής

DoCmd.OpenForm formName, formView, filterName, whereCondition, dataMode, windowMode, openArgs

Στην παραπάνω σύνταξη το μοναδικό όρισμα που πρέπει να καταχωρηθεί υποχρεωτικά, είναι το **formName**, που επιτρέπει τον καθορισμό του ονόματος της φόρμας που επιθυμούμε να χρησιμοποιήσουμε. Από την άλλη πλευρά, το όρισμα **formView** καθορίζει το είδος της προβολής στην οποία θα ανοίξουμε την εν λόγω φόρμα. Η προεπιλεγμένη τιμή για αυτό το όρισμα, είναι η **acNormal** που επιτρέπει τη χρήση της φόρμας στην ομώνυμη προβολή, ενώ μπορούν να χρησιμοποιηθούν ακόμη οι τιμές **acDesign**, **acFormDS** και **acPreview** που επιτρέπουν τη χρήση της φόρμας **σε προβολή σχεδίασης, σε προβολή φύλλου δεδομένων και σε προεπισκόπηση εκτύπωσης**.

Τα υπόλοιπα πέντε ορίσματα που μπορούν να χρησιμοποιηθούν στη σύνταξη αυτής της μεθόδου, είναι παρόμοια με εκείνα που χρησιμοποιούμε στην ενέργεια **OpenForm** που παρουσιάστηκε στο προηγούμενο κεφάλαιο. Πιο συγκεκριμένα, το όρισμα **filterName** δέχεται ως τιμή **το όνομα κάποιου ερωτήματος που θα χρησιμοποιηθεί για να περιορίσει τις εγγραφές της φόρμας**, εμφανίζοντας μόνο εκείνες που πληρούν κάποιες συγκεκριμένες ιδιότητες, ενώ το όρισμα **whereCondition** δέχεται ως τιμή μια συμβολοσειρά που περιέχει μία έγκυρη πρόταση **WHERE** της εντολής **SELECT** της γλώσσας **SQL**, χωρίς ωστόσο την ίδια τη λέξη **WHERE**. Το όρισμα **dataMode** καθορίζει **το είδος της προσπέλασης του χρήστη πάνω στα δεδομένα που συσχετίζονται με την τρέχουσα φόρμα**. Η προεπιλεγμένη τιμή για αυτό το όρισμα είναι η **acFormPropertySettings**. Εάν χρησιμοποιηθεί αυτή η τιμή, τότε το είδος της προσπέλασης επί των δεδομένων της φόρμας, είναι εκείνο που έχει καθορισθεί **κατά τη σχεδίαση της φόρμας και χρησιμοποιώντας το φύλλο ιδιοτήτων της**. Οι άλλες τιμές που μπορεί να λάβει αυτό το όρισμα, είναι οι **acFormAdd**, **acFormEdit** και **acFormReadOnly** που επιτρέπουν **την προσθήκη νέων δεδομένων, τη μεταβολή των δεδομένων που έχουν ήδη καταχωρηθεί και την προσπέλαση των δεδομένων σε συνθήκες μόνο ανάγνωσης**, αντίστοιχα. Το όρισμα **windowMode** καθορίζει **τη μορφή του παραθύρου της φόρμας**, το οποίο μπορεί να εμφανίζεται ως **πλαίσιο διαλόγου** (για την τιμή **acDialog**), ως **εικονίδιο** (για την τιμή **acIcon**), ή ακόμη και **να μην εμφανίζεται καθόλου** (για την τιμή **acHidden**). Η προεπιλεγμένη τιμή για αυτό το όρισμα είναι η **acWindowNormal** που εμφανίζει τη φόρμα στην κανονική της μορφή, έτσι όπως αυτή έχει καθορισθεί κατά το στάδιο της σχεδίασής της. Τέλος το όρισμα **openArgs** χρησιμοποιείται για τον **καθορισμό επιπρόσθετων παραμέτρων** που σε ορισμένες περιπτώσεις είναι αναγκαίο να καθορισθούνε για τη διαδικασία προβολής της τρέχουσας φόρμας. Είναι σημαντικό να αναφερθεί πως αυτό το όρισμα υπάρχει μόνο στη μέθοδο **OpenForm** και όχι στην ομώνυμη ενέργεια που χρησιμοποιείται κατά το στάδιο δημιουργίας μακροεντολών.

Τυπικό παράδειγμα χρήσης της μεθόδου **OpenForm** είναι η κλήση της με τη μορφή

DoCmd.OpenForm "Department", acFormDS, , , acFormReadOnly, acDialog

που επιτρέπει **την προεπισκόπηση της φόρμας «Department» σε προβολή φύλλου δεδομένων, σε κατάσταση μόνο ανάγνωσης και σε μορφή πλαισίου διαλόγου**.

Αντήχηση (Echo Method): χρησιμοποιούμε τη μέθοδο **Echo** για να καθορίσουμε **εάν η αλληλεπίδραση του χρήστη δια μέσου της τρέχουσας διαδικασίας, θα χαρακτηρίζεται από την ιδιότητα της αντήχησης**. Σε πλήρη αναλογία με τα όσα έχουν αναφερθεί στο προηγούμενο κεφάλαιο, η αντήχηση ορίζεται ως **η διαδικασία ενημέρωσης ή ανανέωσης της οθόνης κατά τη διάρκεια εκτέλεσης της διαδικασίας που περιέχει αυτή τη μέθοδο**. Η χρήση της μεθόδου της αντήχησης, χαρακτηρίζεται από μία σύνταξη της μορφής

DoCmd. Echo echoOn, statusBarText

όπου το όρισμα **echoOn** παίρνει μία από τις τιμές **True** ή **False** (ανάλογα με το εάν θα χρησιμοποιηθεί η αντήχηση ή όχι), ενώ το όρισμα **statusBarText** παίρνει ως τιμή μια συμβολοσειρά, το περιεχόμενο της οποίας είναι **το κείμενο που θα εμφανισθεί στη γραμμή κατάστασης της εφαρμογής (status bar)**, στην περίπτωση κατά την οποία λάβει χώρα απενεργοποίηση της αντήχησης. Χαρακτηριστικό παράδειγμα χρήσης αυτής της μεθόδου, είναι η κλήση της με τη μορφή

DoCmd.Echo False, "Visual Basic Code is Executing"

που απενεργοποιεί την αντήχηση και εμφανίζει το προαναφερόμενο μήνυμα κατά τη διάρκεια εκτέλεσης της διαδικασίας που περιέχει αυτή τη μέθοδο. Το παράδειγμα αυτό αναφέρεται στο αρχείο βοήθειας της **Microsoft Access** που περιγράφει τη χρήση αυτής της μεθόδου.

Αντιγραφή αντικειμένου (CopyObject Method): χρησιμοποιούμε τη μέθοδο **CopyObject** για να αντιγράψουμε ένα αντικείμενο μιας βάσης δεδομένων της **Microsoft Access** σε κάποια άλλη βάση. Η μέθοδος αυτή είναι εντελώς ανάλογη με την ομώνυμη ενέργεια που περιγράψαμε στο κεφάλαιο διαχείρισης των μακροεντολών, και η χρήση της χαρακτηρίζεται από μία σύνταξη της μορφής

DoCmd.CopyObject destinationDataBase, newName, sourceObjectType, sourceObjectName

Στην παραπάνω σύνταξη, το όρισμα **destinationDataBase** επιτρέπει τον καθορισμό του ονόματος της βάσης δεδομένων στην οποία επιθυμούμε να αντιγράψουμε το εν λόγω αντικείμενο, και δέχεται ως όρισμα μία συμβολοσειρά που περιέχει **το όνομα και τη διαδρομή της βάσης προορισμού**. Εάν επιθυμούμε να αντιγράψουμε το αντικείμενο στην τρέχουσα βάση, θα πρέπει να μην καταχωρήσουμε τιμή σε αυτό το όρισμα. Από την άλλη πλευρά, το όρισμα **newName** μας επιτρέπει να καθορίσουμε το νέο όνομα του αντικειμένου που θέλουμε να αντιγράψουμε. Το όρισμα αυτό χρησιμοποιείται εφόσον θέλουμε να αποθηκεύσουμε το αντίγραφο του αντικειμένου με διαφορετικό όνομα, ενώ στην αντίθετη περίπτωση, η παράμετρος αυτή δεν χρησιμοποιείται.

Το επόμενο όρισμα που περιλαμβάνεται στη σύνταξη αυτής της μεθόδου, φέρει το όνομα **sourceObjectType** και επιτρέπει τον καθορισμό του **τύπου του αντικειμένου**, επί του οποίου εφαρμόζεται αυτή η μέθοδος. Σε πλήρη αναλογία με την ενέργεια **CopyObject**, αυτό το αντικείμενο μπορεί να ανήκει σε οποιονδήποτε έγκυρο τύπο αντικειμένου της **Microsoft Access**. Ο πίνακας που ακολουθεί παρουσιάζει τις τιμές που μπορεί να λάβει η παράμετρος **sourceObjectType** και ο τύπος αντικειμένου που αντιστοιχεί σε κάθε μία από αυτές.

SourceObjectType Value	Τύπος αντικειμένου
acDataAccessPage	Σελίδα πρόσβασης δεδομένων
acDiagram	Διάγραμμα της βάσης δεδομένων
acForm	Φόρμα
acMacro	Μακροεντολή
acModule	Λειτουργική μονάδα
acQuery	Ερώτημα
acReport	Αναφορά ή έκθεση
acServerView	Προβολή διακομιστή
acStoredProcedure	Αποθηκευμένη διαδικασία
acTable	Πίνακας

Τέλος το όρισμα **sourceObjectName** δέχεται ως τιμή μια συμβολοσειρά που περιέχει **το όνομα του αντικειμένου που επιθυμούμε να αντιγράψουμε**. Εάν δεν καθορίσουμε κάποιο τύπο και κάποιο όνομα δια της χρήσης των δύο τελευταίων ορισμάτων, λαμβάνει χώρα **αντιγραφή του τρέχοντος αντικειμένου**, που βρίσκεται επιλεγμένο στο κεντρικό παράθυρο διαχείρισης της βάσης δεδομένων. Στην περίπτωση αυτή ως προεπιλεγμένη τιμή για τον τύπο αντικειμένου, χρησιμοποιείται η τιμή **acDefault**..

Χαρακτηριστικό παράδειγμα χρήσης της μεθόδου **CopyObject** είναι η κλήση της με τη μορφή

DoCmd.CopyObject "EmployeeBackUp", acTable, "Employees"

που επιτρέπει τη δημιουργία αντιγράφου του πίνακα **Employees** στην τρέχουσα βάση δεδομένων. Χρησιμοποιώντας ανάλογες εντολές, μπορούμε να δημιουργήσουμε αντίγραφα ασφαλείας, για όλα τα αντικείμενα της βάσης δεδομένων.

Αποθήκευση (Save Method): χρησιμοποιούμε τη μέθοδο **Save** για να αποθηκεύσουμε κάποιο από τα αντικείμενα της τρέχουσας βάσης δεδομένων. Εάν δεν καθορισθεί κάποιο τέτοιο αντικείμενο, η μέθοδος αποθηκεύει το ενεργό αντικείμενο της βάσης δεδομένων, δηλαδή το αντικείμενο που έχει επιλεγεί χρησιμοποιώντας τη μέθοδο **SelectObject** που θα περιγραφεί στη συνέχεια. Η κλήση της μεθόδου **Save** ακολουθεί τη σύνταξη

DoCmd.Save objectType, objectName

όπου τα ορίσματα **objectType** και **objectName** επιτρέπουν τον καθορισμό του τύπου και του ονόματος του αντικειμένου που θέλουμε να αποθηκεύσουμε. Σύμφωνα με τα όσα αναφέραμε σε προηγούμενη παράγραφο, οι επιτρεπτές τιμές που μπορεί να λάβει το όρισμα **objectType** είναι οι **acDataAccessPage** (για σελίδα πρόσβασης δεδομένων), **acDiagram** (για διάγραμμα της βάσης δεδομένων), **acForm** (για φόρμα), **acMacro** (για μακροεντολή), **acModule** (για λειτουργική μονάδα), **acQuery** (για ερώτημα), **acReport** (για αναφορά), **acServerView** (για προβολή διακομιστή), **acStoredProcedure** (για αποθηκευμένη διαδικασία) και **acTable** (για πίνακα). Εάν δεν καθορισθεί κάποιος τύπος δεδομένων χρησιμοποιείται η προεπιλεγμένη τιμή **acDefault**. Στην περίπτωση αυτή η **Microsoft Access** αποθηκεύει το τρέχον αντικείμενο της βάσης δεδομένων χρησιμοποιώντας το όνομα που έχει καθορισθεί δια της χρήσης της παραμέτρου **objectName**.

Παράδειγμα χρήσης της μεθόδου **Save**, είναι η κλήση της με τη μορφή

DoCmd.Save acTable, Employees

που αποθηκεύει τον πίνακα **«Employees»** της τρέχουσας βάσης δεδομένων.

Αποκατάσταση (Restore Method): χρησιμοποιούμε τη μέθοδο **Restore** για να επαναφέρουμε το παράθυρο του ενεργού αντικειμένου της βάσης στο φυσικό του μέγεθος, εφ όσον αυτό είναι ελαχιστοποιημένο ή μεγιστοποιημένο. Η μέθοδος καλείται χωρίς ορίσματα, και η σύνταξή της έχει τη μορφή **DoCmd.Restore**.

Αποστολή αντικειμένου (SendObject Method): χρησιμοποιούμε τη μέθοδο SendObject για να αποστείλουμε επιλεγμένα αντικείμενα της βάσης δεδομένων σε κάποιο χρήστη, δια της χρήσης της υπηρεσίας του ηλεκτρονικού ταχυδρομείου. Η κλήση αυτής της μεθόδου ακολουθεί τη σύνταξη

DoCmd.SendObject objectType, objectName, outputFormat, to, cc, bcc, subject, messageText, editMessage, templateFile

Στη σύνταξη αυτή, τα ορίσματα **objectType** και **objectName** χρησιμοποιούνται για τον καθορισμό του τύπου και του ονόματος του αντικειμένου που θέλουμε να αποστείλουμε. Πιο συγκεκριμένα, το όρισμα **objectType**, μπορεί να λάβει μία από τις τιμές **acSendDataAccessPage** (για αποστολή σελίδας πρόσβασης δεδομένων), **acSendForm** (για αποστολή φόρμας), **acSendModule** (για αποστολή λειτουργικής μονάδας), **acSendQuery** (για αποστολή ερωτήματος), **acSendReport** (για αποστολή αναφοράς) και **acSendTable** (για αποστολή πίνακα της βάσης δεδομένων). Όσον αφορά τα υπόλοιπα ορίσματα που χρησιμοποιούνται κατά τη χρήση αυτής της μεθόδου, αυτά σε γενικές γραμμές είναι τα ακόλουθα:

outputFormat: το όρισμα αυτό επιτρέπει **τον καθορισμό της μορφοποίησης με την οποία θα σταλεί το αντικείμενο της βάσης στο χρήστη – παραλήπτη**. Οι επιτρεπτές τιμές που μπορεί να λάβει αυτή η παράμετρος, είναι **acFormatDAP** (για μορφή σελίδας πρόσβασης δεδομένων), **acFormatHTML** (για μορφή σελίδας **HTML**), **acFormatRTF** (για μορφή εμπλουτισμένου κειμένου **(Rich Text Format, RTF)**), **acFormatTXT** (για μορφή απλού κειμένου) και **acFormatXLS** (για μορφή φύλλου δεδομένων του **Microsoft Excel**). Στην περίπτωση κατά την οποία δεν καθο-

ρίσουμε τιμή για αυτό το όρισμα, η **Microsoft Access** θα μας ζητήσει αυτή την πληροφορία αμέσως πριν την εκτέλεση αυτής της διαδικασίας.

to: η παράμετρος αυτή επιτρέπει τον καθορισμό **της ηλεκτρονικής διεύθυνσης του παραλήπτη** στον οποίο επιθυμούμε να αποστείλουμε το επιλεγμένο αντικείμενο της βάσης. Εάν το αντικείμενο πρόκειται να αποσταλεί σε περισσότερους από έναν παραλήπτες, μπορούμε να τοποθετήσουμε τις ηλεκτρονικές μας διευθύνσεις τη μία δίπλα στην άλλη και χωρισμένα με το χαρακτήρα «**;**». Εναλλακτικά μπορούμε αντί για τον εν λόγω χαρακτήρα, να χρησιμοποιήσουμε **το χαρακτήρα διαχωρισμού στοιχείων λίστας** που έχουμε καθορίσει χρησιμοποιώντας το εικονίδιο «**Regional Settings**» του **Πίνακα Ελέγχου (Control Panel)** του λειτουργικού συστήματος.

cc και bcc: τα ορίσματα αυτά δέχονται ως τιμή **τις ηλεκτρονικές διευθύνσεις ενός ή περισσοτέρων παραληπτών** στους οποίους επιθυμούμε να κοινοποιήσουμε το μήνυμα ηλεκτρονικού ταχυδρομείου που περιέχει το επιλεγμένο αντικείμενο της βάσης. Στα πεδία αυτά μπορούμε να καταχωρήσουμε **περισσότερες από μία διευθύνσεις**, χρησιμοποιώντας τους χαρακτήρες διαχωρισμού που παρουσιάσαμε στην προηγούμενη παράγραφο.

subject: το όρισμα αυτό δέχεται ως τιμή μια συμβολοσειρά που περιέχει **το θέμα του μηνύματος** που περιέχει το επιλεγμένο αντικείμενο της βάσης. Στην περίπτωση κατά την οποία το όρισμα αυτό δεν χρησιμοποιηθεί, λαμβάνει χώρα αποστολή του μηνύματος, χωρίς να έχει καθορισθεί κάποιο θέμα.

messageText: η χρήση του ορίσματος **messageText** επιτρέπει **την αποστολή του επιλεγμένου αντικειμένου** συνοδευόμενο από κάποιο κείμενο που περιέχει πάσης φύσεως πληροφορία (αυτή η πληροφορία μπορεί π.χ. να αφορά διευκρινήσεις σχετικά με το ρόλο και το περιεχόμενο του αντικειμένου). Εάν αυτό το όρισμα δεν χρησιμοποιηθεί, το μήνυμα προς αποστολή περιλαμβάνει μόνο το αντικείμενο της βάσης δεδομένων, και τίποτε περισσότερο.

editMessage: αυτό το όρισμα χρησιμοποιείται προκειμένου να καθορίσουμε **εάν είναι δυνατή η επεξεργασία του μηνύματος αμέσως πριν τη διαδικασία αποστολής του, ή εάν το μήνυμα θα σταλεί άμεσα χωρίς να πραγματοποιηθούν κάποιες διορθώσεις σε αυτό**. Εάν το όρισμα αυτό λάβει την τιμή **True** (που είναι και η προεπιλεγμένη τιμή), θα λάβει χώρα **εκκίνηση του προεπιλεγμένου προγράμματος διαχείρισης ηλεκτρονικής αλληλογραφίας** (π.χ. του **Microsoft Outlook**), δια της χρήσης του οποίου μπορούμε να τροποποιήσουμε το μήνυμα αμέσως πριν την αποστολή του. Στην αντίθετη περίπτωση το μήνυμα θα σταλεί άμεσα στον προορισμό του χωρίς καμία περαιτέρω διόρθωση.

templateFile: σε περίπτωση κατά την οποία το επιλεγμένο αντικείμενο θα αποσταλεί με τη μορφή **ιστοσελίδας** (**HTML** format), μπορούμε να χρησιμοποιήσουμε αυτό το όρισμα για να καθορίσουμε **το όνομα και τη διαδρομή κάποιου αρχείου προτύπου (template file)** δια της χρήσης του οποίου θα λάβει χώρα η διαμόρφωση του περιεχομένου της ιστοσελίδας.

Χαρακτηριστικό παράδειγμα χρήσης της μεθόδου **SendObject** είναι η κλήση της με τη μορφή

DoCmd.SendObject acSendReport, "Sales Report", acFormatHTML, amarg@uom.gr, , , "November's Sales Report", , False

που προκαλεί την άμεση αποστολή της αναφοράς «**Sales Reports**» σε μορφή **HTML** στον παραλήπτη **amarg@uom.gr** με θέμα μηνύματος «**November's Sales Report**».

Διαγραφή αντικειμένου (DeleteObject Method): χρησιμοποιούμε τη μέθοδο **DeleteObject** για να διαγράψουμε ένα αντικείμενο από τη βάση δεδομένων της εφαρμογής. Η μέθοδος αυτή είναι εντελώς ανάλογη με την ενέργεια **DeleteObject** που περιγράψαμε στο κεφάλαιο διαχείρισης των μακροεντολών, και χαρακτηρίζεται από μια σύνταξη της μορφής

DoCmd.DeleteObject objectType, objectName

όπου **objectType** είναι ο τύπος του αντικειμένου που επιθυμούμε να διαγράψουμε, και **objectName** το όνομα αυτού του αντικειμένου. Σε πλήρη αναλογία με παρόμοιες μεθόδους που παρουσιάσαμε στις προηγούμενες παραγράφους, ο τύπος του αντικειμένου καθορίζεται από τις τιμές **acDataAccessPage** (σελίδα πρόσβασης δεδομένων), **acDiagram** (διάγραμμα της βάσης δεδομένων), **acForm** (φόρμα), **acMacro** (μακροεντολή), **acModule** (λειτουργική μονάδα), **acQuery** (ερώτημα), **acReport** (αναφορά ή έκθεση), **acServerView** (προβολή διακομιστή), **acStoredProcedure** (αποθηκευμένη διαδικασία) και **acTable** (πίνακας). Εάν λάβει χώρα κλήση αυτής της μεθόδου χωρίς να καθορισθεί ούτε τύπος αλλά και ούτε όνομα αντικειμένου, τότε η μέθοδος αυτή θα εφαρμοσθεί **στο ενεργό αντικείμενο της βάσης**, δηλαδή στο αντικείμενο που είναι επιλεγμένο στο κεντρικό παράθυρο διαχείρισης της βάσης δεδομένων.

Χαρακτηριστικό παράδειγμα χρήσης της μεθόδου **DeleteObject** είναι η κλήση της με τη μορφή

DoCmd.DeleteObject acTable, "EMPLOYEE"

η οποία διαγράφει τον πίνακα «**EMPLOYEE**» από τη βάση δεδομένων της εφαρμογής.

Εκτέλεση εντολής SQL (RunSQL Method): χρησιμοποιούμε τη μέθοδο **RunSQL** για να εκτελέσουμε μια εντολή της γλώσσας **SQL** μέσα από την τρέχουσα διαδικασία. Ας σημειωθεί πως η μέθοδος αυτή χρησιμοποιείται μόνο για την εκτέλεση **ερωτημάτων ενέργειας (action queries)** ή **ερωτημάτων ορισμού δεδομένων (data definition queries)** και όχι για την εκτέλεση **ερωτημάτων επιλογής (select queries)** τα οποία εκτελούνται δια της χρήσης της μεθόδου **OpenQuery** που παρουσιάσαμε σε προηγούμενη ενότητα. Η κλήση της μεθόδου **RunSQL** χαρακτηρίζεται από μία σύνταξη της μορφής

DoCmd.RunSQL sqlStatement, useTransaction

όπου το όρισμα **sqlStatement** περιέχει τον κώδικα **SQL** ενός ερωτήματος ενέργειας ή ορισμού δεδομένων (τα ερωτήματα αυτού του είδους περιλαμβάνουν ενέργειες όπως είναι η **CREATE TABLE, ALTER TABLE, DROP TABLE, CREATE INDEX, DROP INDEX, INSERT INTO, DELETE, UPDATE** κ.λ.π), ενώ στο όρισμα **use-**

Transaction αποδίδεται μια από τις τιμές **True** ή **False** ανάλογα με το εάν το **query** που εκτελείται δια της χρήσης αυτής της μεθόδου, πρόκειται να χρησιμοποιηθεί ή όχι σε μία **διαδικασία συναλλαγής (transaction)**. Αυτή η διαδικασία ορίζεται ως **μία σειρά από μεταβολές που εφαρμόζονται πάνω στα δεδομένα και στο σχήμα μιας βάσης δεδομένων** και χαρακτηρίζεται από τη χρήση των δεσμευμένων λέξεων **BeginTrans** και **CommitTrans**. Η προεπιλεγμένη τιμή για αυτή την ιδιότητα είναι η τιμή **True**.

Παράδειγμα χρήσης της μεθόδου **RunSQL** είναι η κλήση της με τη μορφή

DoCmd.RunSQL "UPDATE EMPLOYEE" & _
SET SALARY=50000 WHERE SSN="123456789"

η οποία τροποποιεί τα περιεχόμενα του πίνακα **EMPLOYEE**, και πιο συγκεκριμένα αποδίδει στο πεδίο **SALARY** της εγγραφής με κωδικό **123456789** την τιμή **50000**.

Εκτέλεση μακροεντολής (RunMacro Method): χρησιμοποιούμε τη μέθοδο **RunMacro** προκειμένου να εκτελέσουμε μια **μακροεντολή** μέσα από την τρέχουσα διαδικασία. Αυτή η μέθοδος είναι εντελώς ανάλογη με την ενέργεια **RunMacro** που περιγράψαμε στο κεφάλαιο διαχείρισης μακροεντολών, και η κλήση της χαρακτηρίζεται από μία σύνταξη της μορφής

DoCmd.RunMacro macroName, repeatCount, repeatExpression

Στην παραπάνω σύνταξη, το όρισμα **macroName** είναι υποχρεωτικό, και επιτρέπει **τον καθορισμό του ονόματος της μακροεντολής** που επιθυμούμε να εκτελέσουμε. Αντίθετα, τα ορίσματα **repeatCount** και **repeatExpression** είναι προαιρετικά, και χρησιμοποιούνται προκειμένου να καθορίσουμε **τον αριθμό επαναλήψεων εκτέλεσης αυτής της μακροεντολής**. Εάν δεν χρησιμοποιήσουμε αυτά τα ορίσματα, η μακροεντολή θα εκτελεστεί **μόνο μία φορά**. Εάν καταχωρήσουμε μια ακέραια τιμή **N** στο όρισμα **repeatCount**, τότε θα λάβει χώρα εκτέλεση της μακροεντολής **N φορές**. Τέλος η παράμετρος **repeatExpression** δέχεται ως όρισμα μία συμβολοσειρά που περιέχει **μία λογική έκφραση**, δηλαδή μία έκφραση που μπορεί να λάβει μία από τις τιμές **True** ή False. Η τιμή αυτής της έκφρασης υπολογίζεται **σε κάθε κύκλο εκτέλεσης της μακροεντολής**, και εφόσον είναι αληθής (True), η μακροεντολή θα εκτελεστεί και στην επόμενη επανάληψη, ενώ στην αντίθετη περίπτωση θα λάβει χώρα τερματισμός της εκτέλεσής της.

Τυπικό παράδειγμα χρήσης της μεθόδου **RunMacro** είναι η κλήση της με τη μορφή

DoCmd.RunMacro "Print Employee List", 3

η οποία προκαλεί **την εκτύπωση της λίστας των υπαλλήλων της εταιρείας σε τρία αντίτυπα**. Η εκτύπωση αυτής της λίστας γίνεται από τη μακροεντολή **«Print Employee List»**, η οποία στην προκειμένη περίπτωση εκτελείται τρεις φορές.

Εκτύπωση (PrintOut Method): χρησιμοποιούμε τη μέθοδο **PrintOut** για να εκτυπώσουμε τα περιεχόμενα του ενεργού αντικειμένου της τρέχουσας βάσης δεδο-

μένων. Αυτή η μέθοδος είναι εντελώς ανάλογη με την ενέργεια **PrintOut** που περιγράψαμε στο κεφάλαιο διαχείρισης των μακροεντολών, και η κλήση της ακολουθεί μία σύνταξη της μορφής

DoCmd.PrintOut printRange, pageFrom, pageTo, printQuality, copies, collateCopies

Στην παραπάνω σύνταξη, το όρισμα **printRange** επιτρέπει **τον καθορισμό της περιοχής σελίδων που επιθυμούμε να εκτυπώσουμε**. Η παράμετρος αυτή παίρνει μία από τις τιμές **acPrintAll** που επιτρέπει την εκτύπωση **του συνόλου των σελίδων** (η τιμή αυτή είναι η προεπιλεγμένη), **acSelection** που επιτρέπει την εκτύπωση **μόνο του τμήματος του αντικειμένου που έχει επιλεγεί** (με το **πληκτρολόγιο** ή με το **ποντίκι**) και **acPages** που επιτρέπει την εκτύπωση **συγκεκριμένης περιοχής σελίδων**. Στην τελευταία περίπτωση μπορούμε να καθορίσουμε την **αρχική** και την τελική **σελίδα** εκτύπωσης, χρησιμοποιώντας τα δύο επόμενα ορίσματα, που φέρουν τα ονόματα **pageFrom** και **pageTo**. Εάν για παράδειγμα επιθυμούμε να εκτυπώσουμε τις σελίδες **5,6,7** και **8** μιας αναφοράς, θα αποδώσουμε σε αυτές τις δύο παραμέτρους τις τιμές **pageFrom=5** και **pageTo=8**

Από την άλλη πλευρά, το όρισμα **printQuality**, επιτρέπει τον καθορισμό **της ποιότητας εκτύπωσης**. Οι τιμές που μπορεί να πάρει αυτό το όρισμα είναι **acDraft** (πρόχειρη εκτύπωση), **acLow** (εκτύπωση χαμηλής ποιότητας), **acMedium** (εκτύπωσης μεσαίας ποιότητας) και **acHigh** (εκτύπωση υψηλής ποιότητας). Η τελευταία τιμή είναι και η προεπιλεγμένη τιμή για την παράμετρο **printQuality**. Εάν επιθυμούμε να εκτυπώσουμε τα περιεχόμενα του επιλεγμένου αντικειμένου **περισσότερες από μία φορές**, καθορίζουμε τον αριθμό των αντιτύπων δια της χρήσης της παραμέτρου **copies**. Τέλος η παράμετρος **collateCopies** δέχεται μία από τις τιμές **True** ή **False** ανάλογα με τον εάν επιθυμούμε ή όχι να χρησιμοποιήσουμε το χαρακτηριστικό της **συρραφής αντιτύπων**. Περισσότερες λεπτομέρειες σχετικά με αυτό το χαρακτηριστικό εκτύπωσης, μπορούν να βρεθούνε στην τεκμηρίωση της ενέργειας **PrintOut** όπως αυτή παρουσιάζεται στο κεφάλαιο διαχείρισης των μακροεντολών.

Παράδειγμα χρήσης της μεθόδου **PrintOut** είναι η κλήση της με τη μορφή

DoCmd.PrintOut acPages, 1, 10, acDraft, 5, True

που έχει ως αποτέλεσμα **την πρόχειρη εκτύπωση 5 αντιτύπων των σελίδων 1 έως 10 του επιλεγμένου αντικειμένου** χρησιμοποιώντας το χαρακτηριστικό της **συρραφής αντιτύπων**.

Ελαχιστοποίηση (Minimize Method): χρησιμοποιούμε τη μέθοδο **Minimize** για να ελαχιστοποιήσουμε το **ενεργό πλαίσιο διαλόγου**. Μετά την ελαχιστοποίηση, αυτό το πλαίσιο εμφανίζεται ως μία **μικρή γραμμή τίτλου (title bar)** στο κάτω μέρος της οθόνης. Η μέθοδος αυτή καλείται χωρίς ορίσματα, και η κλήση της ακολουθεί τη σύνταξη **DoCmd.Minimize**.

Εμφάνιση γραμμής εργαλείων (ShowToolbar Method): χρησιμοποιούμε τη μέθοδο **ShowToolbar** για να εμφανίσουμε ή να αποκρύψουμε μια γραμμή εργαλείων από την οθόνη του υπολογιστή μας. Αυτή η μέθοδος είναι εντελώς ανάλογη με την

ενέργεια **ShowToolbar** που περιγράψαμε στο κεφάλαιο διαχείρισης των μακροεντολών, και η κλήση της χαρακτηρίζεται από μία σύνταξη της μορφής

DoCmd. ShowToolbar toolbarName, show

Στην παραπάνω σύνταξη, το όρισμα **toolbarName** δέχεται ως τιμή μια συμβολοσειρά που περιέχει **το όνομα της γραμμής εργαλείων** που επιθυμούμε να εμφανίσουμε ή να αποκρύψουμε. Από την άλλη πλευρά, το όρισμα **show** επιτρέπει **τον καθορισμό της διαδικασίας που πρόκειται να εφαρμοσθεί πάνω στη γραμμή εργαλείων**. Οι τιμές που μπορεί να λάβει αυτή η παράμετρος, είναι οι **acToolbarYes** που προκαλεί την εμφάνιση της γραμμής εργαλείων (η τιμή αυτή είναι και η προεπιλεγμένη), **acToolbarNo** (που προκαλεί την απόκρυψη της γραμμής εργαλείων) και **acToolbarWhenApprop** που έχει ως αποτέλεσμα την εμφάνιση της γραμμής εργαλείων αλλά μόνο όπου αυτό είναι αναγκαίο.

Παράδειγμα χρήσης της μεθόδου **ShowToolbar**, είναι η κλήση της με τη μορφή

DoCmd.ShowToolbar "EmployeeManagement", acToolbarYes

που προκαλεί την εμφάνιση της γραμμής εργαλείων που φέρει το όνομα **EmployeeManagement**.

Ενημέρωση αντικειμένου (RepaintObject Method): χρησιμοποιούμε τη μέθοδο **RepaintObject** για να επανασχεδιάσουμε την περιοχή της οθόνης που καταλαμβάνεται από το πλαίσιο διαλόγου κάποιου από τα αντικείμενα της βάσης δεδομένων, στην περίπτωση κατά την οποία λάβει χώρα **μεταβολή του μεγέθους του** (**μεγιστοποίηση**, **ελαχιστοποίηση** ή **επαναφορά**). Η κλήση αυτής της μεθόδου ακολουθεί τη σύνταξη

DoCmd.RepaintObject objectType, objectName

όπου **objectName** είναι το όνομα του αντικειμένου και **objectType** ο τύπος του. Σε πλήρη αναλογία με παρόμοιες μεθόδους οι οποίες εφαρμόζονται πάνω στα αντικείμενα της βάσης δεδομένων, οι τιμές που μπορεί να πάρει το όρισμα **objectType** είναι **acDataAccessPage** (σελίδα πρόσβασης δεδομένων), **acDiagram** (διάγραμμα της βάσης δεδομένων), **acForm** (φόρμα), **acMacro** (μακροεντολή), **acModule** (λειτουργική μονάδα), **acQuery** (ερώτημα), **acReport** (αναφορά ή έκθεση), **acServerView** (προβολή διακομιστή), **acStoredProcedure** (αποθηκευμένη διαδικασία) και **acTable** (πίνακας). Παράδειγμα χρήσης της μεθόδου **acRepaintObject** είναι η κλήση της με τη μορφή

DoCmd.RepaintObject actable, "Employees"

που επανασχεδιάζει το παράθυρο που συσχετίζεται με τον πίνακα **Employees** της βάσης δεδομένων.

Έξοδος (Quit Method): χρησιμοποιούμε τη μέθοδο **Quit** για να τερματίσουμε τη λειτουργία της **Microsoft Access**. Αυτή η μέθοδος είναι εντελώς ανάλογη με

την ενέργεια **Quit** που περιγράψαμε στο κεφάλαιο διαχείρισης μακροεντολών, και η κλήση της χαρακτηρίζεται από μία σύνταξη της μορφής

DoCmd.Quit Options

όπου το όρισμα **options** καθορίζει **εάν θα λάβει χώρα αποθήκευση της βάσης δεδομένων πριν τον τερματισμό της λειτουργίας της εφαρμογής**. Οι τιμές που μπορεί να λάβει αυτή η παράμετρος είναι οι **acQuitPrompt** (που προκαλεί την εμφάνιση ενός **πλαισίου διαλόγου** δια της χρήσης του οποίου ο χρήστης καθορίζει εάν επιθυμεί να αποθηκεύσει ή όχι τη βάση δεδομένων), **acQuitSaveAll** (που προκαλεί την αυτόματη αποθήκευση όλων των αντικειμένων της εφαρμογής) και **acQuitSaveNone** (που τερματίζει τη λειτουργία της **Microsoft Access** χωρίς να αποθηκεύσει τις αλλαγές που ενδεχομένως έχουν πραγματοποιηθεί πάνω στη βάση δεδομένων). Εάν δεν καθορισθεί κάποια τιμή, η προεπιλεγμένη τιμή που χρησιμοποιείται, είναι η **acQuitSaveAll**. Παράδειγμα χρήσης της μεθόδου **Quit** είναι η κλήση της με τη μορφή

DoCmd.Quit acQuitPrompt

που τερματίζει τη λειτουργία της **Microsoft Access**, ερωτώντας το χρήστη εάν επιθυμεί να αποθηκεύσει ή όχι την τρέχουσα βάση δεδομένων.

Έξοδος σε (OutputTo Method): χρησιμοποιούμε τη μέθοδο **OutputTo** για να αποθηκεύσουμε τα περιεχόμενα των αντικειμένων της βάσης δεδομένων σε αρχεία συγκεκριμένου τύπου, όπως είναι **λογιστικά φύλλα του Microsoft Excel** ή **αρχεία κειμένου του MS-DOS**. Η μέθοδος αυτή είναι παρόμοια με την ενέργεια **OutputTo** που περιγράψαμε στο κεφάλαιο διαχείρισης μακροεντολών, και η κλήση της χαρακτηρίζεται από μία σύνταξη της μορφής

DoCmd.OutputTo objectType, objectName, outputFormat, outputFile, autoStart, templateFile

Στην παραπάνω σύνταξη, τα δύο πρώτα ορίσματα επιτρέπουν τον καθορισμό του **τύπου** και του **ονόματος** του αντικειμένου τα περιεχόμενα του οποίου επιθυμούμε να αποθηκεύσουμε σε αρχείο. Το όνομα του αντικειμένου αποθηκεύεται στη μεταβλητή **objectName**, ενώ ο τύπος που το περιγράφει, καθορίζεται δια της καταχώρησης στην παράμετρο **objectType** κάποιας από τις τιμές **acOutputDataAccessPage** (σελίδα πρόσβασης δεδομένων), **acOutputForm** (φόρμα), **acOutputModule** (λειτουργική μονάδα), **acOutputQuery** (ερώτημα), **acOutputReport** (αναφορά), **acOutputServerView** (προβολή διακομιστή), **acOutputStoredProcedure** (αποθηκευμένη διαδικασία) και **acOutputTable** (πίνακας της βάσης δεδομένων).

Από την άλλη πλευρά, το όρισμα **outputFormat** δέχεται ως τιμή μια σταθερά που καθορίζει **τον τύπο του αρχείου** στο οποίο θα αποθηκεύσουμε τα περιεχόμενα του επιλεγμένου αντικειμένου. Η σταθερά αυτή μπορεί να λάβει μία από τις τιμές **acFormatASP** (για Active Server Page Format), **acFormatDAP** (για Data Access Page Format), **acFormatHTML** (για HTML Format), **acFormatIIS** (για Internet Information Server Format), **acFormatRTF** (για Rich Text Format), **acFormatTXT** (για Text Format) και **acFormatXLS** (για XLS Format). Όσον αφορά το όνομα και τη διαδρομή του αρχείου εξόδου, αυτά καθορίζονται χρησιμοποιώντας την παράμε-

τρο **outputFile**, ενώ στην περίπτωση κατά την οποία ο τύπος του αρχείου εξόδου είναι **HTML**, μπορούμε χρησιμοποιώντας την παράμετρο **templateFile** να καθορίσουμε το όνομα και τη διαδρομή ενός **αρχείου προτύπου**, με τη βοήθεια του οποίου θα λάβει χώρα η διαμόρφωση του περιεχομένου του αρχείου εξόδου. Η τελευταία παράμετρος που μπορούμε να καθορίσουμε στην κλήση της μεθόδου **OutputTo**, είναι η **autoStart**. Εάν αυτή η παράμετρος λάβει την τιμή «**True**», λαμβάνει χώρα **άμεση εκκίνηση της εφαρμογής που διαχειρίζεται το συγκεκριμένο τύπου του αρχείου εξόδου**. Εάν για παράδειγμα καθορίσουμε ως αρχείο εξόδου ένα **φύλλο δεδομένων (XLS Format)**, η απόδοση της τιμής «**True**» στη μεταβλητή **autoStart**, θα έχει ως αποτέλεσμα την εκκίνηση του **Microsoft Excel** αμέσως μετά την ολοκλήρωση της διαδικασίας έτσι ώστε να προχωρήσουμε σε περαιτέρω επεξεργασία του αρχείου εξόδου (εάν βέβαια κάτι τέτοιο είναι επιθυμητό).

Τυπικό παράδειγμα χρήσης της μεθόδου **OutputTo**, είναι η κλήση της με τη μορφή

DoCmd.OutputTo, acOutputTable, "WorksOn",
acFormatHTML, "Report.htm"

η οποία αποθηκεύει τα περιεχόμενα του πίνακα «**WorksOn**» ως ιστοσελίδα και κάτω από το όνομα «**Report.html**».

Επανεκτέλεση ερωτήματος (Requery Method): χρησιμοποιούμε τη μέθοδο **Requery** για να ανανεώσουμε τα περιεχόμενα ενός στοιχείου ελέγχου, στην περίπτωση κατά την οποία τα περιεχόμενα του αντικειμένου από το οποίο αυτό παίρνει δεδομένα, έχουν μεταβληθεί. Αυτή η μέθοδος είναι παρόμοια με την ενέργεια **Requery** που περιγράψαμε στο κεφάλαιο διαχείρισης των μακροεντολών, και η κλήση της χαρακτηρίζεται από μία σύνταξη της μορφής

DoCmd.Requery controlName

όπου **controlName** είναι το όνομα του στοιχείου ελέγχου τα περιεχόμενα του οποίου επιθυμούμε να ανανεώσουμε. Χαρακτηριστικό παράδειγμα χρήσης αυτής της μεθόδου είναι η κλήση της με τη μορφή

DoCmd.Requery "EmployeeList"

όπου το στοιχείο ελέγχου **«EmployeeList»** είναι ένα **πλαίσιο λίστας (list box)** που εμφανίζει τα προσωπικά στοιχεία των υπαλλήλων που εργάζονται στην εταιρεία. Στην περίπτωση κατά την οποία λάβει χώρα τροποποίηση των περιεχομένων του πίνακα **EMPLOYEE** δια της προσθήκης ή της διαγραφής κάποιων εγγραφών, θα πρέπει να χρησιμοποιήσουμε αυτή τη μέθοδο **για να ανανεώσουμε τα περιεχόμενα του πλαισίου λίστας, έτσι ώστε αυτό σε κάθε περίπτωση να εμφανίζει ενημερωμένα δεδομένα**.

Επιλογή αντικειμένου (SelectObject Method): χρησιμοποιούμε τη μέθοδο **SelectObject** για να επιλέξουμε κάποιο από τα αντικείμενα της τρέχουσας βάσης δεδομένων. Η μέθοδος αυτή είναι παρόμοια με την ομώνυμη ενέργεια που περιγρά-

ψαμε στο κεφάλαιο διαχείρισης των μακροεντολών και η κλήση της χαρακτηρίζεται από μία σύνταξη της μορφής

DoCmd.SelectObject objectType, objectName, inDatabaseWindow

Στην παραπάνω σύνταξη, το όρισμα **objectName** επιτρέπει τον καθορισμό του ονόματος του αντικειμένου που θέλουμε να επιλέξουμε, ενώ ο τύπος αυτού του αντικειμένου καθορίζεται δια της καταχώρησης στην παράμετρο **objectType** μίας εκ των τιμών **acDataAccessPage** (σελίδα πρόσβασης δεδομένων), **acDiagram** (διάγραμμα της βάσης δεδομένων), **acForm** (φόρμα), **acMacro** (μακροεντολή), acModule (λειτουργική μονάδα), **acQuery** (ερώτημα), **acReport** (αναφορά ή έκθεση), **acServerView** (προβολή διακομιστή), **acStoredProcedure** (αποθηκευμένη διαδικασία) και **acTable** (πίνακας της βάσης δεδομένων). Τέλος το όρισμα **inDatabaseWindow** δέχεται μία από τις τιμές **True** ή **False** και καθορίζει **εάν το αντικείμενο θα επιλεγεί ή όχι στο κεντρικό παράθυρο διαχείρισης της βάσης δεδομένων**. Σύμφωνα με τα όσα έχουμε αναφέρει στην παρουσίαση της ομώνυμης ενέργειας, η χρήση της τιμής **False** σε αυτή την παράμετρο, είναι επιτρεπτή, **μόνο όταν το αντικείμενο που θέλουμε να χρησιμοποιήσουμε έχει ήδη ανοίξει σε προηγούμενο στάδιο της διαδικασίας**. Στην αντίθετη περίπτωση η **Microsoft Access** θα εμφανίσει ένα μήνυμα λάθους.

Χαρακτηριστικό παράδειγμα χρήσης της μεθόδου **SelectObject** είναι η κλήση της με τη μορφή

DoCmd.SelectObject acForm, "Department"

που επιτρέπει την επιλογή της φόρμας **«Department»** ως το ενεργό αντικείμενο της βάσης δεδομένων.

Επισημάνσεις (SetWarnings Method): χρησιμοποιούμε τη μέθοδο **SetWarnings** για να ενεργοποιήσουμε ή να απενεργοποιήσουμε την εμφάνιση **προειδοποιητικών μηνυμάτων (warnings)** κατά τη διάρκεια εκτέλεσης μιας διαδικασίας. Η κλήση αυτής της μεθόδου χαρακτηρίζεται από μία σύνταξη της μορφής

DoCmd.SetWarnings warningsOn

όπου το όρισμα **warningsOn** δέχεται μία από τις τιμές **True** ή **False** που ενεργοποιούν ή απενεργοποιούν αντίστοιχα την εμφάνιση των προειδοποιητικών μηνυμάτων. Τυπικό παράδειγμα χρήσης αυτής της μεθόδου, είναι η κλήση της με τη μορφή

DoCmd.SetWarnings False

η οποία αποτρέπει την εμφάνιση αυτών των μηνυμάτων κατά τη διάρκεια εκτέλεσης της διαδικασίας που περιέχει αυτή τη μέθοδο.

Εύρεση εγγραφής (FindRecord Method): χρησιμοποιούμε τη μέθοδο **FindRecord** για να αναζητήσουμε την πρώτη εκ των εγγραφών του ενεργού αντικειμένου της βάσης δεδομένων που χαρακτηρίζεται από κάποια κριτήρια. Αυτή η μέθοδος είναι παρόμοια με την ομώνυμη ενέργεια που περιγράψαμε στο κεφάλαιο διαχείρισης των μακροεντολών, και η κλήση της χαρακτηρίζεται από μία σύνταξη της μορφής

DoCmd.FindRecord findWhat, match, matchCase, search, searchFormatted, onlyCurrentField. findFirst

Στην παραπάνω σύνταξη, το όρισμα **findWhat** δέχεται ως τιμή μια συμβολοσειρά που περιέχει μία έκφραση η οποία καθορίζει και **το κριτήριο αναζήτησης της εγγραφής του ενεργού αντικειμένου**. Εάν για παράδειγμα επιθυμούμε να αναζητήσουμε την εγγραφή του πίνακα **EMPLOYEE** που αναφέρεται στον εργαζόμενο με επώνυμο **Smith**, θα καταχωρήσουμε ως τιμή σε αυτή την παράμετρο, τη συμβολοσειρά «**Smith**». Εναλλακτικά μπορούμε εκτός από συμβολοσειρές να τοποθετήσουμε στην έκφραση αναζήτησης, και άλλους τύπους δεδομένων, όπως είναι για παράδειγμα αριθμούς και ημερομηνίες.

Από την άλλη πλευρά, το όρισμα **match** επιτρέπει τον καθορισμό **του τρόπου αναζήτησης του δεδομένου που έχει καθοριστεί**. Αυτό το όρισμα παίρνει μία από τις τιμές **acAnyWhere**, **acEntire** και **acStart**. Χρησιμοποιώντας την τιμή **acAnyWhere** θεωρούμε πως το δεδομένο αναζήτησης **αποτελεί τμήμα κάποιου πεδίου** ενώ η τιμή **acEntire** (που είναι και η προεπιλεγμένη) διαπραγματεύεται το δεδομένο αναζήτησης **ως ολόκληρη την τιμή του πεδίου**. Τέλος η χρήση της τιμής **acStart** θεωρεί πως το δεδομένο αναζήτησης αποτελεί μόνο πρόθεμα των τιμών των πεδίων του πίνακα, και επομένως η διαδικασία αναζήτησης θα επιστρέψει μόνο τις εγγραφές που χαρακτηρίζονται από αυτή την ιδιότητα.

Για να κατανοήσουμε καλύτερα αυτή τη διαδικασία, ας θεωρήσουμε πως ο πίνακας **EMPLOYEE** περιέχει δύο εργαζόμενους με επώνυμα **James** και **Jameson**. Εάν καταχωρήσουμε στο όρισμα **findWhat** τη συμβολοσειρά **James** και αποδώσουμε στην παράμετρο **match** την τιμή **acAnyWhere** η διαδικασία αναζήτησης θα επιστρέψει και τις δύο εγγραφές, καθώς θα αναζητήσει τη συμβολοσειρά **James** σε οποιοδήποτε τμήμα των επωνύμων που έχουν καταχωρηθεί στο πεδίο LNAME του πίνακα **EMPLOYEE**. Αντίθετα η χρήση της τιμής **acEntire** θα επιστρέψει μόνο την εγγραφή στην οποία περιέχεται το επώνυμο **James** καθώς στην προκειμένη περίπτωση η συμβολοσειρά αναζήτησης θα θεωρηθεί πως αποτελεί **ολόκληρη την τιμή του πεδίου, και όχι κάποιο τμήμα του**. Εάν επιθυμούμε η διαδικασία αναζήτησης **να κάνει διάκριση ανάμεσα στα μικρά και στα κεφαλαία γράμματα (case sensitivity)**, θα πρέπει να χρησιμοποιήσουμε το επόμενο όρισμα που φέρει το όνομα **matchCase**, και να αποδώσουμε σε αυτό την τιμή **True**. Αντίθετα, εφόσον το όρισμα αυτό λάβει την τιμή **False**, η αναζήτηση θα πραγματοποιηθεί, χωρίς να λάβει χώρα αυτή η διάκριση.

Το επόμενο πεδίο που μπορούμε να καθορίσουμε στη σύνταξη αυτής της μεθόδου, φέρει το όνομα **search** και επιτρέπει **τον καθορισμό του συνόλου των εγγραφών στις οποίες θα λάβει χώρα η αναζήτηση του καθορισμένου δεδομένου**. Αυτό το όρισμα μπορεί να λάβει μία από τις τιμές **acDown**, **acUp** και **acSearchAll**. Εάν χρησιμοποιήσουμε την τιμή **acDown**, η διαδικασία αναζήτησης θα περιορισθεί μόνο σε εκείνες τις εγγραφές που βρίσκονται κάτω από την τρέχουσα εγγραφή, ενώ η χρήση της τιμής **acUp** θα περιορίσει την αναζήτηση μόνο στις εγγραφές του ενεργού αντικειμένου που βρίσκονται πάνω από την τρέχουσα εγγραφή. Εάν επιθυμούμε να επεκτείνουμε την αναζήτηση σε όλες τις εγγραφές του ενεργού αντικειμένου, ανεξάρτητα από τη θέση της τρέχουσας εγγραφής, θα πρέπει να αποδώσουμε σε αυτή την

παράμετρο την τιμή **acSearchAll** που είναι άλλωστε και η προεπιλεγμένη τιμή για αυτό το όρισμα.

Σε περιπτώσεις κατά τις οποίες τα δεδομένα που βρίσκονται αποθηκευμένα στους πίνακες της βάσης χαρακτηρίζονται από κάποιο τύπο **διαμόρφωσης**, μπορούμε να χρησιμοποιήσουμε την επόμενη παράμετρο (που φέρει το όνομα **searchFormatted**) για να καθορίσουμε **εάν η διαδικασία αναζήτησης θα πραγματοποιηθεί λαμβάνοντας υπ' όψιν ή όχι αυτή τη διαμόρφωση**. Στην περίπτωση κατά την οποία επιθυμούμε να ληφθεί υπ' όψιν αυτή η διαμόρφωση θα πρέπει να αποδώσουμε στην παράμετρο αυτή την τιμή **True**, ενώ στην αντίθετη περίπτωση θα πρέπει να χρησιμοποιήσουμε την τιμή **False**. Προκειμένου να γίνει κατανοητή η χρήση αυτής της παραμέτρου, ο αναγνώστης παραπέμπεται στο παράδειγμα που περιλαμβάνεται στην τεκμηρίωση της ομώνυμης ενέργειας στο κεφάλαιο διαχείρισης των μακροεντολών της εφαρμογής.

Από την άλλη πλευρά, το όρισμα **onlyCurrentField**, παίρνει μία από τις τιμές **acAll** και **acCurrent** και καθορίζει **εάν η αναζήτηση του καθορισμένου δεδομένου θα περιοριστεί μόνο στο αντίστοιχο πεδίο των εγγραφών του ενεργού αντικειμένου, ή θα επεκταθεί σε όλα τα πεδία αυτών των εγγραφών**. Εάν για παράδειγμα αναζητούμε τον υπάλληλο της εταιρείας με το επώνυμο **Smith**, η χρήση της τιμής **acCurrent** στην παράμετρο **onlyCurrentField** θα περιορίζει τη διαδικασία αναζήτησης μόνο στις τιμές της στήλης **LNAME** του πίνακα **EMPLOYEE**. Αντίθετα η χρήση της τιμής **acAll** θα επιστρέψει τις εγγραφές του εν λόγω πίνακα που περιέχουν τη συμβολοσειρά **Smith** σε οποιοδήποτε από τα πεδία τους.

Τέλος το πεδίο **findFirst** παίρνει κάποια από τις τιμές **True** ή **False** και καθορίζει **εάν η διαδικασία αναζήτησης θα ξεκινήσει από την πρώτη εγγραφή του ενεργού αντικειμένου ή από την εγγραφή εκείνη που βρίσκεται αμέσως μετά την τρέχουσα εγγραφή**. Η προεπιλεγμένη τιμή για αυτή την παράμετρο είναι η τιμή **True** που ξεκινά τη διαδικασία αναζήτησης από την πρώτη εγγραφή του ενεργού αντικειμένου.

Τυπικό παράδειγμα χρήσης της μεθόδου **FindRecord** είναι η κλήση της με τη μορφή

DoCmd.FindRecord Smith, acAnyWhere, True, acDown, False, True, True

που επιστρέφει τις εγγραφές του ενεργού αντικειμένου που βρίσκονται μετά την τρέχουσα εγγραφή η οποίες περιέχουν τη συμβολοσειρά **Smith** σε οποιοδήποτε τμήμα της τιμής του πεδίου τους.

Εύρεση επομένου (FindNext Method): χρησιμοποιούμε τη μέθοδο **FindNext** για να αναζητήσουμε την αμέσως επόμενη εγγραφή του ενεργού αντικειμένου που πληροί τα κριτήρια αναζήτησης που έχουν καθορισθεί δια της χρήσης της μεθόδου **FindRecord**. Η μέθοδος αυτή καλείται χωρίς ορίσματα και η κλήση της χαρακτηρίζεται από τη σύνταξη **DoCmd.FindNext**.

Εφαρμογή φίλτρου (ApplyFilter Method): χρησιμοποιούμε τη μέθοδο **ApplyFilter** για να εφαρμόσουμε **ένα φίλτρο, ένα ερώτημα ή μία πρόταση WHERE**

της εντολής SELECT της γλώσσας SQL πάνω στα δεδομένα ενός πίνακα, μιας φόρμας ή ενός ερωτήματος. Αυτή η μέθοδος είναι παρόμοια με την ομώνυμη ενέργεια που παρουσιάσαμε στο κεφάλαιο διαχείρισης των μακροεντολών, και η κλήση της χαρακτηρίζεται από μία σύνταξη της μορφής

DoCmd.ApplyFilter filterName, whereCondition, filterType

Στην παραπάνω σύνταξη, το όρισμα **filterName** δέχεται ως τιμή μια συμβολοσειρά που περιέχει το όνομα κάποιου φίλτρου ή ερωτήματος της βάσης δεδομένων ενώ η τιμή που καταχωρούμε στο όρισμα **whereCondition** περιέχει μία πρόταση **WHERE** της εντολής **SELECT** της γλώσσας **SQL** η οποία ωστόσο δεν θα πρέπει να περιλαμβάνει την ίδια τη λέξη **WHERE** (η οποία υπονοείται). Τέλος το όρισμα **filterType** δέχεται μία από τις τιμές **acFilterNormal** και **acServerFilter** που καθορίζουν εάν η διαδικασία αναζήτησης θα επεκταθεί και στα δεδομένα τα οποία βρίσκονται αποθηκευμένα στους πίνακες της βάσης αλλά η διαμόρφωσή τους είναι διαφορετική από εκείνη με την οποία εμφανίζονται στην οθόνη. Η προεπιλεγμένη τιμή για αυτή την παράμετρο είναι η **acFilterNormal** που περιορίζει τη διαδικασία αναζήτησης **μόνο σε εκείνα τα δεδομένα που ακολουθούν τον ίδιο τρόπο διαμόρφωσης** τόσο για την εμφάνισή τους στην οθόνη όσο και για την αποθήκευσή τους στη βάση δεδομένων της εφαρμογής.

Παράδειγμα χρήσης της μεθόδου **ApplyFilter** είναι η κλήση της με τη μορφή

DoCmd.ApplyFilter FName="John"

που εμφανίζει μόνο τις εγγραφές του ενεργού αντικειμένου για τις οποίες το πεδίο **LName** έχει την τιμή «**John**». Είναι σημαντικό να αναφερθεί στο σημείο αυτό, πως πριν από τη μέθοδο **ApplyFilter** θα πρέπει να χρησιμοποιήσουμε τη μέθοδο **Select Object** προκειμένου να καθορίσουμε το ενεργό αντικείμενο της βάσης.

Ηχητικό σήμα (Beep Method): χρησιμοποιούμε τη μέθοδο **Beep** για να αναπαράγουμε ένα **ηχητικό σήμα** δια της χρήσης του **μεγαφώνου (speaker)** του υπολογιστή. Η μέθοδος καλείται χωρίς ορίσματα και η σύνταξή της έχει τη μορφή **DoCmd.Beep**. Η μέθοδος αυτή είναι ιδιαίτερα χρήσιμη σε περιπτώσεις κατά τις οποίες επιθυμούμε να υποδηλώσουμε **την ολοκλήρωση μιας διαδικασίας ή την πραγματοποίηση κάποιου σφάλματος**.

Κατάσταση στοιχείου menu (SetMenuItem Method): χρησιμοποιούμε τη μέθοδο **SetMenuItem** για να καθορίσουμε την κατάσταση των στοιχείων ενός **menu επιλογών**. Αυτή η μέθοδος είναι παρόμοια με την ομώνυμη ενέργεια που περιγράψαμε στο κεφάλαιο διαχείρισης των μακροεντολών, και η κλήση της χαρακτηρίζεται από μια σύνταξη της μορφής

DoCmd.SetMenuItem menuIndex, commanIndex, subCommandIndex, flag

Στην παραπάνω σύνταξη, οι παράμετροι **menuIndex**, **commandIndex** και **subCommandIndex** επιτρέπουν τον καθορισμό **της θέσης της εντολής** την κατάσταση της οποίας επιθυμούμε να μεταβάλλουμε. Οι τιμές που καταχωρούμε σε αυτές τις παραμέτρους είναι **ακέραιοι αριθμοί** που ξεκινούν από το **μηδέν** και καθορίζουν

πλήρως τη θέση της εντολής μέσα στο menu. Για παράδειγμα εάν αποδώσουμε στις τιμές **menuIndex** και **commandIndex** τις τιμές **1** και **4**, αναφερόμαστε **στην πέμπτη εντολή της δεύτερης ομάδας επιλογών του menu του ενεργού παραθύρου**. Εάν το στοιχείο του **menu** που καθορίζουμε δια της χρήσης αυτών των δύο τιμών δεν αντιστοιχεί σε εντολή αλλά σε menu επιλογών χαμηλότερου επιπέδου, μπορούμε να χρησιμοποιήσουμε το τρίτο όρισμα (**subCommandIndex**) για να καθορίσουμε **τη θέση της εντολής μέσα σε αυτό το menu**. Τέλος, η κατάσταση στην οποία θέλουμε να μεταφέρουμε την επιλογή που έχουμε καθορίσει, χρησιμοποιούμε το τέταρτο όρισμα. Το όρισμα αυτό παίρνει μία από τις τιμές **acMenuCheck** (επιλογή στοιχείου), **acMenuGray** (απενεργοποίηση στοιχείου), **acMenuUncheck** (κατάργηση επιλογής στοιχείου) και **acMenuUngray** (ενεργοποίηση στοιχείου). Η επιλογή ενός στοιχείου menu γίνεται τοποθετώντας δίπλα στο όνομά του το σύμβολο «√». Θα πρέπει να σημειωθεί, πως αυτή η ενέργεια μπορεί να εφαρμοσθεί **μόνο στα προσαρμοσμένα menus (custom menus)** που δημιουργούνται προκειμένου να καλύψουν τις ανάγκες του χρήστη, και όχι στα **προεπιλεγμένα menus (default menus)** της **Microsoft Access**.

Παράδειγμα χρήσης της μεθόδου **SetMenuItem** είναι η κλήση της με τη μορφή

DoCmd.SetMenuItem 0, 1, acMenuUngray

που ενεργοποιεί τη δεύτερη εντολή που βρίσκεται στην πρώτη ομάδα επιλογών του προσαρμοσμένου menu του ενεργού παραθύρου της εφαρμογής.

Κλείσιμο (Close Method): χρησιμοποιούμε τη μέθοδο **Close** για να κλείσουμε το παράθυρο κάποιου αντικειμένου της **Microsoft Access** ή το ενεργό παράθυρο εάν δεν καθοριστεί κάποιο άλλο. Η μέθοδος αυτή είναι παρόμοια με την ομώνυμη ενέργεια που περιγράψαμε στο κεφάλαιο διαχείρισης των μακροεντολών, και η κλήση της ακολουθεί μία σύνταξη της μορφής

DoCmd.Close objectType, objectName, save

Στην παραπάνω σύνταξη, το πεδίο **objectName** επιτρέπει τον **καθορισμό του ονόματος του αντικειμένου** το παράθυρο του οποίου επιθυμούμε να κλείσουμε, ενώ το όρισμα **objectType** καθορίζει τον τύπο αυτού του αντικειμένου, και δέχεται μία από τις τιμές **acDataAccessPage** (σελίδα πρόσβασης δεδομένων), **acDiagram** (διάγραμμα της βάσης δεδομένων), **acForm** (φόρμα), **acMacro** (μακροεντολή), **acModule** (λειτουργική μονάδα), **acQuery** (ερώτημα), **acReport** (αναφορά ή έκθεση), **acServerView** (προβολή διακομιστή), **acStoredProcedure** (αποθηκευμένη διαδικασία) και **acTable** (πίνακας). Τέλος η παράμετρος save δέχεται μία από τις τιμές **acSaveNo**, **acSavePrompt** και **acSaveYes** και καθορίζει εάν θα λάβει χώρα αποθήκευση των περιεχομένων του αντικειμένου πριν το κλείσιμο του παραθύρου με το οποίο συσχετίζεται. Η προεπιλεγμένη τιμή αυτής της παραμέτρου είναι η τιμή **acSavePrompt**. Στην περίπτωση χρήσης αυτής της τιμής η **Microsoft Access** ρωτά το χρήστη εάν επιθυμεί ή όχι την αποθήκευση των περιεχομένων του επιλεγμένου αντικειμένου.

Τυπικό παράδειγμα χρήσης της μεθόδου **Close**, είναι η κλήση της με τη μορφή

DoCmd.Close acForm, "DEPARTMENT", acSaveYes

η οποία κλείνει τη φόρμα **DEPARTMENT** αποθηκεύοντας τις αλλαγές που ενδεχομένως έχουν πραγματοποιηθεί πάνω σε αυτό το αντικείμενο.

Κλεψύδρα (Hourglass Method): χρησιμοποιούμε τη μέθοδο **Hourglass** για να αντικαταστήσουμε το δείκτη του ποντικιού με μία **κλεψύδρα** για όσο χρονικό διάστημα λαμβάνει χώρα η εκτέλεση της τρέχουσας διαδικασίας. Με τον τρόπο αυτό διευκολύνουμε την αλληλεπίδραση του χρήστη με το πρόγραμμα, καθώς παρατηρώντας το εικονίδιο της **κλεψύδρας**, αντιλαμβάνεται πως κάποιο συμβάν βρίσκεται σε εξέλιξη. Η κλήση αυτής της μεθόδου χαρακτηρίζεται από μία σύνταξη της μορφής **DoCmd.Hourglass hourglassOn** όπου το όρισμα **hourglassOn** δέχεται μία από τις τιμές **True** ή **False** και καθορίζει εάν θα λάβει χώρα χρήση της κλεψύδρας ή όχι.

Μεγιστοποίηση (Maximize Method): χρησιμοποιούμε τη μέθοδο **Maximize** για να μεγιστοποιήσουμε το ενεργό παράθυρο της βάσης δεδομένων της **Microsoft Access**, έτσι ώστε να καλύψει όλη την επιφάνεια εργασίας. Η μέθοδος καλείται χωρίς ορίσματα και η κλήση της χαρακτηρίζεται από μία σύνταξη της μορφής **DoCmd.Maximize**.

Μετάβαση σε εγγραφή (GoToRecord Method): χρησιμοποιούμε τη μέθοδο **GoToRecord** για να ορίσουμε κάποια από τις εγγραφές ενός ανοικτού πίνακα, φόρμας ή ερωτήματος, ως την τρέχουσα εγγραφή. Η κλήση αυτής της μεθόδου χαρακτηρίζεται από μία σύνταξη της μορφής

DoCmd.GoToRecord objectType, objectName, record, offset.

Στην παραπάνω σύνταξη η παράμετρος **objectName** χρησιμοποιείται για τον καθορισμό **του ονόματος του αντικειμένου**, ενώ ο τύπος του καθορίζεται δια της καταχώρησης στο πεδίο **objectType**, μιας εκ των τιμών **acDataForm** (φόρμα), **acDataQuery** (ερώτημα) και **acDataTable** (πίνακας). Τα υπόλοιπα δύο ορίσματα που χαρακτηρίζουν τη σύνταξη αυτής της μεθόδου, επιτρέπουν **τον καθορισμό της εγγραφής στην οποία επιθυμούμε να μεταβούμε**. Πιο συγκεκριμένα, το όρισμα **record** δέχεται μία από τις τιμές **acFirst**, **acLast**, **acNext**, και **acPrevious** που επιτρέπουν τη μετάβαση **στην πρώτη, στην τελευταία, στην επόμενη και στην προηγούμενη εγγραφή αντίστοιχα**. Εάν θέλουμε να μεταφερθούμε στη θέση νέας εγγραφής (new record) θα χρησιμοποιήσουμε την τιμή **acNewRec**, ενώ σε κάθε περίπτωση έχουμε τη δυνατότητα να μεταβούμε **σε οποιαδήποτε εγγραφή του επιλεγμένου αντικειμένου**, καταχωρώντας στο όρισμα record την τιμή **acGoTo**. Εάν δεν καταχωρηθεί κάποια τιμή σε αυτό το όρισμα, χρησιμοποιείται ως προεπιλεγμένη τιμή η **ac Next** που μεταφέρει την εστίαση του συστήματος στην επόμενη της τρέχουσας εγγραφής του ενεργού αντικειμένου.

Το τελευταίο όρισμα που μπορούμε να χρησιμοποιήσουμε στη σύνταξη αυτής της μεθόδου, φέρει το όνομα **offset** και δέχεται ως τιμή **μια αριθμητική ποσότητα**. Στην περίπτωση κατά την οποία στο όρισμα **record** καταχωρήσουμε μία από τις τιμές **acNext** και **acPrevious**, η ποσότητα αυτή εκφράζει **τον αριθμό των εγγραφών που θα πρέπει να μετακινηθούμε προς τα εμπρός ή προς τα πίσω αντίστοιχα, για να**

προσπελάσουμε τη νέα εγγραφή. Εάν για παράδειγμα χρησιμοποιήσουμε τις τιμές **record = acNext** και **offset = 15**, η νέα εγγραφή του ενεργού αντικειμένου είναι αυτή που βρίσκεται **15 θέσεις πιο μπροστά από την τρέχουσα εγγραφή**. Αντίθετα, εάν στην προηγούμενη παράμετρο έχουμε καταχωρήσει την τιμή **acGoTo**, χρησιμοποιούμε αυτό το όρισμα **για να καθορίσουμε επακριβώς την εγγραφή στην οποία θέλουμε να μεταβούμε**. Έτσι, χρησιμοποιώντας τις τιμές **record = acGoTo** και **offset = 185**, μεταφερόμαστε στην εγγραφή **υπ' αριθμόν 185** του ενεργού αντικειμένου, ανεξάρτητα από την εγγραφή στην οποία βρισκόμαστε τώρα.

Τυπικό παράδειγμα χρήσης της μεθόδου **GoToRecord**, είναι η κλήση της με τη μορφή

DoCmd.GoToRecord acDataTable, "Employee", acGoTo 20

η οποία μας μεταφέρει στην **υπ' αριθμόν 20** εγγραφή του πίνακα Employee.

Μετάβαση σε σελίδα (GoToPage Method): χρησιμοποιούμε τη μέθοδο **GoToPage** για να μεταφέρουμε την εστίαση του συστήματος **στο πρώτο στοιχείο ελέγχου μιας σελίδας**. Αυτή η μέθοδος είναι ιδιαίτερα χρήσιμη σε περιπτώσεις φορμών που διαθέτουν περισσότερες από μία σελίδες, και η κλήση της χαρακτηρίζεται από μία σύνταξη της μορφής

DoCmd.GoToPage pageNumber, right, down

Στην παραπάνω σύνταξη, το όρισμα **pageNumber** περιέχει **τον αριθμό της σελίδας της φόρμας στην οποία θέλουμε να μεταβούμε** – ας σημειωθεί πως για να χρησιμοποιηθεί αυτή η μέθοδος θα πρέπει πρώτα να επιλέξουμε τη φόρμα που θέλουμε να χρησιμοποιήσουμε, δια της κλήσης της μεθόδου **SelectObject**. Από την άλλη πλευρά, τα ορίσματα **right** και **down** ορίζουν **τις συντεταγμένες της άνω αριστερής γωνίας του τμήματος της σελίδας που θέλουμε να εμφανίσουμε**. Αυτές οι δύο παράμετροι χρησιμοποιούνται σε περιπτώσεις κατά τις οποίες **δεν επιθυμούμε να εμφανίσουμε ολόκληρη τη σελίδα της φόρμας αλλά μόνο ένα μέρος αυτής**, και οι τιμές που καταχωρούμε σε αυτές είναι εκπεφρασμένες σε **twips**. Το **twip** είναι μία μονάδα μέτρησης μήκους που χρησιμοποιείται από τη **Microsoft Access** και ορίζεται ως **το 1/1440 μιας ίντσας** (ένα **εκατοστό** έχει **567 twips**).

Τυπικό παράδειγμα χρήσης της μεθόδου **GoToPage** είναι η κλήση της με τη μορφή

DoCmd.GoToPage 3, 1440, 567

που μεταφέρει την εστίαση του συστήματος **στην τρίτη σελίδα της φόρμας και στο σημείο με συντεταγμένες (1440, 567)**.

Μετάβαση σε στοιχείο ελέγχου (GoToControl Method): χρησιμοποιούμε τη μέθοδο **GoToControl** για να μεταφέρουμε την εστίαση του συστήματος σε κάποιο πεδίο ή στοιχείο ελέγχου της τρέχουσας εγγραφής ενός ανοικτού αντικειμένου (αυτό το αντικείμενο μπορεί να είναι μία φόρμα ή ένας πίνακας σε κανονική προβολή ή προβολή φύλλου δεδομένων, ή ακόμη και μία αναφορά). Αυτή η μέθοδος είναι παρό-

μοια με την ομώνυμη ενέργεια που περιγράψαμε στο κεφάλαιο διαχείρισης μακροεντολών, και η κλήσης χαρακτηρίζεται από μία σύνταξη της μορφής

DoCmd.GoToControl controlName

όπου **controlName** είναι **το όνομα του στοιχείου ελέγχου** στο οποίο επιθυμούμε να μεταφέρουμε την εστίαση του συστήματος.

Τυπικό παράδειγμα χρήσης αυτής της μεθόδου, είναι η κλήση της με τη μορφή

DoCmd.GoToControl "EmployeeSSN"

που μεταφέρει την εστίαση του συστήματος στο πεδίο «**EmployeeSSN**» της φόρμας **EMPLOYEES**.

Μετακίνηση – Προσαρμογή μεγέθους (MoveSize Method): χρησιμοποιούμε τη μέθοδο **MoveSize** για να μετακινήσουμε ή να αλλάξουμε το μέγεθος του ενεργού παραθύρου της βάσης δεδομένων. Αυτή η μέθοδος είναι παρόμοια με την ενέργεια **MoveSize** που περιγράψαμε στο κεφάλαιο διαχείρισης των μακροεντολών, και η κλήση της χαρακτηρίζεται από μία σύνταξη της μορφής

DoCmd.MoveSize right, down, width, height

όπου τα ορίσματα **right** και **down** καθορίζουν **τις συντεταγμένες της άνω αριστερής γωνίας της νέας θέσης του παραθύρου** (αυτές οι συντεταγμένες μετρώνται σε σχέση με τις συντεταγμένες των πλευρών του παραθύρου που περιέχει το ενεργό παράθυρο της βάσης δεδομένων), ενώ τα ορίσματα **width** και **height** εκφράζουν **τις νέες διαστάσεις του παραθύρου**, και πιο συγκεκριμένα, το **πλάτος** και το **ύψος** του. Είναι σημαντικό να αναφερθεί, πως οι αριθμητικές τιμές που καταχωρούμε σε αυτά τα τέσσερα ορίσματα, είναι εκπεφρασμένες σε **twips**. Όπως έχει αναφερθεί σε προηγούμενη παράγραφο, το **twip** είναι μία ειδική μονάδα μέτρησης μήκους της **Microsoft Access**, και ορίζεται ως **το 1/1440 μιας ίντσας**.

Τυπικό παράδειγμα χρήσης της μεθόδου **MoveSize** είναι η κλήση της με τη μορφή

DoCmd.MoveSize 1440, 1440

που μετακινεί το παράθυρο σε μία νέα θέση επί της οθόνης, χωρίς ωστόσο να μεταβάλει τις διαστάσεις του.

Μεταφορά βάσης δεδομένων (TransferDatabase Method): χρησιμοποιούμε τη μέθοδο **TransferDatabase** για να κάνουμε **εισαγωγή, εξαγωγή ή σύνδεση δεδομένων** ανάμεσα σε μία άλλη βάση δεδομένων και στην τρέχουσα βάση. Η μέθοδος αυτή είναι παρόμοια με την ομώνυμη ενέργεια που περιγράψαμε στο κεφάλαιο διαχείρισης των μακροεντολών, και η κλήση της χαρακτηρίζεται από μία σύνταξη της μορφής

DoCmd.TransferDatabase transferType, databaseType, databaseName, objectType, source, destination, structureOnly, saveLoginId.

Στην παραπάνω σύνταξη, το όρισμα **transferType** επιτρέπει τον καθορισμό **του τύπου της μεταφοράς των δεδομένων** που θα λάβει χώρα ανάμεσα στις δύο βάσεις. Αυτό το όρισμα παίρνει μία από τις τιμές **acExport**, **acImport** και **acLink** που καθορίζουν αντίστοιχα διαδικασία **εξαγωγής, εισαγωγής και σύνδεσης δεδομένων**. Η προεπιλεγμένη τιμή για αυτή την παράμετρο είναι η **acImport**, που επιτρέπει **την εισαγωγή δεδομένων στην τρέχουσα βάση**. Είναι σημαντικό να αναφερθεί στο σημείο αυτό, πως η διαδικασία **σύνδεσης δεδομένων (data linking)** που καθορίζεται δια της χρήσης της τιμής **acLink**, είναι δυνατή, μόνο όταν και οι δύο βάσεις δεδομένων έχουν δημιουργηθεί με τη **Microsoft Access**.

Από την άλλη πλευρά, το όρισμα **databaseType** επιτρέπει τον καθορισμό **του τύπου της βάσης δεδομένων** από ή προς την οποία θα μεταφέρουμε δεδομένα. Υπάρχουν πολλοί διαφορετικοί τύποι βάσεων δεδομένων που μπορούνε να χρησιμοποιηθούνε και οι οποίοι καθορίζονται αποδίδοντας στην παράμετρο **databaseType** μια από τις τιμές «**Microsoft Access**» (η τιμή αυτή είναι και η προεπιλεγμένη) , «**Jet 2.x**», «**Jet 3.x**», «**dBase III**», «**dBase IV**», «**dBase 5**», «**Paradox 3.x**», «**Paradox 4.x**», «**Paradox 5.x**», «**Paradox 7.x**», «**ODBC Databases**». Έχοντας καθορίσει τον τύπο της βάσης δεδομένων που επιθυμούμε να χρησιμοποιήσουμε θα πρέπει στη συνέχεια να καθορίσουμε και το όνομά της καταχωρώντας το όνομα και τη διαδρομή του αρχείου που την περιέχει, στην παράμετρο **databaseName**.

Μετά τον καθορισμό του τύπου και του ονόματος της βάσης, θα πρέπει να προσδιορίσουμε **τον τύπο και το όνομα του αντικειμένου που θέλουμε να χρησιμοποιήσουμε**. Ο τύπος του αντικειμένου καθορίζεται κατά τα γνωστά καταχωρώντας στην παράμετρο **objectType** μια εκ των τιμών **acTable** (πίνακας), **acQuery** (ερώτημα), **acForm** (φόρμα), **acReport** (αναφορά ή έκθεση), **acMacro** (μακροεντολή), **acModule** (λειτουργική μονάδα), **acDataAccessPage** (σελίδα πρόσβασης δεδομένων), **acServerView** (προβολή διακομιστή), **acDiagram** (διάγραμμα της βάσης δεδομένων) και **acStoredProcedure** (αποθηκευμένη διαδικασία). Η προεπιλεγμένη τιμή για αυτή την ιδιότητα είναι η **acTable**. Από την άλλη πλευρά, τα ονόματα των αντικειμένων προέλευσης και προορισμού καθορίζονται δια της χρήσης των ορισμάτων **source** και **destination** – εάν δεν χρησιμοποιηθεί το όρισμα **destination** τότε το αντικείμενο προέλευσης θα μεταφερθεί στην τρέχουσα βάση δεδομένων με το ίδιο όνομα. Μία ακόμη παράμετρος που μπορούμε να καθορίσουμε στο σημείο αυτό αφορά **το τμήμα του αντικειμένου που επιθυμούμε να μεταφέρουμε**. Αν και στις πιο πολλές περιπτώσεις η μεταφορά αυτή περιλαμβάνει **τόσο τη δομή του αντικειμένου όσο και το σύνολο των δεδομένων που βρίσκονται αποθηκευμένα σε αυτό**, εν τούτοις σε πολλές περιπτώσεις ζητούμε να μεταφέρουμε **μόνο τη δομή του αντικειμένου και όχι τα δεδομένα που περιέχει**. Για να το κάνουμε αυτό θα πρέπει να καταχωρήσουμε στο όρισμα **structureOnly** την τιμή «**True**». Τέλος το όρισμα **saveLoginId** δέχεται μία από τις τιμές «**True**» ή «**False**» ανάλογα με το εάν επιθυμούμε ή όχι να αποθηκεύσουμε το **όνομα (login name)** και τον **κωδικό πρόσβασης (password)** που καταχώρησε ο χρήστης προκειμένου να συνδεθεί σε μία βάση δεδομένων η οποία ακολουθεί το μοντέλο του **ODBC (Open Database Connectivity)**.

Παράδειγμα χρήσης της μεθόδου **TransferDatabase** είναι η κλήση της με τη μορφή

DoCmd.TransferDatabase acImport, "Microsoft Access", _
"c:\dbases\company.mdb", acTable, "Employee", ,True

η οποία μεταφέρει τη δομή του πίνακα «**Employee**» από τη βάση δεδομένων «**c:\dbases\company.mdb**» στην τρέχουσα βάση.

Μεταφορά κειμένου (TransferText Method): χρησιμοποιούμε τη μέθοδο **TransferText** για να πραγματοποιήσουμε διαδικασία ε**ισαγωγής, εξαγωγής ή σύνδεσης δεδομένων** ανάμεσα σε ένα αρχείο κειμένου και στην τρέχουσα βάση. Η κλήση αυτής της μεθόδου ακολουθεί τη σύνταξη

DoCmd.TransferText transferType, specificationName, tableName, fileName, hasFieldNames, htmlTableName, codepage

Στην παραπάνω σύνταξη το όρισμα **transferType** δέχεται μία από τις τιμές **acExportDelim**, **acExportFixed**, **acExportHTML**, **acExportMerge**, **acImportDelim** (η τιμή αυτή είναι και η προεπιλεγμένη), **acImportFixed**, **acImportHTML**, **acLinkDelim**, **acLinkFixed**, **acLinkHTML** οι οποίες καθορίζουν **τον τύπο της μεταφοράς (εισαγωγή, εξαγωγή ή σύνδεση)** καθώς και **τον τύπο του αρχείου κειμένου**. Η χρήση αυτής της παραμέτρου είναι ιδιαίτερα σημαντική καθώς μας δίνει τη δυνατότητα **να κάνουμε εισαγωγή κειμένου που είναι διαμορφωμένο σε γραμμές και στήλες, σε κάποιον από τους πίνακες της τρέχουσας βάσης**. Το όνομα του πίνακα στον οποίο θα λάβει χώρα η αποθήκευση των δεδομένων (σε περίπτωση διαδικασίας εισαγωγής κειμένου) καθορίζεται δια της χρήσης της παραμέτρου **tableName**, ενώ το όνομα **specificationName** δέχεται ως τιμή ένα **όνομα προδιαγραφής (specification name)** που καθορίζει με μοναδικό τρόπο το είδος της ανταλλαγής πληροφορίας ανάμεσα στο αρχείου κειμένου και στην τρέχουσα βάση. Περισσότερες πληροφορίες σχετικά με το θέμα αυτό ξεφεύγουν από το σκοπό της συγγραφής αυτού του βιβλίου και ο αναγνώστης παραπέμπεται στα αρχεία τεκμηρίωσης της **Microsoft Access** ή σε κάποιο άλλο εγχειρίδιο που καλύπτει με περισσότερη λεπτομέρεια τη χρήση αυτής της εφαρμογής.

Η αμέσως επόμενη παράμετρος που θα πρέπει να καθοριστεί, αφορά **το όνομα του αρχείου κειμένου που θα χρησιμοποιηθεί στη διαδικασία μεταφοράς δεδομένων από ή προς την τρέχουσα βάση**. Για να καθορίσουμε αυτή την πληροφορία θα πρέπει να καταχωρήσουμε στο όρισμα **fileName** το όνομα και τη διαδρομή αυτού του αρχείου κειμένου. Η παράμετρος **hasFields** δέχεται μία από τις τιμές «**True**» ή «**False**» ανάλογα με το εάν **η πρώτη γραμμή του αρχείου δεν περιέχει τιμές αλλά ονόματα κάποιων πεδίων**. Αυτό ισχύει κυρίως σε περιπτώσεις αρχείων κειμένου **τα περιεχόμενα των οποίων είναι οργανωμένα σε γραμμές και στήλες**. Εάν η παράμετρος αυτή λάβει την τιμή «**True**» που σημαίνει πως η πρώτη γραμμή του αρχείου κειμένου περιέχει ονόματα πεδίων, η **Microsoft Access** χρησιμοποιεί αυτά τα ονόματα ως τα ονόματα των αντίστοιχων πεδίων του πίνακα που θα δημιουργηθεί για την αποθήκευση αυτών των δεδομένων. Από την άλλη πλευρά, το όρισμα **htmlTableName** χρησιμοποιείται στην περίπτωση κατά την οποία το αρχείο κειμένου είναι διαμορφωμένο ως σελίδα **HTML** (αυτό σημαίνει πως στο όρισμα **transferType** έχουμε

καταχωρήσει μία από τις τιμές **acImportHTML** ή **acLinkHTML**). Εάν τα δεδομένα της σελίδας **HTML** βρίσκονται οργανωμένα σε κατάλληλα διαμορφωμένους **πίνακες** ή **λίστες**, μπορούμε να χρησιμοποιήσουμε το όρισμα **htmlTableName** για να καθορίσουμε **το όνομα του πίνακα (ή της λίστας) τα δεδομένα του οποίου επιθυμούμε να μεταφέρουμε στην τρέχουσα βάση**. Εάν αυτό το όρισμα δεν χρησιμοποιηθεί, θα λάβει χώρα μεταφορά των περιεχομένων **του πρώτου πίνακα ή λίστας της συγκεκριμένης ιστοσελίδας**. Τέλος το όρισμα **codePage** χρησιμοποιείται για τον καθορισμό της **κωδικοσελίδας** που χρησιμοποιείται για τη διαμόρφωση των περιεχομένων του αρχείου κειμένου που θέλουμε να χρησιμοποιήσουμε.

Τυπικό παράδειγμα χρήσης της μεθόδου **TransferText**, είναι η κλήση της με τη μορφή

DoCmd.TransferText acExportDelim, "defaultSpec", "Employees", _
"c:\dBases\employeeReport.txt"

που μεταφέρει τα περιεχόμενα του πίνακα **"Employees"** στο αρχείου κειμένου **"c:\dBases\employeeReport.txt"**.

Μεταφορά Υπολογιστικού Φύλλου (TransferSpreadsheet Method): χρησιμοποιούμε τη μέθοδο **TransferSpreadSheet** για να κάνουμε **ανταλλαγή δεδομένων (εισαγωγή, εξαγωγή ή σύνδεση)** ανάμεσα σε ένα **φύλλο δεδομένων** και στην τρέχουσα βάση. Αυτή η μέθοδος είναι παρόμοια με την ενέργεια **TransferSpreadsheet** που παρουσιάσαμε στο κεφάλαιο διαχείριση των μακροεντολών, και η κλήση της χαρακτηρίζεται από μία σύνταξη της μορφής

DoCmd.TransferSpreadsheet transferType, spreadsheetType, tableName,
filename, hasFieldNames, range

Σε πλήρη αναλογία με τις προηγούμενες μεθόδους που συσχετίζονται με διαδικασίες μεταφοράς δεδομένων, το πεδίο **transferType** δέχεται μία από τις τιμές **acImport**, **acExport** και **acLink** ανάλογα με το εάν η διαδικασία μεταφοράς είναι **εισαγωγή, εξαγωγή** ή **σύνδεση**. Από την άλλη πλευρά, το όρισμα **spreadSheetType**, επιτρέπει **τον καθορισμό του τύπου του υπολογιστικού φύλλου που θέλουμε να χρησιμοποιήσουμε**. Η τρέχουσα έκδοση της **Microsoft Access** επιτρέπει τη χρήση υπολογιστικών φύλλων που έχουν δημιουργηθεί από διαφορετικές εκδόσεις των δύο πιο δημοφιλών προγραμμάτων διαχείρισης υπολογιστικών φύλλων, δηλαδή του **Microsoft Excel** και του **Lotus 1-2-3**. Ο καθορισμός του τύπου του υπολογιστικού φύλλου που επιθυμούμε να χρησιμοποιήσουμε γίνεται δια της καταχώρησης στο πεδίο **spreadSheetType** μίας εκ των τιμών «**acSpreadSheetTypeExcel3**», «**acSpreadSheetTypeExcel4**», «**acSpreadSheetType Excel5**», «**acSpreadSheetTypeExcel7**», «**acSpreadSheetTypeExcel8**» (η τιμή αυτή είναι και η προεπιλεγμένη), «**acSpreadSheetTypeExcel9**», «**acSpreadSheetTypeLotusWK1**», «**acSpreadSheetTypeLotusWK3**» και «**acSpreadSheetTypeLotusWK4**».

Τα επόμενα τρία ορίσματα της μεθόδου **TransferSpreadsheet**, είναι παρόμοια με εκείνα της μεθόδου **TransferText** που περιγράψαμε στην προηγούμενη παράγραφο. Έτσι το όρισμα **tableName** επιτρέπει τον καθορισμό του ονόματος του πίνακα που θα χρησιμοποιηθεί στη διαδικασία εισαγωγής, εξαγωγής ή σύνδεσης δεδο-

μένων, ενώ το όνομα **fileName** δέχεται ως τιμή μια συμβολοσειρά που περιέχει **το όνομα και τη διαδρομή του αρχείου του υπολογιστικού φύλλου που επιθυμούμε να χρησιμοποιήσουμε**. Στην περίπτωση κατά την οποία η πρώτη γραμμή του υπολογιστικού φύλλου δεν περιέχει δεδομένα αλλά ονόματα πεδίων, θα πρέπει να αποδώσουμε στην παράμετρο **hasFieldNames** την τιμή «**True**». Στην περίπτωση αυτή, αυτά τα ονόματα των πεδίων θα χρησιμοποιηθούν ως τα ονόματα των πεδίων του πίνακα που θα δημιουργήσουμε για την αποθήκευση αυτών των δεδομένων.

Τέλος το όρισμα **range** χρησιμοποιείται σε περίπτωση κατά την οποία **δεν επιθυμούμε να χρησιμοποιήσουμε τα δεδομένα ολόκληρου του υπολογιστικού φύλλου**, αλλά μόνο εκείνα που ανήκουν σε μια συγκεκριμένη περιοχή του. Στην περίπτωση αυτή θα πρέπει να καταχωρήσουμε στην παράμετρο **range** μία συμβολοσειρά η τιμή της οποίας είναι **μία έγκυρη έκφραση περιοχής φύλλου δεδομένων**. Ας σημειωθεί πως η παράμετρος αυτή χρησιμοποιείται **μόνο για την εισαγωγή δεδομένων από ένα υπολογιστικό φύλλο στην τρέχουσα βάση δεδομένων**, και όχι για την εξαγωγή δεδομένων προς ένα υπολογιστικό φύλλο – στην τελευταία περίπτωση, η χρήση της παραμέτρου range θα οδηγήσει σε αποτυχία της διαδικασίας εξαγωγής δεδομένων.

Τυπικό παράδειγμα χρήσης της μεθόδου **TransferSpreadsheet** είναι η κλήση της με τη μορφή

```
DoCmd.TransferSpreadsheet acImport, acSpreadSheetTypeExcel8, "Sales", _
    "c:\dataFiles\sales.xls", True, "A1:D15"
```

που έχει ως αποτέλεσμα τη μεταφορά των περιεχομένων της περιοχής **A1:D15** του φύλλου δεδομένων **«c:\dataFiles\sales.xls»** στον πίνακα **«Sales»** της τρέχουσας βάσης δεδομένων.

Μετονομασία (Rename Method): χρησιμοποιούμε τη μέθοδο **Rename** για να αλλάξουμε το όνομα κάποιου από τα αντικείμενα της βάσης δεδομένων. Αυτή η μέθοδος είναι παρόμοια με την ομώνυμη ενέργεια που περιγράψαμε στο κεφάλαιο διαχείρισης των μακροεντολών, και η κλήση της χαρακτηρίζεται από μία σύνταξη της μορφής

```
DoCmd.Rename newName, objectType, oldName
```

Στην παραπάνω σύνταξη τα ορίσματα **oldName** και **newName** εκφράζουν **το παλαιό και το νέο όνομα του αντικειμένου αντίστοιχα**, ενώ το όρισμα **objectType** χρησιμοποιείται για τον καθορισμό του τύπου του αντικειμένου και δέχεται κατά τα γνωστά μία εκ των τιμών **acDataAccessPage** (σελίδα πρόσβασης δεδομένων), **acDiagram** (διάγραμμα της βάσης δεδομένων), **acForm** (φόρμα), **acMacro** (μακροεντολή), **acModule** (λειτουργική μονάδα), **acQuery** (ερώτημα), **acReport** (αναφορά), **acServerView** (προβολή διακομιστή), **acStoredProcedure** (αποθηκευμένη διαδικασία) και **acTable** (πίνακας της βάσης δεδομένων).

Τυπικό παράδειγμα χρήσης της μεθόδου **Rename** είναι η κλήση της με τη μορφή

```
DoCmd.Rename "Employees2003", acTable, "Employees2002"
```

πού προκαλεί τη μετονομασία του πίνακα «**Employees2002**» δίδοντάς του το όνομα «**Employees2003**».

Όλες οι εγγραφές (ShowAllRecords): σε περίπτωση κατά την οποία λαμβάνει χώρα **προεπισκόπηση μόνο ενός μέρους των εγγραφών του ενεργού αντικειμένου** – αυτό συμβαίνει συνήθως όταν έχει εφαρμοσθεί επί του αντικειμένου κάποιο φίλτρο – μπορούμε να χρησιμοποιήσουμε αυτή τη μέθοδο γ**ια να καταργήσουμε το φίλτρο ή τη συνθήκη περιορισμού των εγγραφών που έχει εφαρμοσθεί, και να εμφανίσουμε όλες τις εγγραφές**. Η μέθοδος αυτή χρησιμοποιείται χωρίς ορίσματα και η κλήση της χαρακτηρίζεται από μία σύνταξη της μορφής **DoCmd.ShowAllRecords**.

Προσθήκη Menu (AddMenu Method): χρησιμοποιούμε τη μέθοδο **AddMenu** για να κατασκευάσουμε διάφορα είδη **menu επιλογών** όπως είναι για παράδειγμα **τα αναδυόμενα menus, τα menus συντόμευσης και οι γραμμές menu (menu bars)**. Αυτή η μέθοδος είναι παρόμοια με την ομώνυμη ενέργεια που παρουσιάσαμε στο κεφάλαιο διαχείρισης μακροεντολών, και η κλήση της χαρακτηρίζεται από μία σύνταξη της μορφής

DoCmd.AddMenu menuName, menuMacroName, statusBarText

Στην παραπάνω σύνταξη, το όρισμα **menuName** χρησιμοποιείται για τον καθορισμό του ονόματος του menu που θα εμφανίζεται στη γραμμή εργαλείων, ενώ το όρισμα **menuMacroName** επιτρέπει τον καθορισμό του ονόματος της ομάδας μακροεντολών οι ενέργειες της οποίας θα συσχετιστούν με το νέο menu επιλογών. Τέλος το όρισμα **statusBarText** χρησιμοποιείται για τον καθορισμό του κειμένου που θα εμφανίζεται στη **γραμμή κατάστασης (status bar)** της εφαρμογής, κάθε φορά που θα χρησιμοποιείται το νέο menu επιλογών.