
TO THE INSTRUCTOR . . .

The Instructor's Answer Manual for Access cases in *Solve it! Version 3.0 for Windows* contains a brief description of the learning objectives for each case, and a step-by-step detailed answer. There are 3 Solution File diskettes for Access for *Solve it! Version 2.8*. Separate sets of solution files have been provided for users of Access 2000 and Access for Office97.

Detailed Instructions for Database Case 1 Using Access

These instructions apply for both Access for Office97 and Access for Office 2000 users.

FILE: CASE1.MDB

1. Create a new database in Access, and then save it. Now open DBASES.MDB on your *SolveIt!* diskette, and copy and paste the ROLLASON data table (data and structure) into the new database. From the Database window, select and open the ROLLASON table. Browse records.
2. **TASK 1:** Use the Table Design toolbar button or select View/Table Design from the menu to amend the structure of the ROLLASON table. Add the five new fields. (Use Edit/Insert Row from the menu or click the Insert Row toolbar button to add empty fields between existing ones). All fields should be set to the Text data type except for Company No, which will be Numeric. Note that the default size for a Text field is 50 characters, which is far too generous for the Product Group and Order Pattern fields. Edit Field Size accordingly.

Select File/Save from the menu or click the Save toolbar button to save changes to the table structure.
3. **TASK 2:** Switch to Datasheet View using the toolbar button or selecting View/Datasheet from the menu. Enter data for the new fields to complete existing records. Use the Tab key to advance to the next field within a record. Now enter three new records. As you move between records, Access will automatically save any changes made to previous ones.
4. **TASK 3:** From Datasheet View, select Edit/Find from the menu. From the Find in Field dialog box locate the Rotary Wings record, and amend the record as required. You may need to use File/Save Record.
5. **TASK 4:** Send table to print by selecting File/Print from the menu or clicking the Print toolbar button. You may need to select File/Print Setup and change the orientation to Landscape in order to show all fields in each record on one line.
6. **TASK 5:** From the Database window (use F11 key to assist), create a query by clicking on Queries and then the New button. Click OK. This generates an empty Select Query type. From the Show Table, select

ROLLASON, click Add and then click Close to add the ROLLASON table to the new query. Select and drag the required fields down to the Field: bar of the QBE Grid. Save (e.g.: TASK5) and Run the query. Send the query to Print.

7. **TASK 6:** Create a new query. Add the ROLLASON table. Select and drag the Company, Client, Title and City fields down to the Field: bar of the QBE Grid. In the QBE Grid, deselect the Show: check box under the City field. In the Criteria: bar of the City field, type Green Bay. (Access will automatically enclose the text in quotation marks, indicating that it is text within a field). Save (e.g.: TASK6) and Run the query. Send the query to Print.

8. **TASK 7:** A simple solution would be to create several Product Group fields, to accommodate customers who order from more than one group. Also explore the concept of one to many relationships (e.g.: one Customer and many Products). As the database becomes more advanced and complicated, a separate Product Groups lookup table, which links back to a Customer Details table, may be a more efficient way of handling this problem.

Detailed Instructions for Database Case 2 Using Access

These instructions apply for both Access for Office97 and Access for Office 2000 users.

FILE: CASE2.MDB

1. Create a new database in Access, and then save it. Now open DBASES.MDB on your *SolveIt!* diskette, and copy and paste the CNN data table (data and structure) into the new database. From the Database window, select and open the CNN table. Browse records.
2. **TASK 1:** Use the Table Design toolbar button or select View/Table Design from the menu to amend the structure of the CNN table. Add the new fields. Save changes.
3. **TASK 2:** Switch to Datasheet View using the toolbar button or selecting View/Datasheet from the menu. Enter data for the new fields to complete existing records. Now enter three new records.
4. **TASK 3:** Print the table using File/Print from the menu, or clicking the Print toolbar button.

Create a query based on the CNN table. Select and drag the Last_Name and Comments fields down onto the Field: bar of the QBE Grid. Save (e.g.: TASK3) and Run and then Print the query.

5. **TASK 4:** Create a new query based on the CNN table. Select and drag the Last_Name and Hrly_Rate fields onto the Field: bar of the QBE Grid. In the QBE Grid, click in the next blank field (i.e.: next to Hrly_Rate), and enter the following expression (required syntax is exactly as shown below, and you may find it useful to right click and select Zoom to give yourself more room while entering this expression):

TOTAL HOURS: [Q1 1997]+[Q2 1997]+[Q3 1997]+[Q4 1997]


This action has the effect of creating a new, calculated field (TOTAL HOURS) which adds together the four quarterly hours worked field for 1997. Save (eg: TASK4) and Run the query. Exit to the Database window (press F11).

Create a report. Click on Report and then click New. From the Select a Table/Query window, choose the TASK4 query, and then click the Report Wizards button. Select Tabular format. The first of several Report wizard

dialog boxes appears. Make these choices as you proceed through the dialog boxes:

- Include all fields in the report using the >> button
- Click Next
- Sort by Last_Name
- Click Next
- Give your report a name (e.g.: Casual Staff Report)
- Click Finish

Access generates the report and displays it in Print Preview mode.

Change to Report Design view by clicking the Close  button.

6. **TASK 5:** Return to the Database window (F11). Open the TASK4 query. Use File/Save As from the menu to save the query with a new name (eg: TASK5). In the Criteria: bar of the Hrly_Rate field enter >100. Select and drag the Skills field onto the Field: bar of the QBE Grid. In the Criteria: bar of the Skills field enter PS (Process Server). Save the changes, and Run the Query.

Create a new report based on the TASK5 query using the Tabular report wizard. Make the following choices:

- Include all fields
- Sort on Last_Name
- Give report an appropriate name

7. **TASK 6:** Extensions suggested by students usually involve using Access functions not yet covered in class. Possible extensions include:

- sorting the table by skill and then by hourly rate
- extending the notation used in the Skill field for Expert Witnesses. At the moment it is not possible to determine what area these people are expert in. Implement a simple code (e.g.: EWP for Psychologists, EWM for Medical, EWE for Engineering, etc).
- grouping the report by skill
- devising a scoring system for each contract staffer based on an assessment of their work by the Partner who employed them

-
- producing a summary report of CNN's usage of contract staff for each skill showing the range of hourly rates. This report could also contain the percentage of total contractor expense accounted for by each skill type.
8. **TASK 7:** This task could be easily implemented by independently assessing the abilities of each contract staffer and placing the score in a separate, non-memo field. Use some sort of standardised score to enable searching and sorting.

Detailed Instructions for Database Case 3 Using Access

These instructions apply for both Access for Office97 and Access for Office 2000 users.

FILE: CASE3.MDB

Note: *Instructors should spend some time during this case explaining the intricacies and various components of the Access report design window.*

1. 1. Create a new database and then save it. Now open DBASES.MDB on your SolveIt! diskette, and copy and paste the FARRING data table into the new database. Browse records.
2. **TASK 1:** Amend structure by adding two new fields, Inventory (Numerical data type) and Value (Currency data type). Set the Format for the Value field in the Field Properties section to Currency as well. Select File/Save from the menu or click the Save toolbar button to save changes to the table structure.

Switch to Datasheet view of the FARRING table and enter data for the new fields to complete existing records. Ensure the data entered closely parallels that shown in the FARRING table of the CASE3.MDB answer file.


Enter six additional records, which relate to the bolt sets (bolt, nut and spacer) detailed in the Task 1 question for this case. Print the amended table.



3. **TASK 2:** Create a query based on the FARRING table. Select and drag the Part_Code and Part_Name fields onto the Field: bar of the QBE Grid. Create five new calculated fields; one, which calculates values for each of the four product groups, and one for the value of Inventory holdings:

Swanval: [Swanborn]*[Value]
Deluxval: [Deluxe]*[Value]
Directval: [Director]*[Value]
Chairval: [Chairman]*[Value]
Inventval: [Inventory]*[Value]

Format each new field to Currency. (Click on each calculated field in turn, click right, select Properties, click Format, and from the drop down list select Currency).

Select View/Totals or click the Σ Totals toolbar button. Access adds a Total: bar to the QBE Grid. (In the Total: bar, the default value is Group By).

Click on the  button on each of the five calculated fields in turn, and from the drop down list select Sum. (Leave Part_Name and Part_Code set to Group By).

Save the query (e.g.: TASK2), and double click in top left   of query window to close and return to Database window. Re-enter query and notice the differences. Access has added the Sum function to each of the calculated fields. Sum results will not appear in query mode, so we need to create a report.

Create a report based on the TASK2 query and using the Report Wizard. Make the following choices:

- Include all fields
- Sort by Part_Code
- Give the report a title (e.g.: Materials Costs Report)

Access will generate the report and display it in Print Preview mode. The report is sorted by Part_Code.

Field formatting and calculations can be performed at either query design or report design level. As report design mode is a great deal more confusing than that of query design (especially for new users), it's probably a good move to set up as many requirements as possible in query design mode.

4. **TASK 3:** Create a new query based on the FARRING table. Drag the Part_Code and Part_Name fields onto the QBE Grid. Create three new calculated fields:

Swan: [Swanborn]*3
Direct: [Director]*42
Total Components: [Long]+[Reach]

Now drag the Inventory field onto the QBE Grid. Lastly, under the Part_Code field, click on the Sort: bar and from the drop down list, choose Ascending order.

Save (e.g.: TASK3) and run the query. The resulting dynaset shows the number of components required to complete the order: firstly by product type (Swan and Direct), and then by total components (Swan+Direct). Finally the Inventory holdings are displayed. A quick comparison of Total Components and Inventory shows a deficit for 5 part types.

For a quick report, print out at the query level. Otherwise, create a new report based on the TASK3 query, using Tabular format, and including all fields.

-
5. **TASK 4:** Go into the TASK3 query and using File/Save As from the menu, save with a new name (e.g.: TASK4). Under the Criteria cell for the Direct field, type >[Inventory]. Save changes and run the query. Access returns a reduced dynaset of 5 records, where order requirements exceed those of current inventory holdings.
6. **TASK 5:** Create a new query based on the FARRING table. Drag the Part_Code and Part_Name fields onto the QBE Grid. Click on the Sort: cell under the Part_Code field and set to Ascending order. Save query (e.g.: TASK5), and then Run. Send query to print.
7. **TASK 6:** Go into the TASK5 query, and using File/Save As from the menu, save with a new name (e.g.: TASK6). Drag the Value field onto the QBE Grid. Click on the Sort: cell under the Value field and set to Descending order. Click on the Sort: cell under the Part_Code field, and set to (not sorted). Save changes, and then run the query. Send to print.
8. **TASK 7:** A useful addition might be the introduction of a new field to represent a 'reserved stock' for existing orders. This may facilitate a more accurate illustration of component ordering requirements. Queries would then need to account for reserved stock and inventory holdings when examining particular orders.

Another change might be the incorporation of other production costs and additional costs held in other databases to enable even more information for accurate pricing. These other costs would be included in other databases used in conjunction with the BoM list.

Detailed Instructions for Database Case 4 Using Access

These instructions apply to both Access for Office 97 and Access for Office 2000 users.

FILE: CASE4.MDB

This case is useful for illustrating the database concept of concatenation, and for demonstrating how existing queries can be modified and saved as new queries.

1. 1. Create a new database, and then save it. Now open DBASES.MDB on the SolveIt! diskette, and copy and paste the FAGAN data table into the new database.

2. **TASK 1:** Create a new query based on the FAGAN table. Drag the Drug_Code, Drug_Name, Supplier, Tabs, and Mg_Per_Tab fields onto the QBE Grid. The wanted drug classes are A2, A3, A4, G9, P1 and P4. To filter out the unwanted drug classes we need to use the OR operator. In the Criteria: cell under the Drug_Code field type:

A2 or A3 or A4 or G9 or P1 or P4 all on the one line.

Note: FAGAN.DBF does not include any records for Drug Code P4.

Run the query and view the reduced dynaset. There should be only 19 records. Go back into query design, and in the Sort: cell of the Drug_Code field, select Ascending order. Run the query again to see the sorted records. Save the query (e.g.: TASK1).

3. **TASKS 2and 3:** Go into the TASK1 query and using File/Save As from the menu, save with a new name (e.g.: TASK2). Create a new calculated field to give total drug usage:

Total Usage: [Tabs]*[Mg_Per_Tab]

Save changes, and run the query to see the effect of the calculation. Exit to the Database window.


Create a new report based on the TASK2 query, and using the Report Wizard. Make the following choices:

- Include all fields
- For *Group*, select Drug_Code
- Click Next
- Click Summary Options

- For *What summary values would you like calculated:* select Sum for Total Usage
- Click OK
- Click Next
- Retain default selection- Portrait Orientation.
- Click Next
- Click Next
- Enter a Report Title (e.g.: Government Drug Usage Report)
- Click Finish

Access generates the report and displays it in Print Preview mode. Use File/Print SetUp from the menu to set report to Portrait orientation (this could also have been done during the wizard setup procedure). The report subtotals by each drug code and finishes with a grand total of 22,400.

Save the report (e.g.: TASK2/3 REPO), and for Task 3, send to print. Exit (F11) to the Database window.

4. **TASK 4:** Go into the TASK2 query and using File/Save As from the menu, save with a new name (e.g.: TASK4). This task requires usage of the full record set of the FAGAN table, so delete the OR statement in the Criteria: cell under the Drug_Code field. Scroll along the QBE Grid, and also delete the Tab and Mg_Per_Tab field columns. (Move the mouse to the top of each field column until the pointer changes to a  symbol. Click to highlight the whole column, and then press the delete key). Save query changes. Run the query to view the dynaset (29 records), and then exit to the Database window.


Create a new report based on the TASK4 query. Select the Report Wizard option from the New Reports dialog window. Make the following choices:

- Include all fields
- Click Next
- Group by Drug_Code
- Click Next
- Sort by Supplier and then perhaps by Drug_Name
- Click Summary Options
- For *What summary values would you like calculated:* select Sum for Total Usage
- Click OK
- Click Next
- Click Next
- Click Next
- Give the report a title (e.g.: Drug Company Report)

-
- Click Finish.

The resulting report sorts firstly by Drug_Code, and then within this, by Supplier and then Drug_Name. Sub totals are presented for each drug code, and the grand total for Total Usage is 33,113. Save the report (e.g.: TASK4 REPO), and print the report. Exit to the Database window.

5. **TASK 5:** Go into the TASK4 query and using File/Save As from the menu, save with a new name (e.g.: TASK5). In the Criteria: cell under the Supplier field, type *Roche*. Save changes and run the query. The query should display a reduced dynaset of just eight records. Exit to the Database window.

Enter design view of the TASK4 REPO report, and Save As TASK5 REPO. Select View/Properties from the menu (or click the Properties toolbar button) to open the Report properties window. (Make sure it is Report properties, and not a properties window relating to a control or section of the report itself). Click on the Record Source bar, and change to TASK5. Double click in top left  to close the properties window. Save changes, and select Print Preview to view the new report. Fagan stocks G9, P2 and P3 drug products from Roche. Total Usage is 12,054.

6. **TASK 6:** This task can either be used as an exercise for students who have covered data flow diagrams or a research task for students new to the process. The reference given in SolveIt! provides sufficient detail for even the novice student to produce a serviceable data flow diagram. Students' answers on this task will vary greatly in quality, and it is a good way of differentiating between students.

Figure 1 contains a first-level data flow diagram for the pharmacy system using the conventions described in the Laudon and Laudon reference. Items to check include the logic and numbering of processes, data stores and outputs (i.e.: repeats, sticky label, receipt).x

Detailed Instructions for Database Case 5 Using Access


These instructions apply for both Access for Office97 and Access for Office 2000 users.

FILE: CASE5.MDB

1. Create a new database, and then save it. Now open DBASES.MDB on the SolveIt! diskette, and copy and paste the CHAPMAN data table into the new database. In Table Design view, format the Alumni and Male fields so that they will display as Yes or No in Datasheet view. Save changes, and exit to the Database window.

2. **TASK 1:** Create a new query based on the CHAPMAN table. Include fields GPA, GMAT, Name_Last and School. Under GPA field, click on the Sort: cell and choose Descending order. Repeat this action for the GMAT field. Save the query (eg: TASK1A), and then run the query.

The resulting dynaset produces a listing, firstly sorted in descending order by GPA, and then within this, by descending GMAT. Print out this listing.

From design view of the TASK1A query, save query with a new name (e.g.: TASK1B). Click on the GPA field in the Field: bar of the QBE Grid. Click on the  button and from the drop down list, replace GPA with Refs. Run query to test that the sort order sequence is working. Save changes, and send to print. Exit (F11) to the Database window.

3. **TASK 2:** Open the CHAPMAN table in Datasheet view. Click on any record in the Name_Last field to make it the current field. Select Edit/Find from the menu.

If you are using Access 2000, select Name_Last from the Look In drop-down list of the Find and Replace dialog box. Type a last name (e. g.: Hunter) in the Find What text box and click the Find Next button. Access immediately moves to the first occurrence of this last name. Walk students through the other find combinations available in the Find and Replace dialog box.

If you are using Access 97, type a last name (e.g.: Hunter) in the Find What text box and click the Find First button. Access immediately moves to the first occurrence of this last name. Clicking on the Find Next button will find the second Hunter record. Walk students through the other find combinations available in the Find dialog box.

4. **TASK 3:** In design view of the TASK1A query, drag the Alumni field to the next empty field column on the Field: bar in the QBE Grid. Type *Yes* in the Criteria cell of the Alumni field. Save the query with a new name (e.g.: TASK3) using File/Save As from the menu. Run and print the query. The resulting dynaset should contain just 9 records.

5. **TASK 4:** Create a new query based on the CHAPMAN table. Include fields Name_Last, GPA, Refs and Major. Save the query (e.g.: TASK4) and exit.

From the Database window, create a new report based on the TASK4 query. Select the Report Wizard option from the New Reports dialog window. Make the following choices:

- Include all fields
- Click Next
- Group on Major
- Click Next
- Click Next
- Sort on GPA
- Click Next
- Set to Portrait orientation
- Click Next
- Give report a Title (e.g.: Majors Report)
- Click Finish

The report will need some amendments done in design view. Switch to report design view: click on the Major control in the Major Header section and widen. Select View/Sorting and Grouping from the menu (or click equivalent toolbar button). Click to highlight Major in the Sorting and Grouping window. Then click on the Group Footer bar in the Group Properties section, and set to No. Set sort order for GPA field to descending. Double click in top left to close and save. Delete all controls in the Report Footer section (meaningless in this report).

Save report (e.g.: TASK4 REPO), go to Print Preview, and send to print.

6. **TASK 5:** The simplest way to handle this task is to create a new query based on the CHAPMAN table which includes the Major, GPA, GMAT and Refs fields. Click on the Σ Totals toolbar button, and replace Group By in the Totals: bar in the GPA, GMAT and Refs fields with Avg. Save (e.g.: TASK5) and run the query. Access returns the averages of all Major types for

the 3 numeric fields. (You may need to format the field properties of the 3 numeric fields to Fixed with 2 decimal places).

7. **TASK 6:** A number of improvements are possible, including:

- adding a prospective student number to uniquely identify each candidate, and reduce future problems associated with duplicate last names
- creating a main menu of major tasks to improve the effectiveness of the system. This is simple to do in Access with a form and a few linked command buttons.
- adding a Memo field to the CHAPMAN table so that comments and notes can be made against each record as needed.

Detailed Instructions for Database Case 6 Using Access

These instructions apply for both Access for Office97 and Access for Office 2000 users.

FILE: CASE6.MDB

1. Create a new database, and then save it. Now open DBASES.MDB on the SolveIt! diskette, and copy and paste the SONIC_A and SONIC_B data tables into the new database.
2. **TASK 1:** Create a new query. Add the SONIC_A and SONIC_B tables. To establish a link between the two tables, we need to join on the common field Org. Click on ORG in the SONIC_A fieldlist and then drag across to ORG in the SONIC_B fieldlist. Access draws a connection line, which creates a link known as an "equi-join". An equi-join selects all the records from both tables that have the same value in the field selected as being common to both.

Select View/SQL to see the equivalent of this action in Access Basic code. Select View/Query Design to return to query design view. Select and drag the following fields into the QBE Grid:

From SONIC_A:
Name_First
Name_Last

From SONIC_B:
Organization
Address
City
State
Country
Zip

Lastly, drag ORG from the SONIC_A fieldlist into first QBE field column. Do View/SQL to see what difference this action has made to the programming code. Select View/Query Design to return to query design view. Run the query and view the expanded dynaset. The query should contain 28 records.

Access has matched and merged the address data in the SONIC_B table with the name data in the SONIC_A table to create a composite file containing non-duplicative information from both. Save the query (e.g.: Task1).

3. Send the TASK1 query listing to print.

4. **TASK 2:** Create a new report based on the TASK1 query. Select the Report Wizard option from the New Reports dialog window. Make the following choices:

- For Fields, select: Organization, Name_Last and Name_First
- Click Next
- For Group, select Organization
- Click Next
- Click Next
- For Sort, select Name_Last
- Click Next
- Accept the default presentation style and set to Portrait orientation
- Click Next
- Enter a Report Title (e.g.: Marketing Report) and set Calculate % to off.
- Click Finish

Save the report (e.g.: TASK2/3 REPO).

5. **TASK 3:** Send TASK2/3 REPO to print, and return to the Database window.

6. **TASK 4:** Create mailing labels. Create a new report based on the TASK1 query, and using the Mailing Label report wizard. Set up the design of the label by doing the following:

- Choose the Label size
- Click Next
- Choose appropriate Font and Size
- Click Next
- Select Name_First, click > to lodge in label window
- Click Space
- Select Name_Last, click >, then click NewLine
- Select Organization, click >, click NewLine
- Select Address, click >, then click NewLine
- Select City, click >, Add a comma, click Space
- Select State, click >, Add a comma, click Space
- Select Country, click >, Add a comma, click Space
- Select Zip, click >, then click Next to finish label entry

Choose fields to Sort on:

- Sort Order = Organization, then click >
 Name_Last, then click >
- Click Next

-
- Click Next
 - Click Finish

Save report (e.g.: TASK4 REPO) and send to print. Return to the Database window.

7. **TASK 5:** Go into design view of the TASK1 query, and save with a new name (e.g.: TASK5). Under the Criteria: and or: cells of the Country field, enter the text USA. Delete the Name_First, Address, City, State, and Zip fields in the QBE Grid as they are not needed for this task. Running the query at this stage should produce a reduced dynaset of 13 records (i.e.: all USA records).

Return to query design view, and from the SONIC_A fieldlist, drag the MUSINTA and MUSINTB fields onto the QBE Grid. In the first Criteria: cell of MUSINTA enter 64 or 77. In the or: cell of the MUSINTB field, enter 64 or 77. Rerun the query. This time there should only be 7 records in the dynaset.

Save changes, and exit to the Database window.

8. **TASK 6:** This task is relatively simple to achieve. Firstly create a new data table (e.g.: CODES) and in the structure add a new number field (e.g.: MUSCODES). Then add a new text field (e.g.: MUSDECODE). Save changes. Switch to datasheet view of the CODES table. Open the SONIC_A table in datasheet and select all contents of the MUSINTA field. Copy and paste contents to the MUSCODES field of the CODES table. Sort the MUSCODES field of the CODES table by ascending order. Delete all duplicate records. There are 19 music interest codes. Save changes. Enter music interest descriptions against each code in the MUSDECODE field. Save changes. The MUSCODES field of the CODES table can now be joined to the MUSINTA and/or MUSINTB fields of the SONIC_A table in any query where it is relevant to show music interest descriptions.

Detailed Instructions for Database Case 7 Using Access

These instructions apply for both Access for Office97 and Access for Office 2000 users.

FILE: CASE7.MDB

Note: One new function is introduced – the Action Query.

1. Create a new database, and then save it. Now open DBASES.MDB on the *SolveIt!* Diskette, and copy and paste the 1_MONTH, 2_MONTH, 3_MONTH and CONTRACT data tables into the new database. Do not create primary keys at this stage, as this will cause problems during execution of the APPEND sequence.

TASK 1: Create a blank table containing the structure of 1_MONTH, 2_MONTH or 3_MONTH. From the database window, click and highlight one of the above, and click Copy. Click Paste. From Paste Table As window, type the name of the new table, e.g.: ALLMONTH, and click the Structure Only radio button. Click OK.

Create an Append Query to add all records from 1_MONTH, 2_MONTH, and 3_MONTH into the ALLMONTH table. From the database window, click Query and then New. Add the 1_MONTH, 2_MONTH and 3_MONTH tables to the Query window. Select Query/Append to turn the Select Query into an append Action Query. Table Name will be ALLMONTH. Click OK.

Click on * (all fields) in the 1_MONTH field list, and drag down to the QBE Grid Field bar. Click on Datasheet button to test query execution. Return to Query Design view, and click Run. Nine records should be appended to ALLMONTH.

Delete 1_MONTH.* from the Field bar. Click * on the 2_MONTH field list and drag to Field Bar. Test query execution, return to Query Design view, and Run the query. Eight records should be appended. Repeat above procedure for the 3_MONTH table. Nine records should be appended.

Save query (e.g.: APPEND TASK1) and return to database window using F11. ALLMONTH should now contain 26 records. Click and highlight Name_Last field and Sort Ascending to better appreciate the effect of the append sequence.

2. **TASK 2A:** Create a new Query based on the ALLMONTH table. Select and drag all fields down into the Field bar of the QBE Grid. Create calculated fields for each of the following (required syntax is exactly as shown below):

GROSSPAY: [HRSWORKED]*[HRLY_RATE]
FEDTAX: [GROSSPAY]*0.174
STATETAX: [GROSSPAY]*0.08
FICA: [GROSSPAY]*0.027
NETPAY: [GROSSPAY]-[FEDTAX]-[STATETAX]-[FICA]

This can most easily be achieved by directly typing each calculation into the relevant field on the Field bar (click the right mouse button and use the Zoom menu option to assist). For each of the above, set the Field Property to Currency format (click the right mouse button, select Properties, and choose Currency). Make any necessary Column Layout changes (i.e.: narrowing). Sort the Name_Last field in Ascending order. Run and Save the Query (e.g.: ALLMONTHQ). Return to the database window using F11.

3. **TASK 2B:** Using report wizards, create a Tabular Report based on the ALLMONTHQ query which includes the following fields: MONTH, ID, NAME_LAST, GROSSPAY, FEDTAX, STATETAX, FICA and NETPAY. We have already sorted on Name_Last. Give the report a title (e.g.: Quarterly Report). Save the report (e.g.: QUARTER). Display/print in landscape orientation.

For a more professional appearance to this report, go into Report Design view and edit the ID, GROSSPAY, FEDTAX, STATETAX, FICA and NETPAY fields in adjust the alignment in the Page Header, Detail and Report Footer sections. The Sum([ID]) notion in the Report Footer section and the 2 horizontal lines immediately above this should also be deleted.

4. **TASK 3:** Using report wizards, create a Summary report (e.g.: Quarterly Report (Summary)) based on the ALLMONTHQ query. Save the report (e.g.: GRAND). Display/print in portrait orientation. Note: ignore the first wizard window when creating this report. To give this report a more professional appearance, in Report Design view, go into the Report Footer, and alter the data properties Format bar (right click and then select Properties) of each field to display as Currency.

4. **TASK 4:** From Design view of the QUARTER report, select the Sorting and Grouping tool. In the Sorting and Grouping Window, do the following:

	Field/Expression	Sort	Group Header	Group
Footer				
	ID	Ascending	YES	YES
	MONTH	Ascending	YES	YES

To get Subtotals, Copy and Paste the row of Sum calculations for GROSSPAY, FEDTAX, STATETAX, FICA and NETPAY in the Report Footer section into the ID Footer Section.

Run the report. Save the report with a new name (e.g.: TAXSORT) using File/Save As. Return to the database window.

5. **TASK 5:** From the database window, select the ALLMONTHQ query. In query design view, click on Show Table, and add the CONTRACTS table to the Query window. From the fieldlists section of the query window, click and drag a join between the two ID fields.

From the CONTRACT table, drag the NAME_FIRST, ADDRESS, CITY, STATE and ZIP fields down into the Field bar of the QBE Grid. Save the query with a New Name (e.g.: File/Save As MERGE (Task5)). Return to the database window, and create a new report in Tabular format, choosing only those fields required in Task 5. Grouping by Name_Last or ID could be useful.

Detailed Instructions for Database Case 8 Using Access

These instructions apply for both Access for Office97 and Access for Office 2000 users.

FILE: CASE8.MDB

No programming skills are needed to solve this case, although plenty of code is generated behind the scenes as a result of the various actions performed. This case provides a good introduction for students in using Access to build an integrated system, and to ease into the macro and Access Basic programming environments. Two new concepts are introduced - Forms and Macros - and we also look at constructing Update and Parameter queries. The solution to Case 8 is based on a series of forms and queries, linked by simple macros and event procedures.

1. Create a new Access database, and copy and paste the CARS data table from DBASES.MDB into it. Creation of primary keys is optional.

Switch to CARS table design view, and change the format of the LIMITED field in the Field Properties section to Yes/No. Change the Data Type of the CHARGE field to Currency, and the Field Properties format to Fixed and two Decimal Places. Save the changes, and exit to the Database window.

2. **TASK 1:** This firstly involves the creation of two Forms. From the Database window, click Form and New. Base the form on the CARS table, and select an AutoForm wizard. Access automatically generates a simple form showing all fields in the CARS table. Save and call this form SHOWALL.

In SHOWALL design view; click to select CARS in the form Header section. Right click the mouse button, and select Properties. Change the Caption to "Pickering Rentals". Double click in top left to save, close Properties window, and return to form design view. Exit to the Database window.

Create a second form. Click Form and New, and select Design View. Base the record source for the new form on the CARS table. In design view, select VIEW/FORM HEADER/FOOTER to add a Header to the new form. Click on the Label toolbox button, and then click in the Header Section. Type a heading for the form e.g.: Pickering Rentals: Transaction Codes Check Screen. (The Ctrl/Enter key sequence will give you a multi-line label box). Change box and font size to suit.

Click on the Text Box toolbox button, and then click in the Detail section of the form. This will create an unbound text box and label. Click on the label, and then click right and select Properties. Change the Caption to: Enter a Transaction Code and Click the Record Show Button. Double click in top left to save, and exit back to form design view.

Click on the unbound text box, and then right click and select Properties. Change the Name to Check Codes. Scroll down, click on the Validation Rule bar and enter \geq "S24154" And \leq "S24170" to set up a valid transaction code range. (Right click and select Zoom if you need more room to type). Click on Validation Text bar and enter in some infringement text (e.g.: The Transaction Code you have entered is Invalid. Click OK and ReEnter).

Click on the Status Bar Text line and enter the text: Please Enter the Transaction Code. Double click in top left to save Property changes, and return to design view. Save your form (e.g.: CODE CHECK).

3. Next, we create two Command Buttons. One that links the current form through an event procedure to the SHOWALL form and displays the relevant records, and one that Exits the code check sequence and returns to the database window.

Creating an event procedure in a Report or Form without having to directly do any programming is done through the Control Wizards function. Click on the Control Wizard toolbox button, or select VIEW/TOOLBOX from the menu. Click the Command Button tool in the toolbox, and then click on an empty section of the detail section in the CODE CHECK form.

The first of several Command Button wizard dialog boxes appears. Make these choices as you proceed through the dialog boxes:

- Under Categories, select Form Operations
- Under Actions, select Open Form
- Click Next
- Select SHOWALL as the Form to Open
- Click Next
- Choose to Open the Form and Find Specific Data
- Click Next
- From CODE CHECK, highlight the Check Codes control
- From SHOWALL, highlight the Transact field and click the <-> button to match the fields
- Click Next
- Select Text, and change the button caption to Show Record
- Click Finish to generate the Command Button

Save the CODE CHECK form again.

Click on the Show Record button, and then right click and select Properties. Scroll down the list to the On Click: bar. Notice that Access has appended an Event Procedure to this line. Click in the On Click: bar, and then click on the ... Build button. Access displays the event procedure code behind the Show Record button. Exit code window, and double click in top left to return to form design view.

4. **Time to Test:** Switch to Form View. Enter a Transaction Code (e.g.: S24154), and click the Show Record Button. The sequence works, but there is no easy or immediate way to run the procedure again, or to exit from SHOWALL. We need to add an Exit and ReRun button to SHOWALL.

Change to design view in SHOWALL. Create another command button using the control wizards to Exit SHOWALL and ReRun the code check sequence. This procedure will be based around closing a form. Save the changes to SHOWALL, return to form view, and test the new button. It should close SHOWALL and return to Code Check.

Finally, we need a graceful way to exit the CODE CHECK procedure and return to the database window. Create another command button using the control wizard. (Hint: it's identical to the one just created in SHOWALL). Save and test the button, by clicking and exiting to the database window.

Click Form and Code Check and Open to test and run the procedure several times from the beginning. Also enter a code not in the valid range to make sure the validation procedure is working. With two simple forms, and three easy to generate command buttons, we have created a robust, easy to use Transaction Code checking program.

5. **TASK 2:** Task 2 is the most complex in this case, but again Access handles it easily without obvious recourse to programming. One of the premises is that Pickering Rentals should have a system that minimises intervention of set-up procedures by the user. For instance, changes to car rental rates should be able to be done from a table, rather than requiring a user to make risky edits to programming code. For this reason, Task 2 firstly creates a new table - RATES1 - that contains the following fields:

Model No	(same data as Model field in CARS table)
LRate	(Limited rental rates)
Ladd	(Limited additional charge rates)
ULRate	(Unlimited rental rates)

Create this new table making sure that that Model No is the same Data Type as Model in the CARS table. Set the remaining fields to a Currency Data Type. Save the table as RATES1, and say No to Create a Primary Key. Enter the data listed in the table shown under the Task 2 problem in Case 8. Return to the Database window.

6. Create a Relationship (using the Relationships tool bar button) between RATES1 and CARS using Model No and Model as the join. Click on the Join Properties button, and select Join Type 3 to include all records from CARS and only those in RATES1 where the join fields are equal. Click Create, and return to the Database window.

7. **Create 3 Update Queries:** Query 1 - Unlimited Transactions. Create a new query based on the CARS and RATES1 tables. Drag Charge, Model and Limited fields down to the Field: bar of the QBE Grid. Under Model, type in the Criteria: bar S or M or L or X or W. Under Limited, enter No into the Criteria: bar.

Select Query/Update from the menu or click the Update button on the toolbar. In the Update To: bar under Charge, enter the expression $[ULRate] * [Days]$. Save the query (e.g.: RATES CALC1), and then run the query. Access should confirm that 8 records will be updated. Click OK. Exit the query, and open the CARS table to see the difference made to the Charge column.

Query 2 - Limited Rate <100 Miles. Exit CARS and go into design view of the RATES CALC1 query. Give it a new name by saving as RATES CALC2. Under the Limited field, change the criteria to Yes. Click and drag the Miles field down into the Field Bar: of the QBE Grid, and enter <100 in the Criteria Bar:. Change the expression under Charge to: $[LRate] * [Days]$. Save the query, and run. Access should update 7 records this time.

Query 3 - Limited Rate >100 Miles. Save RATES CALC2 with a new name e.g.: RATES CALC3. Change the Miles criteria to >100. Change the expression in the Update To: bar of Charge to: $([LRate] * [Days]) + (([Miles] - 100) * [LAdd])$. Save query and run. Access should update 2 records. Exit to the Database window, and see the difference to the Charge column in the CARS table.

8. **TASK 3:** Task 3 introduces the parameter query. Create a new table (e.g.: DEPREC) and include the following fields and data types:

Model Type	Text
Purchase Price	Currency

Deprec Rate	Number (format as Percent in Field Properties)
-------------	--

Model Type contains the same data as Model No in the RATES1 table. Purchase Price and Deprec Rate data are taken from the table shown at the bottom of the Task 2 problem in Case 8.

Create a new Query, based on the CARS and DEPREC tables. Drag and create a join between the Model and Model Type fields. Select and drag the Model, Reg_No, Odometer and Date Fields into the Field Bar: of the QBE Grid.

Create a calculated field called Calc Deprec which include the following expression in the Field Bar: $([\text{Deprec Rate}]/100)*[\text{Purchase Price}]*[\text{Odometer}]/6000$

Create a parameter on the Reg_No field by typing in the Criteria Bar the following statement: [Enter the Rego:]. When the query runs, this will prompt the user to enter a car registration number.

Save the query (e.g.: DEPREC QUERY), and exit back to the Database window. Create a new form through the form wizards in Tabular format, based on and including all the fields in the DEPREC QUERY. Save the form (e.g.: DEPREC FORM). Using macros or control wizards, add two command buttons to the new form:

- One that activates DEPREC QUERY calculates in response to the parameter entry
- One that exits from the Deprec Form

Save and test the procedure.

Detailed Instructions for Database Case 9 Using Access

These instructions apply to both Access 2.0 and Access for Office97 users.

FILE: CASE9.MDB

1. Create a new database, copy and paste the data tables BOOKINGS, ROOMS and VISITORS from DBASES.MDB into it. View fields in each table.
2. Make a copy of the ROOMS table using Edit/Copy and Edit/Paste from the menu. Call the new table ROOMS LOOKUP. This will later be used as the basis for a lookup table in the registration procedure, and for a Task 4 menu item.
 - (a) Amend the structure of the ROOMS table by deleting the Empty and Feature fields. Add a Cust No field. Format the Rate field to Currency. *Delete all existing records from the ROOMS table.*
 - (b) Amend the structure of the ROOMS LOOKUP table. Keep only the following fields: Room_No, Empty, Rate, Features. Format the Empty field to show Yes/No. Format the Rate field to Currency. Create a simple select query which includes the following fields: Room_No, Rate, Features and Empty. In the Criteria cell under the Empty field, type "Yes". This limits the displays of rooms to only those which are vacant. Save the query (eg: ROOMQ).
 - (c) Amend structure of the BOOKINGS table. Add a Cust No field. Format the Charge field to Currency.
 - (d) Amend the structure of the VISITORS. Add a Cust No field.
 - (e) Create relationships between the four tables using Edit/Relationships from the menu or clicking on the Relationships toolbar button. Create One to Many relationships on the Cust No. field between:
VISITORS and ROOMS
VISITORS and BOOKINGS

Create One to One relationships on the Room_No field between:
BOOKINGS and ROOMS
ROOMS and ROOMS LOOKUP

Save and return to the Database window.

3. TASKS 1 and 2: From the Database window, click on the Forms object and create a Main/Subform using the Form Wizards. Use the VISITORS table as the basis for the Main form, and base the Subform on the ROOMS table. Include the following fields in each form:

Main (VISITORS)	SubForm (ROOMS)
Cust No	Room_No
Surname	Surname
First Name	Rate
	Day_In
	Cust No

Save the Main form as REGO FORM

Save the SubForm as ROOM FORM

(a) Create a Tabular form based on the ROOMQ query. Save this form (eg: AVAILABLE ROOM FEATURES). The form should include the following fields: Room_No, Empty, Rate, Features.

Using the form control wizards, include an Exit control button. Include some simple instructions for use of the form (eg: "Choose a room from the list, and then click on X to turn the room to occupied"). The contents of this form will be accessed in 2 ways:

- through a command button on REGO FORM as part of Task 2
- as an item in the main menu required for Task 4

(b) In Design view of ROOM FORM, click on Room_No control to highlight, then right click and select Properties from the menu. Scroll down to the After Update bar in the Properties window, and create a macro which will lookup the appropriate room rate for a room in the ROOMS LOOKUP table and automatically append it to the ROOMS subform. Use the Set Value and DLookup functions to do this. The macro will contain the following instructions (required syntax is exactly as shown):

Macro Action: Set Value

Item: [Rate]

Expression:

DLookup("[Rate]", "[ROOMS LOOKUP]", "[Room_No]=[ROOMS]![Room_No]")

Save the macro as RATE LOOKUP and return to REGO FORM design.

(c) Create a date control based on the Last_Stay field of the VISITORS table, and link it to the Day_In field of the ROOMS FORM subform.

Create 3 command buttons using the Form Control wizards which will:

- Exit the form
- Add a new registration record
- View available rooms (ie: empty ones).

Include some simple instructions for use of the REGO FORM

(d) Test the REGO FORM for robustness by using it to enter a few new records:

- Click on the Add New Record button
- Enter a Guest ID
- Enter a First_Name
- Enter a Surname (Surname should append automatically to the subform)
- Enter a date
- Click on Room_No in the subform
- Click on the View Available Rooms button, and select a room
- Enter the chosen room number in the Room_No field of the subform (the date from the main form should append automatically to the Date_In field of the subform)
- Click on the Rate field of the subform (the DLookup function from the Room_No field appends the room rate from the ROOMS LOOKUP table)
- Click on the Exit command button
- Open the VISITORS and ROOMS tables to check that records have been added

3. TASK 3: Checking out a Guest.

This task is accomplished through the creation of a series of forms and action queries which are run in sequence through a macro.

(a) The queries needed include:

- An Append query based on the ROOMS table which appends a specified checkout record to the BOOKINGS table. The query includes the following fields: Room_No, Surname, First_Name, Day_In, Customer No. The Surname field will include a *parameter* instruction in the Criteria: cell to: "Enter the Guest Checkout Surname". Save this query (eg: APPEND QUERY1)

-
- A Delete query (based on the ROOMS table) which removes the selected record as per the Append query from the ROOMS table. This query will have the same fields as APPEND QUERY 1. The Surname field will include a *parameter* instruction in the Criteria: cell to: "ReEnter the Guest Checkout Surname". Save this query (eg: DELETE QUERY1).
 - An Update query based on the BOOKINGS and ROOMS LOOKUP tables (these tables should be linked by the Room_No field, as per relationships established earlier in this case). This query calculates the charges owed by the guest who is checking out, and includes the following fields:
 - Surname (with a *parameter* instruction in the Criteria: cell to: "ReEnter the Guest Checkout Surname")
 - Charge (this column includes a calculation in the Update To: cell which determines the difference between dates entered in the Day_In and Day_Out fields in the BOOKINGS table, and then multiplies this result by the Rate in the ROOMS LOOKUP table).

This calculation is performed using the DateDiff function. The required syntax is exactly as follows:
DateDiff("d",[Day_In],[Day_Out])*[Rate]

Save the query (eg: CHECKOUT QUERY).

(b) Next, create the forms needed for the checkout procedure. This solution uses three forms to do this:

- Create a new blank form which will be used to run the checkout procedure on the click of a command button. This form will also be used as an item in the Task 3 Main Menu. Clicking on the Checkout Procedure command button activates a macro (CHECKOUT MAC) which sequentially opens and runs each of the queries associated with Task 3.

Include an Exit command button to close the form. Include some instructional text on what the form does (eg: "This procedure will firstly transfer the selected Checkout Record from the ROOMS table to the BOOKINGS table, before deleting it from the ROOMS table, calculating the appropriate charge for the exiting guest, and then displaying all checkout details"). Save the form (eg: CHECKOUT1).

-
- Create a form to calculate the charges and activate a display of all relevant checkout details. Base the form on the BOOKINGS table and include the following fields: Surname, Day_In and Day_Out. The form will automatically display Surname and Day_In data from the underlying table. The user should enter the relevant checkout (Day_Out) date.

Create 3 command buttons which:

- Saves the Current Record. On Click, this saves the current record after data has been entered in the Day_Out field.
- Calculates Charges. On Click, this activates the CHECKOUT QUERY created earlier, performs the calculation, and saves the updated information to the underlying BOOKINGS table.
- Shows Checkout Data. On Click, this opens the CHECKOUT2 form (the creation of which is described below), and displays all relevant checkout details

When the form is opened, the Surname and Checkin date is displayed. The user then enters the Checkout date, and clicks on the Save Current Record Button. Next the Calculate Charges button is clicked, before finally clicking on the Show Checkout Data button. Save the form (eg: CHECKOUT1A)

- Create an Autoform based on the BOOKINGS table. This form shows all relevant checkout data (ie: all fields in the BOOKINGS table). Include an Exit command button which closes the form. Save the form (eg: CHECKOUT2).

After completion of Task 4 (creation of a Main Menu form), create an Exit macro (EXIT MACRO) for the CHECKOUT2 Exit button which closes all forms associated with the checkout procedure, and returns to the main menu.

4. **TASK 4:** Create a Main Menu

This task is simply accomplished in Access. From the Database window, create a new blank form. Save the form (eg: MAIN MENU). Using the Form Control wizards, create four command buttons which perform the following functions:

- Register a Guest. On Click, this button opens REGO FORM
- Checking Out. On Click, this button opens the CHECKOUT1 form
- List Available Rooms. On Click, this button opens and displays the contents of the AVAILABLE ROOM FEATURES form

-
- Exit the Program. On Click, this button closes the MAIN MENU form and returns to the Database window

Save all changes, and then run and test the command buttons.

5. **TASK 5:** Further improvements.

The first improvement has already been implemented (although not really used) by the establishment at the beginning of this case of Customer No fields as a unique guest identifier. So far, this mechanism has only been used in this case to illustrate different types of relationships between fields in different tables.

A second improvement can be achieved by adding extra fields (Surname and Reserved) to the ROOMS LOOKUP table. Create a new query based on this table using just the Surname and Reserved fields, and where the Criteria: cell in the Reserved field shows "Yes". Add an additional command button to REGO FORM to enable the user to check the reserved status of rooms.

Detailed Instructions for Database Case 10 Using Access

These instructions apply for both Access for Office97 and Access for Office 2000 users.

FILE: CASE10.MDB

1. Access database files can be copied, renamed or deleted by using the Windows File Manager utilities program or the appropriate DOS commands.

Take a copy of the file CASE6.MDB and rename as CASE10.MDB.

2. **TASK 1:** Create a new blank form as the basis for the menu using the Form wizards feature. In Form design view and using the Forms toolbox, create four command buttons to establish the menu items required for this task. The four items are:

Add a New Customer
Add a New Company
Produce a Complete Mailing List
Exit the System

Three of these buttons will be created by just clicking on the command button tool. Links to these buttons are established in a later task. The fourth button - Exit the System is created in combination with the Form control wizards. From the wizard window, select Form Operations and the Close Form action.

Position, resize and name each button on the form. Use the View/Form Header/Footer option from the menu to create a form header. Use the Label toolbox button to add some header text to the form (eg: "Anabasis Marketing Database - Main Menu"). Save the form (eg: MAIN MENU), switch to Form View, and test that the Exit the System button works by clicking and exiting back to the Database window.

3. **TASK 2:** Most of the work was completed during Case 6. The creation of two AutoForms must be completed first, since these will be used as the object source link for option buttons 1 (Add a New Customer) and 2 (Add a New Company) on the Main Menu.

- (a) Menu Option1: Add a New Customer.

From the Database window, create an AutoForm based on the SONIC_A table using the form wizards. This will form the basis for the data entry screen required for option 1. Switch to Form design view and using the Form control wizards include an:

- Exit command button (same as that created in Task 1)
- Add New Record button (select Record Operations and Add New Record)

Use the VIEW/Form Header/Footer option from the menu to create a form header. Use the Label toolbox button to add some header text to the form (eg: "Add a New Customer"). Save the form (eg: ADD CUSTOMER), switch to Form View, and test that the Add New Record button works. Test that the Exit button works by clicking and exiting back to the Database window.

(b) Menu Option 2: Add a new Company.

Repeat the same procedure as for (a) above. The AutoForm will be based on the SONIC_B table, should have some header text (eg: "Add a New Company"), and should include the same Exit and Add New Record command buttons. Save the form (eg: ADD COMPANY), switch to Form View, and test that both command buttons are working.

(c) Link the two forms to their Main Menu buttons.

In design view of the MAIN MENU form, right click on Option Button 1 (Add New Customer). Select Build Event from the menu, and then Macro Builder. Give the new macro a name (eg: CUSTOMER MACRO). This is a simple macro that uses the Open Form action and the ADD CUSTOMER form. Save macro changes, and exit the macro. Access automatically adds the new macro to the On Click line of the Add New Customer button properties sheet. Test the button by switching to Form View of the MAIN MENU form and clicking on the customer button.

Repeat the above procedure for Option Button 2 (Add a New Company). Give the macro a name (eg: COMPANY MACRO). Base the macro Open Form action on the ADD COMPANY form. Save macro changes and exit the macro. Switch to Form View of the MAIN MENU form, and test the newly linked button.

4. **TASK 3:** This task requires the creation of an Autoexec Macro which opens the MAIN MENU form. Clear instructions for doing this are provided in the tutorial which accompanies this case.

5. **TASK 4:** To complete this task, two different types of queries need to be created.

(a) Create a Union Query, which combines SONIC_A and SONIC_C. From the Database window select the Queries object and then New Query. When the Query window appears, close the Show Table dialog box. From the menu click on Query/SQL Specific and then select the Union option. Add the following statement to the SQL Union window:

```
SELECT [NAME_LAST], [NAME_FIRST], [ORG]
FROM [SONIC_A]
UNION SELECT [NAME_LAST], [NAME_FIRST], [ORG]
FROM [SONIC_C]
ORDER BY [ORG];
```

Be sure to complete the Access syntax correctly by including a semi colon at the end of this statement. Save this query (eg: TASK4A). Run and test the query. SONIC_A has 28 records and SONIC_C has 12 records. There are 5 records which appear in both files. After execution of the Union query there should be 35 records and no duplicates.

(b) Create a Select Query, which joins the Union Query to the file SONIC_B. From the Database window select the Query object and then New Query. As with Report and Form objects, Select queries can be based on tables or on other queries. From the Show Table dialog box, click on the Both tab to see a list of all tables and queries. Select SONIC_B and then click on Add. Next select the TASK4A query and then click on Add. Click on Close to exit the Show Table dialog box.

Create an equi-join using the ORG field from each fieldlist. Select and drag the following fields into the QBE Grid:

From SONIC_B:
Organization
Country

From TASK4A:
Name_Last
Name_First

Run the query. The query should contain 35 records with no duplicates. Save the query (eg: TASK4B).

6. **TASK 5:** Most of the work for this task was performed in Case 6. Open the TASK 2/3 REPORT created in Case 6 in Report design view. In the Report Header section, amend the title of the report (eg: "Full Mailing List

Report”). Change the record source for the report to the TASK4B query. Add the Country field. Save changes, and return to the database window.

Repeat the procedure described in 3(c) above to link the TASK 2/3 REPORT to the third Option Button (Produce Mailing List) on the MAIN MENU form. Give the macro a name (eg: JOIN MACRO), and use the Open Report macro action and the TASK 2/3 REPORT. Save changes, switch to Form View of the MAIN MENU form, and test the button.

7. **TASK 6:** Add new Sales and Number of Sound Recordings fields to the SONIC_A table. Add data for these new fields to existing records. Calculations for total sound recordings and total sales can be performed by creating a new query (based on SONIC_A), and then creating new calculated fields to compute this data. Link this query to the MAIN MENU form by creating a new button (eg: SHOW SALES DATA).