Access exercises I

1 Understanding Databases

Open the Access database **Student Records 1.accdb** for these exercises

1. Open the **Students** table in datasheet view and enter the following new student either by navigating to the empty field or using the 'New record' button. Note the pencil symbol when editing:

studentID	**forename**	**surname**	**email**
10510	Paul	Hitch	p.hitch@gmail.co.uk

2. Enter yourself as a new student. Save by moving to a different record or using the Save button.

   If you left the **studentID** field empty you will receive a warning message. This is because **studentID** is the primary key and must be included in any record.

   Either enter a made-up **studentID** (10505) or delete the record.

3. Find student number **10330**, Tara Maples. Correct her email to **yahoo.com**.

4. Close **Students** and open the **Modules** table. Find the Plasma Physics module (**moduleID PHY201**) and set the **roomID** to G/A/114.

5. Correct the spelling of POL103 to 'War and Peace'.

Extension

6. Prof Leo Richards is the new tutor for the module CSC203, AJAX Development. Locate his TutorID (**Tutors** table) and update the tutorID for this module in the **Modules** table.

7. Joseph Smith has received a mark of 79 in Nuclear Chemistry. Enter this into the **ModuleChoices** table.

2 Query Essentials

Continue using the Access database **Student Records 1.accdb** for these exercises

1. Create a new query in Design view. Add the **Students** table. You may want to resize the table so you can see all the fields listed.

   Add **studentID**, **forename** and **surname** to the grid.

   (Try using both the double-click and drag-and-drop methods and see which you prefer)

2. Use the View button to switch to Datasheet view. Find student 10091 and correct her name to Suzy Watson.

   Note: Even though queries do not store data, the results of a query are from the underlying table, and a change made here is changing the data stored within the table.)

3. Switch back to Design view. Apply a Sort to order surnames alphabetically and check the results in the datasheet view.

4. Amend your query to sort first by surname, then by forename.

   (Hint: Access applies sort orders from left to right. You will need the surname field to appear first in the Query By Example grid)

5. The field **currentYear** indicates the year of study of students. Add **currentYear** to your query and apply a filter to only show only 1st years.

6. Modify the query to only show 1st years with a surname "Jones"
7. Save the query as **qryFirstYears**
8. Create a new query based on the **Students** table that will show the names and ID of all female 2nd year students.
9. Edit the query design to show female 2nd year students with the surname "Chapman".
10. Save the query as **qrySecondYears**

Extension

11. Create a new query that will display the studentID, name and date of birth for all students born on 26/09/1995.
12. Modify the query to find students born in 1995 (ie between 01/01/1995 and 31/12/1995).

    Hint: See Conditions, ranges and wildcards
13. Modify the query again to only show 1st years born in 1995. Also include their email address and sort by surname.
14. Save the query as **qryDob**
15. Create a query to show all 2nd and 3rd year students.
16. Save the query as **qrySecondAndThirdYears**

2.1 Multi-table queries

Continue using the Access database **Student Records 1.accdb** for these exercises.

1. We want to view information about the modules the students are taking. This will require data from two related tables, Students and ModuleChoices. Both tables contain studentID fields which are linked with a 1-to-many relationship.

   Create a new query and add only the **Students** table. Add **studentID** and both name fields; view the datasheet.
2. Add the table **ModuleChoices** and view the datasheet again.

   There are about 3 times as many results as before. This is because each student studies several modules and is listed once for each module they take.

   Add the **moduleID** field and view the datasheet again to see the different modules each student is taking.
3. Return to the design and filter the results to show details for **studentID** 10175. (David McDonald).
4. It would be helpful to see the module titles, and you may also need to view other module information. To do this we need to add a third table to our query.

   Add the **Modules** table to the design and include the **moduleName** and **credits** fields.

   View the datasheet again.
5. Clear the studentID from the criteria and save the query as **qryModules**.

Extension

6. Create a new query that will list each module and the tutor who teaches it. Show the module ID, module name and term along with the Tutor's ID, full name and email. (44)
7. Modify the design to display only History modules.

   Hint: Use wildcards to find only the modules with IDs which start with HIS.
8. Modify the query to show all modules again. Add the room ID and capacity. You will need to add a 3rd table to do this
9. Why do you think adding the rooms table to the query has reduced the number of records returned? Looking carefully at the Modules table may help.
10. Modify your query to show the college that the rooms are in.
11. Save the query with a suitable name.

3 Editing Table Design

3.1 Task Design

Open the Access database **Student Records 2.accdb** for these exercises.

1. Open the **Modules** table in design view. Examine the fields in this table, paying attention to the data type of each, and the different field properties available.
2. Set the field size of **moduleID** to prevent codes longer than 6 characters from being entered.
3. The **departmentID** is always a 3 character code. Change the field size, and format it to always display in uppercase.
4. Some modules are compulsory for all students. Add a new field called **compulsory** with **Yes/No** data type to record this. Set the default value to **Yes**.
5. There are never more than 100 credits awarded for a module. Adjust the number field size to the most appropriate type. Set a validation rule to prevent entries higher than 100 and enter suitable validation text.
6. Add a new field **modCreated** to record the date a new module is entered into the database. Format as Short Date and set the default value to be the current date.
7. Add a new field **tutorID** to store which tutor takes this module. Set as **Long Integer**.
8. Save your changes to the table design and switch to datasheet view. Pay attention to the warning messages. Enter an imaginary new module. Try entering:
   - a **departmentID** longer than 3 characters
   - **credits** greater than 100
   - The changes you made should prevent this. You should see your default values in the **modCreated** and **compulsory** fields.

A new table

9. Create a new table in design view called **Tutors**. Add the following fields:

Field Name	Data Type	Field properties
tutorID	AutoNumber	Primary Key
title	Short Text	field size 10
firstName	Short Text	field size 50
lastName	Short Text	field size 50
dateOfBirth	Date/Time	format as short date

10. Switch to datasheet view and enter a new tutor. Note **tutorID** is automatically completed when you start entering data in any other fields.

3.2 Relationships

Use the Access database **Student Records 3.accdb** for these exercises.

1. Open the **Relationships** window from the Database Tools tab. This isn't currently showing all tables in the database. Add the **Departments** and **Tutors** tables.
2. Create a relationship between **tutorID** (Tutors table) and **tutorID** (Modules table) and enforce referential integrity.
3. Create a relationship between **deptID** (Departments table) and **departmentID** (Modules table) and enforce referential integrity. Close the Relationship window, saving the changes.
4. The Politics department have changed and are now the department of Politics, Economics and Philosophy.

   Open the Departments table in Datasheet view and attempt to change deptID from POL to PEP. If you set up the relationship correctly you will receive an error.

   (Note: This is because there are related records in the Modules table that have the department code POL. These records would become 'orphaned' if the corresponding POL name was changed in the Departments table.)

   Use 'Esc' to undo the update.

5. Attempt to delete the Politics department. This will also fail for similar reasons. Close the table.

6. Open the **Relationships** window. Modify the relationship between Modules and Departments to **Cascade Update** and **Cascade Delete**. Close the window.

7. Open the **Departments** table and attempt to change POL to PEP again. You should receive no errors. Open the Modules table. Politics modules will now show this new PEP department code.

   (Note: This is due to the 'Cascade updates' option)

8. Return to the **Departments** table and delete the Politics department. You will receive a warning that because of the cascade delete this will not only delete Politics from the departments table but also delete all related records in other tables. Choose **Yes** and return to the modules table to confirm that all Politics modules have now gone.

   (Note: cascade delete should be used very carefully)