2025

# Databases

Orb education

Dan Collingbourne

| Do This | Section |
|---|---|
| ☐ | 1. Wordlist |
| ☐ | 2. Information Systems |
| ☐ | 3. Introducing Databases |
| ☐ | 4. Designing a Database |
| ☐ | 5. Creating the Database |
| ☐ | 6. Entering Data |
| ☐ | 7. Data Validation |
| ☐ | 8. Sorting and Filtering |
| ☐ | 9. Visual Queries |
| ☐ | 10. Query Criteria |
| ☐ | 11. Wildcards |
| ☐ | 12. Select SQL |
| ☐ | 13. Reports |
| ☐ | 14. Exporting Data |
| ☐ | 15. Skills 1 - School |
| ☐ | 16. Skills 2 - Sports |
| ☐ | 17. Skills 3 - Entertainment |
| ☐ | 18. Skills 4 - Populations |

Data is information. This may include names, places, dates, phone numbers and email addresses. Each field that you set up in a database must be given a data type. For example, names and addresses will be assigned as the type ***Short Text***; ages or goals scored will be set as the data type ***Number***; amounts of money will be stored as ***Currency***.

***Aim:*** *To consider data types and the information we might store.*

Microsoft Access also allows you to use automatic data called ***AutoNumber***. AutoNumber assigns a different number to each record. This is often used as a membership number, with the first person entered being given the number 1, the second number 2 etc. With the use of AutoNumber, it is not possible for two people to have the same membership number.

## Task 1 – Data Types

Some of the available data types are shown below. Link each type of data to the example on the right.

| No | Type | | Example |
|---|---|---|---|
| 1 | **AutoNumber** (numbers filled automatically 1, 2, 3...) | • | • Age |
| 2 | **Short Text** (letters and numbers up to 255 characters) | • | • Membership number |
| 3 | **Number** (includes digits only) | • | • Amount paid |
| 4 | **Currency** (amounts of money) | • | • Registered? |
| 5 | **Date & Time** (date and time information) | • | • First name |
| 6 | **Yes/No** (only 2 possible answers) | • | • Date first played |

## Task 2 – Data Type Examples

The lists below show some information that might be included in a database. Write down whether each field is **Data Type 1**, **2**, **3**, **4**, **5** or **6** from the last task. Two answers have been completed for you.

***Note:*** *Phone numbers e.g. 0123 9876 5432 will be* ***Short Text*** *rather than* ***Number*** *because of the spaces.*

| Personal Details | | Contact Details | | Other Details | | | |
|---|---|---|---|---|---|---|---|
| Last name | 2 | Street address | 2 | Date joined | ___ | Goals scored | ___ |
| First name | ___ | Area/Suburb | ___ | Fees paid? | ___ | Tasks to complete | ___ |
| Date of birth | ___ | Country | ___ | Amount paid | ___ | Game number | ___ |
| Age | ___ | Mobile number | ___ | Position | ___ | Did they play? | ___ |
| Gender | ___ | Email address | ___ | Member number | ___ | Amount owing | ___ |

Queries are used to fetch and organise data from database tables. At this stage, we will only be using something called a SELECT query, which is really a filter that you can edit. Data can be retrieved based on things such as:

***Aim:*** *To create queries using the query builder.*

- ***Tables and Fields*** *You can decide on the fields that are displayed from a certain table.*
- ***Records*** *Records can be filtered by date, the text they contain, numbers over a set amount etc.*
- ***Order*** *The records can be placed in a particular order.*

When a query is executed, the results appear in a datasheet. This is a filtered display of data – the original tables have not been changed in any way. The tasks below use a *query builder* (or query design) to produce some queries.

***Note:*** *We will practice queries in some simple practice databases created for these tasks. Once you have learned the skills, you should create some queries in your own database.*

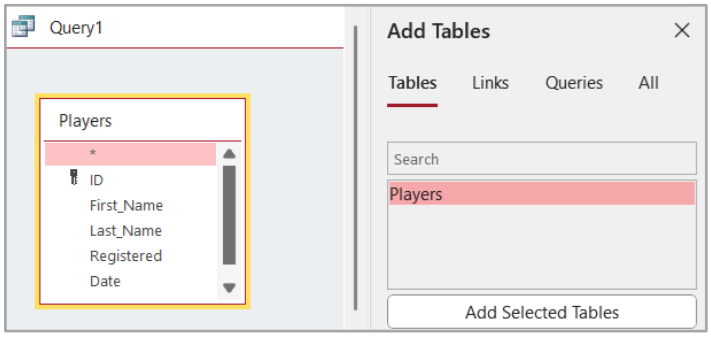
## Task 1 – Building a Query

**a.** Create a database named '**09. Visual Queries**' with a table called '**Players**'. Use the fields and data shown below.

| ID | First_Name | Last_Name | Registered | Date | Played |
|---|---|---|---|---|---|
| 1 | Sarah | Johns | ✓ | 01-Mar-25 | 5 |
| 2 | Gareth | Jones | | | 4 |
| 3 | Amelia | McDermott | ✓ | 01-Mar-25 | 3 |
| 4 | Mark | Collins | ✓ | 08-Mar-25 | 4 |

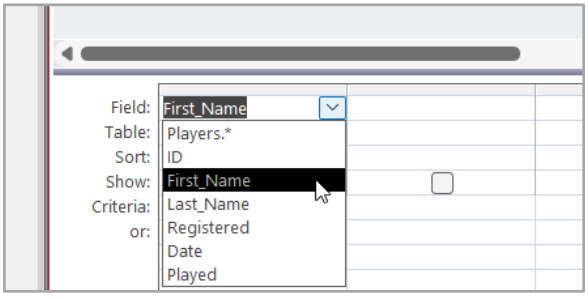
**b.** Select '**Create / Queries / Query Design**'.

**c.** Make sure that the *Players* table is selected in the *Add Tables* panel on the right and click *Add Selected Tables* (or double-click on the table name). A small box showing the fields in your table will appear in the query builder window.




**d.** Working in the lower half of the display, select *First_Name* from the dropdown list in the first column. This field will be displayed first in your results.

**e.** Select *Last_Name* in the second column. Click on the *Sort* box underneath and select *Ascending* from the choices (see the picture on the next page).

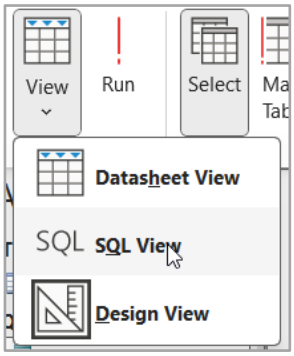


SQL stands for Structured Query Language. It is a form of computer code used to interact with a database. When using MS Access, you do not need to think about the SQL that is being used; you can create a query easily using either the *Query Wizard* or *Query Design* tools. However, it is actually the SQL doing the work behind the scenes. The tools in Access are SQL builders; they help produce the SQL that communicates with the database.

***Aim:*** *To write queries using the SQL programming language.*

If, for example, you open the query *09T1_Played5* from the *Visual Queries* resource, a list of members who have played 5 games is retrieved.

| Field: | First_Name | Last_Name | Played |
|---|---|---|---|
| Table: | Players | Players | Players |
| Sort: | | Ascending | |
| Show: | ☑ | ☑ | ☐ |
| Criteria: | | | 5 |
| or: | | | |




Clicking on the arrow below the *View* icon and selecting *SQL View* (see right) displays the code shown below left.

The code below right is a simplified version of the same SQL. You may type your code in this form and ignore the changes made by Access when you save your query (these are just Access setting things up for more complicated situations you might learn about later).

09T1_Played5

```
1  SELECT Players.First_Name, Players.Last_Name
2  FROM Players
3  WHERE (((Players.Played)=5))
4  ORDER BY Players.Last_Name;
5
```

**Code generated by MS Access**

09T1_Played5

```
1  SELECT First_Name, Last_Name
2  FROM Players
3  WHERE Played=5
4  ORDER BY Last_Name
5
```

**Simplified version of the same code**

The SQL code can be broken down into the four parts below.

| | | | |
|---|---|---|---|
| **1.** | **SELECT** | some fields | *e.g. display First_Name and Last_Name, in that column order* |
| **2.** | **FROM** | a table | *e.g. from the Players table* |
| **3.** | **WHERE** | some criteria is matched | *e.g. only if Played = 5* |
| **4.** | **ORDER BY** | some fields | *e.g. sort by Last_Name* |

# Select SQL (page 2)

## Task 1 – Simple SELECT SQL

SELECT SQL is code that selects data for display. There are also commands for database functions such as UPDATE and DELETE, but we will only consider SELECT statements at this time. Consider the examples below.

| Example 1 | Example 2 |
|---|---|
| **SELECT** First_Name | **SELECT** First_Name, Last_Name, DoB |
| **FROM** Personals | **FROM** Personals |
| **WHERE** Sport = 'Netball' | **WHERE** Vegetarian = Yes |
| **ORDER BY** First_Name | **ORDER BY** DoB DESC |

***Note:*** *Order by 'DESC' means Descending, or starting with the largest. The alternative, 'ASC' (meaning Ascending) is used by default; it does not need to be typed.*

**a.** In *Example 1*, which field will be displayed? _____

**b.** In which table does this field belong? _____

**c.** What criteria must be matched? _____

**d.** How will the data be sorted? _____

**e.** Describe what *Example 2* will display (note that 'DESC' means starting with the highest or most recent).

_____

_____

## Task 2 – Typing SELECT SQL

**a.** Open the database with the *Personals* table from either the *Query Criteria* or *Wildcards* resources. Click '**Create / Queries / Query Design**'.

**b.** Select the *SQL* icon in the top left and name the query '**12SQL_Example_1**'.

**c.** Type the SQL from *Example 1* above. The formatting (e.g. bold) has no effect.

**d.** Switch to *Datasheet View* and check that the results show only the netball players. Close and save the query.

**e.** Repeat this process for Example 2, saving as '**12SQL_Example_2**'.

***Note:*** *If you are searching for a piece of text then it must be placed in single quotes (e.g. 'Netball'). If you are searching for a number, date or a Yes/No value, then it should not be placed in quotes.*

These tasks consider aspects of school life such as the number of students in each year and the products sold in the school canteen. We have assumed you have the basic database skills such as creating tables, forms, queries and reports.

**Skills Check**

| | |
|---|---|
| Validation | ☐ |
| Sorting / Filtering | ☑ |
| Visual Queries | ☑ |
| SQL | ☑ |
| Reports | ☑ |
| Exporting | ☑ |

## Task 1 – School Population

The table shows the number of students in several year groups at school.

| Year Group | Number of Students |
|---|---|
| 7 | 122 |
| 8 | 110 |
| 9 | 125 |
| 10 | 118 |
| 11 | 102 |
| 12 | 98 |

**a.** Create a database named '**15. School Population**' with a single table named '**Year_Groups**'. Add the 3 fields shown in the data dictionary below and enter the above data.

| Field Name | Data Type | Description |
|---|---|---|
| ID | AutoNumber | ID number (completed automatically) |
| Year_Group | Number | Year group |
| No_Students | Number | Number of students |

**b.** Open *Datasheet View* and sort the data by the number of students, starting with the highest.

**c.** Create a new query using '**Create / Queries / Query Design**'. Add the *Year Groups* table and use the settings shown on the right.

| Field: | Year_Group | No_Students |
|---|---|---|
| Table: | Year_Groups | Year_Groups |
| Sort: | Ascending | |
| Show: | ☑ | ☑ |
| Criteria: | =9 Or =10 | |

Save the query as '**Years 9 and 10**'.

**d.** Create similar queries showing each of the selections in the table below.

| Query name | Selection | Display order | Hint |
|---|---|---|---|
| **Years 11 and 12** | Years 11 and 12 only, ordered by *Year_Group,* ASC | Year Group, No_students | = 11 or =12 |
| **Years 7 to 10** | Years 7–10, ordered by *No_Students*, DESC | No_students, Year Group | <11 |
| **Over 110** | Years with more than 110 students | Year Group, No_students | >110 |
| **Above 9 and 100** | Above year 9 with 100 students or more | Year Group, No_students | >9, >=100 |

## Task 2 – The School Canteen

**a.** The table shows the energy and fat content of some of the products sold in a school canteen.

| Product | Energy (kJ per serve) | Saturated Fat (g per serve) | An Everyday Meal |
|---|---|---|---|
| Pizza | 1120 | 9.8 | N |
| Pasta | 614 | 6.9 | Y |
| Noodle Salad | 711 | 4.2 | Y |
| Hamburger | 1240 | 11.1 | N |
| Chicken Sandwich | 820 | 6.6 | Y |

**b.** Create a database named '**15. School Canteen**' with a single table named '**Products**'. Add the fields shown in the data dictionary below and enter the above data.

| Field Name | Data Type | Description |
|---|---|---|
| ID | AutoNumber | ID number (completed automatically) |
| Product | Short Text | Name of product |
| Energy | Number | Energy (kJ per serve) |
| Fat | Number | Saturated fat (g per serve) |
| Everyday | Yes/No | Available every day |

**c.** What results do you think the SQL query on the right will produce? Create the query, naming it '**Everyday**'.

***Note:*** *When writing criteria involving a Yes/No field, you may use either Yes/No, True/False or 1/0. You cannot use the letters 'Y' and 'N'. Whichever you choose, it should be in quotes.*

**Everyday**

**SELECT** Product
**FROM** Products
**WHERE** Everyday = Yes
**ORDER BY** Product

**d.** Write an SQL query to produce the datasheet below. It should only display the products with less than 10g of fat per serve. The results should be ordered by *Fat DESC*. Name the query '**Low Fat**'.

| Product | Fat | Energy |
|---|---|---|
| Pizza | 9.8 | 1120 |
| Pasta | 6.9 | 614 |
| Chicken Sandwich | 6.6 | 820 |
| Noodle Salad | 4.2 | 711 |

***Note:*** *Remember to select the three fields in order.*

Write down or type your simplified SQL statement.

**Low Fat**

These sports themed tasks will help you practice a range of database skills. We have assumed you are able to create tables, forms, queries and reports.

**Skills Check**

- Validation ☑
- Sorting / Filtering ☑
- Visual Queries ☑
- SQL ☐
- Reports ☑
- Exporting ☑

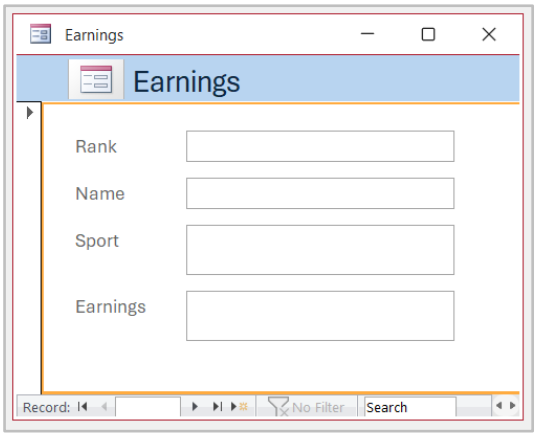
## Task 1 – Top Earners

**a.** Create a database called '**16. Sports Top Earners**'. Use *Design View* to create a table using the data dictionary below.

| Field Name | Data Type | |
|---|---|---|
| ID | AutoNumber | Completed automatically |
| Rank | Number | Ranking starting with 1 |
| Name | Short Text | Name of competitor |
| Sport | Short Text | Sport played |
| Earnings | Number | Total earned for the year US$M |

**b.** Close and save the table as '**Earnings**'. Do not enter any data at this stage.

**c.** Create a form to enter the data ('**Create / Forms / Form**').




**d.** Delete the ID field from the form; these numbers will be completed automatically.

***Note:*** *It's not a good idea to use the AutoNumber field for the ranking, even though they might both end up with identical numbers. If you make a mistake and have to delete data, then some ID numbers (and therefore ranking positions) will be missing.*

**e.** Reduce the width of the boxes.

**f.** Click on the *View* button to switch to *Form View* and enter the data below. It shows the top earners in the world of sport during the year 2024.

**g.** Use the *Tab* key to move to the next field or 'Shift + Tab' to move to the previous one.

| Rank | Name | Sport | Total Earnings (US$M) |
|---|---|---|---|
| 1 | Cristiano Ronaldo | Soccer | 260 |
| 2 | Jon Rahm | Golf | 217 |
| 3 | Lionel Messi | Soccer | 135 |
| 4 | LeBron James | Basketball | 128 |
| 5 | Giannis Antetokounmpo | Basketball | 111 |
| 6 | Kylian Mbappé | Soccer | 110 |