Advanced Databases

Microsoft Access 2013

Important Notice

All candidates who follow an ICDL/ECDL course must have an official ICDL/ECDL Registration No. (which is proof of your Profile Number with ICDL/ECDL and will track all tests taken). Without such a Registration No., no tests can be taken and the candidate will not be able to obtain an International Computer Driving Licence, nor any other form of certificate or recognition for the course.

Registration numbers are obtainable from ICDL/ECDL Training and Testing Centres or directly from Specto®.

SPECTO
مهارات لقيادة الحاسوب

Tel: +962 6 5932028
Fax: +962 6 5932029
W: www.specto.co

This Manual was compiled by:

Marianne Lubbe

Published by:

Let's Do It!

PO Box 589, Melkbosstrand, 7437

Phone: +27 (0) 21 553 5455

E-Mail: leonard@letsdoit.co.za

First Edition 2016

Published in the Republic of South Africa, Cape Town

ISBN: 978-1-928280-16-3

Disclaimer

"European Computer Driving Licence" and ECDL /ICDL are registered trademarks of the European Computer Driving Licence Foundation Limited in Ireland and other countries. Let's Do It! is an independent entity from the European Computer Driving Licence Foundation Limited, and not affiliated with the European Computer Driving License Foundation Limited in any manner. "Let's Do It! ICDL: Advanced Databases" may be used in assisting students to prepare for the European Computer Driving Licence Examination. Neither the European Computer Driving Licence Foundation Limited nor Let's Do It! warrants that the use of this "Let's Do It! ICDL: Advanced Databases" will ensure passing the relevant Examination. Use of the ECDL-F approved Courseware Logo on this product signifies that it has been independently reviewed and approved in complying with the necessary standards.

Acceptable coverage of all courseware content related to ECDL Syllabus Version 2.0. Any and all assessment tests and/or performance based exercises contained in this "Let's Do It! ICDL: Advanced Databases" relates solely to this "Let's Do It! ICDL: Advanced Databases" and do not constitute, or imply, certification by the European Driving Licence Foundation in respect of any ECDL Examinations. For details on setting ECDL/ICDL Examinations in your country please contact the local ECDL Licensee or visit the European Computer Driving Licence Foundation Limited web site at http://www.ecdl.com.

References to the European Computer Driving Licence (ECDL) include the International Computer Driving Licence (ICDL). ECDL Foundation Syllabus Version 2.0 is published as the official syllabus for use within the European Computer Driving Licence (ECDL) and International Computer Driving Licence (ICDL) certification programme.

Let's Do It! and the authors take care to ensure the accuracy and quality of this publication and accompanying materials, but we cannot guarantee their accuracy. All materials are provided without any warranty whatsoever, including but not limited to the implied warranties merchantability or fitness for a particular purpose. The related people, their positions, names addresses and other details, companies, organisations and products used for instructional purposes in this manual and accompanying material are fictitious. No associations with any real people, company, organisation or product are intended nor should any be inferred.

How to use this Manual

Using this manual you will encounter the following features:

Let's Remember!

Take note of the informative reminders.

Let's Do It!

Practical Exercises.

This is a practical guide for anyone using *Microsoft® Office 2013* software.

Copy and **paste** the complete **Adv Databases** folder from Specto website (www.specto.co\data) to the hard drive (C-Drive) of your computer. Open the templates from this folder. A separate folder can be created to save completed exercises if necessary. Additional templates are provided for demonstration and practice purposes. Where a template is available for demonstration and practice purposes, it will be mentioned at the beginning of a section.

Model answers for all exercises are provided in the **Model Answers** folder.

This manual was written using *Windows® 10*. If a different operating system is used, some dialog boxes may look different, but the content is the same.

A screen resolution of *1366 x 768* was used in compiling this manual. Working in a different screen resolution, or with an application window which is not maximized, will change the look of the *Office 2013 Ribbon*. The *Ribbon* appearance is dynamic, as it will change to fit the space available. For example, the full *Ribbon* may show a group containing several options, but if space is restricted it may show a single button that you need to click to see the same options.

Table of Contents

Module Goals

Advanced Database requires the candidate to understand key database concepts and use a relational database application to create an advanced database structure and outputs.

The candidate shall be able to:

- Understand key concepts of database development and usage.
- Create a relational database using advanced table creation features and complex relationships between tables.
- Design and use queries to create a table, update, delete and append data. Refine queries using wildcards, parameters and calculations.
- Use controls and subforms to enhance forms and improve functionality.
- Create report controls to perform calculations. Create subreports and enhance report presentation.
- Enhance productivity by working with macros and use linking and importing features to integrate data.

Database Concepts

The following outcomes will be covered in this category:

- Know that different types of database models exist
- Know the life cycle stages of a database
- Recognize common business applications of databases
- Know the term SQL and understand its main use in querying database content

Database Development and Use

Different Types of Database Models

A database model refers to the way a database is structured and used. The database model determines in which manner data can be stored, organised and manipulated in a database system. Most database systems are built around one particular data model. There are various database models, e.g.

- Flat model
- Hierarchical model
- Network model
- Relational model
- Dimensional model
- Object-oriented model

Let's have a look at the following database models:

Hierarchical Model

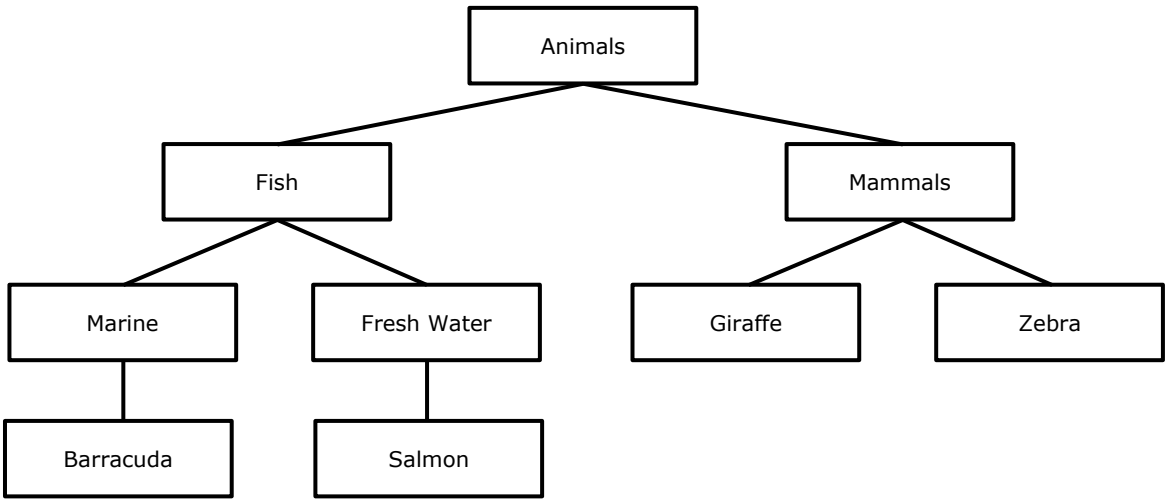
In a hierarchical model, data is organised into a tree-like structure. Think of an upside down tree of data. In this tree, a single table acts as the "root" of the database from which other tables "branch" out. There is a hierarchy of parent and child segments.

Relationships in such a system are thought of in terms of children and parents where a child may only have one parent, but a parent can have multiple children, in other words, the parent-child relationship is one-to-many. Parents and children are tied together by links called **pointers**. A parent will have a list of pointers to each of their children.

In a hierarchical model, all the instances of a specific record are collected as a record type. These record types are equivalent to tables in a relational model with the individual records being the equivalent of rows.

The **advantages** of a hierarchical model are that they are fast and conceptually simple. A **disadvantage** is that a record cannot be added to a child table until it has already been incorporated into a parent table. The hierarchical model also creates repetition of data within the database. Redundancy will occur as hierarchical databases handle one-to-many relationships well, but not many-to-many relationships. To find data in a hierarchical model, the hierarchical structure of the database should be known to the user.

Example of a hierarchical model:



Uses:

Today, the hierarchical model is rarely used in modern databases.

- Drug databases
- XML documents

Relational Model

The relational model is the most popular model as it provides useful tools for database administration. The relational model was developed by E.F. Codd in 1970. As he was a mathematician, he built the model on mathematical concepts.

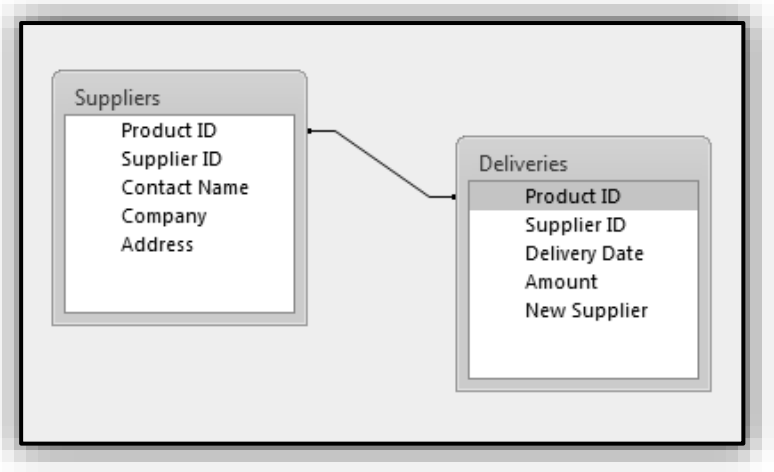
Three key terms are used in relational database models: **relations**, **attributes** and **domains**. A relation is a table with columns (fields) and rows (records). The named columns of the table are called attributes (field names) and the domain is the set of values the attributes are allowed to take.

Each table (relation) is made up of records (horizontal rows, also known as tuples) and fields (vertical columns, also known as attributes). Each row is unique and each column has a unique name.

A relational database contains **multiple tables**. Any value occurring in two different records (belonging to the same table or to different tables) implies a relationship between those two records. Primary keys are commonly used to join or combine data from two or more tables.

Unlike a hierarchical database model, a user does not have to have an understanding of how the data is structured within the relational database in order to retrieve, insert, update or delete records from the database. All the user needs to know is the name of the table in order to use it.

Example of a relational model:



Uses:

- Patient databases
- Student databases

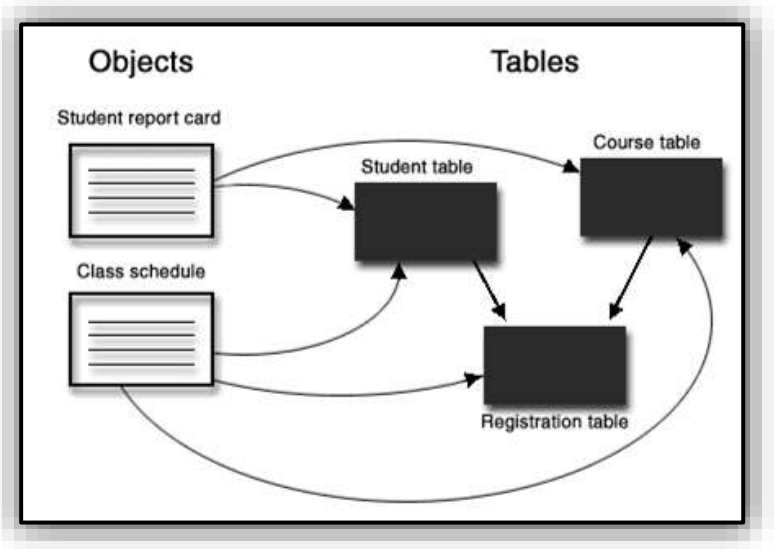
Object-Oriented Model

An object-oriented database model is a database model in which information is represented in the form of objects as used in object-oriented programming. Object databases are a niche field within the broader database management system market. Object databases have been used for quite a few years, but have made little impact on mainstream commercial data processing, though there is some usage in specialised areas.

Object-oriented databases store objects rather than data, such as integers, strings or real numbers. Objects are used in object oriented languages, such as *Smalltalk* and *Java*. Objects basically consist of the following: **attributes** (attributes are data which defines the characteristics of an object) and **methods** (methods define the behaviour of an object and are what was formally called procedures or functions).

The one-to-one mapping of object programming language objects to database objects has two **benefits** over other storage approaches: it provides higher performance management of objects and it enables better management of the complex interrelationships between objects. With traditional databases, data manipulated by the application is temporary and data in the database is stored on a permanent storage device. In object-oriented databases, the application can manipulate both temporary and persisted data.

Example of an object-oriented model:



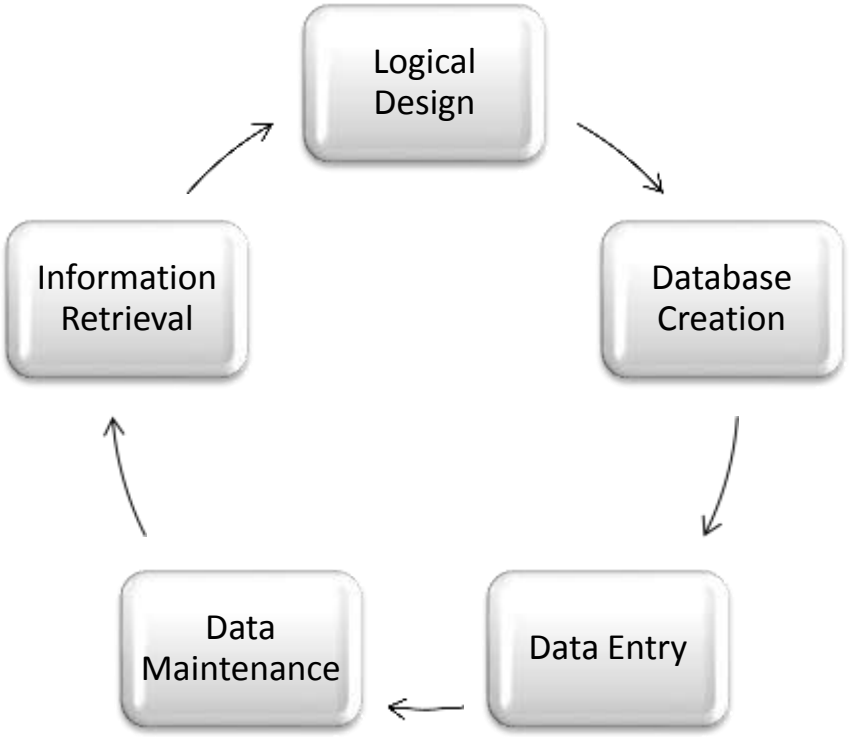
Uses:

- Multimedia databases
- Hypertext databases

The Life Cycle Stages of a Database

The life cycle of a database is the cycle of development and changes that a database goes through during the course of its life. The cycle typically consists of several stages:

- Logical Design
- Database Creation
- Data Entry
- Data Maintenance
- Information Retrieval



Logical Design

The logical design stage is the first and most important stage in the database life cycle. A database's usefulness is directly proportional to the efficiency of its design. When designing a database, it is important to keep data retrieval, storage and modification efficiency in mind. A requirement analysis has to be done in order to collect and analyse the requirements and expectations of the users. A conceptual design is then created based on the results of the requirement analysis.

In this stage, it will also be decided which database model is perfectly suited for the organisation's requirements. A designer must consider end-user views, define entities, attributes, relationships and identify processes. It is also necessary for the designer to consider future expansion of the database and to make sure the design is easily extendible. The conceptual design is then translated into the logical design.

Database Creation

After the design stage, the database system is created. The purpose of this stage is to construct and install the information system according to the plan and design. Creating a database includes creating database definitions (such as tables, indexes, etc.), developing applications, testing the system and developing operational procedures and documentation.

Database creation is vital in the database life cycle and has great significance as a bad design can result in deprived performance.

Data Entry

Once the database has been created, the data must be loaded into the database. The data required must be converted to the new database if the loaded data is stored in a different system or in a different format. This stage of the life cycle also requires that the database performance is evaluated, security standards set up, and backup and recovery procedures put in place as well as data integrity enforcement.

Testing is also performed at this stage. Testing the database includes fine-tuning the database for performance, integrity, access and security constraints. It is recommended to create a test database separate of the production database. The test database will be useful for testing any changes and new and modified applications before applying the changes to the production database. Careful testing before handover will minimize the expense of later modifications to the database.

Data Maintenance

Data maintenance is important and it is an on-going stage in the database life cycle. Factors, such as new business needs, new information requirements and acquisition of new data will make it essential to formulate on-going changes and improvements to the existing database. Tasks in this stage also include database backup and recovery, performance tuning, design modifications, audits, usage monitoring and hardware maintenance.

Information Retrieval

Once the data is loaded into the database and it is fully tested, the database is then released into production. The database is accessed by the users and application programmes. The queries and reports created in the database can now be used to retrieve specific information needed by the user. Information retrieval is important for users or business management to make decisions based on the given results.

Common Business Applications of Databases

A database management system (DBMS) is a collection of programmes that enables you to store, modify and extract information from a database. There are many different types of DBMSs, ranging from small systems that run on personal computers to huge systems that run on mainframes. Here are some business applications of databases:

Dynamic Websites

Classical hypertext navigation occurs among "static" documents, and for a web user, this experience is reproduced using static web pages, meaning that a page retrieved by different users at different times is always the same and in the same form. A dynamic web page is a kind of web page that has been prepared with fresh information for each individual viewing.

This means that the web page can provide a live user experience. Content on a web page can change in response to different contexts or conditions. In dynamic websites, page content and page layout are created separately. The content is retrieved from a database and is placed on a web page only when needed or asked. This allows for quicker page loading and it allows just about anyone with limited web design experience to update their own website via an administrative tool.

Customer Relationship Management Systems

Customer Relationship Management (CRM) is a widely implemented strategy for managing a company's interactions with customers, clients and sales prospects. It involves using a database management system to organise, automate and synchronise business processes. The overall goals are to find, attract and win new

clients, nurture and retain those the company already has, entice former clients back and reduce the cost of marketing and client service.

One of the most important functions of a CRM system is that it stores information about your customers. This is something that is essential, not only for being able to contact them easily, but also for documenting their ordering history. The better you know your customers, the more effectively you can retain their business. A good CRM system allows you to easily retrieve relevant information about your customers. This can save you a lot of time and money.

Enterprise Resource Planning Systems

Enterprise Resource Planning (ERP) is an industry term for the broad set of activities that helps a business manage all facets of the business. The information made available through an ERP system provides visibility for key performance indicators required for meeting corporate objectives. ERP systems can be used to manage product planning, parts purchasing, and inventories, interacting with suppliers, providing customer service and tracking orders. ERP can also include business activities, such as finance (e.g. accounts received and accounts payable) and human resources (e.g. payroll) aspects of a business. Typically, an ERP system uses or is integrated with a relational database system.

Website Content Management Systems

A Web Content Management System (WCMS) is a software system that provides website authoring, collaboration, and administration tools designed to allow users with little knowledge of web programming languages or mark-up languages to create and manage website content with relative ease. A WCMS provides the foundation for collaboration, offering users the ability to manage documents and output for multiple author editing and participation.

Most systems use a database to store page content, metadata, and other information assets that might be needed by the system. Content is frequently, but not universally, stored as XML to facilitate, reuse, and enable flexible presentation options. Administration is typically done through browser-based interfaces, but some systems require the use of a fat client.

A WCMS allows non-technical users to make changes to a website with little training. A WCMS typically requires a systems administrator and /or a web developer to set up and add features, but it is primarily a website maintenance tool for non-technical staff.

Structured Query Language (SQL)

To communicate with the database system itself, a language is needed. *Structured Query Language* (*SQL*) is a database computer language designed for managing data in relational database management systems (RDBMS), and originally based upon relational algebra and calculus.

Donald D. Chamberlin and Raymond F. Boyce developed SQL at IBM in the early 1970s. This version, initially called *SEQUEL* (*Structured English Query Language*), was designed to manipulate and retrieve data stored in IBM's original relational database management system. The acronym *SEQUEL* later changed to *SQL*.

Its scope includes data insert, query, update and delete, schema creation and modification, and data access control. *SQL* was designed with a specific purpose in mind: to **query** data contained in a relational database. *SQL* is about data and results; each *SQL* statement returns a result, whether that result is a query, an update to a record or the creation of a database table.

For example, the *SQL* command for creating a database is:

CREATE DATABASE database_name;

The *SQL* language is sub-divided into several language elements, including:

- **Clauses**: fundamental components of statements and queries.
- **Expressions**: used to produce either scalar values or tables consisting of columns and rows of data.
- **Predicates**: specifies conditions that can be evaluated to *SQL* three-valued logic (3VL) or Boolean (true/false/unknown) truth values and which are used to limit the effects of statements and queries, or to change programme flow.
- **Queries**: retrieve data based on specific criteria. *This is the most important element of SQL*.
- **Statements**: may have a persistent effect on schemata and data, or which may control transactions, programme flow, connections, sessions or diagnostics.
- **Insignificant whitespace** is generally ignored in SQL statements and queries, making it easier to format *SQL* code for readability.

The most common operation in *SQL* is the query, which is performed with the declarative *SELECT* statement. *SELECT* retrieves data from one or more tables, or expressions.

An example of a SELECT statement:

The following is an example of a *SELECT* query that returns a list of expensive books. The query retrieves all rows from the *Book* table in which the price column contains a value greater than 250.00. The result is sorted in ascending order by *title*. The asterisk (*) in the select list indicates that all columns of the *Book* table should be included in the result set.

```
SELECT *
    FROM Book
    WHERE price > 250.00
    ORDER BY title;
```

Let's Do It! ⇨ 1

1. Name three types of database models.
2. What are primary keys used for in a relational database?
3. Name one use of an object-oriented database model.
4. Name the five stages of the database life cycle.
5. What happens during the database creation stage in the database life cycle?
6. Explain how a database can be used in customer relationship management.
7. What does SQL stand for?
8. In SQL, which language element is used to retrieve data based on specific criteria?
9. What is a website content management system used for?
10. In what stage of the database life cycle is the database tested?

Tables and Relationships

The following outcomes will be covered in this category:

- Create, modify, delete a lookup in a field/column
- Create, modify, delete an input mask in a field/column
- Set data entry for a field/column: required, not required
- Create, modify, delete a one-to-one, one to many relationship between tables
- Create, modify a many-to-many relationship using a junction table

- Apply referential integrity between tables
- Apply automatic update of related fields
- Apply automatic deletion of related records
- Apply, modify an inner join, outer join
- Create, modify a subtract join
- Apply a self-join

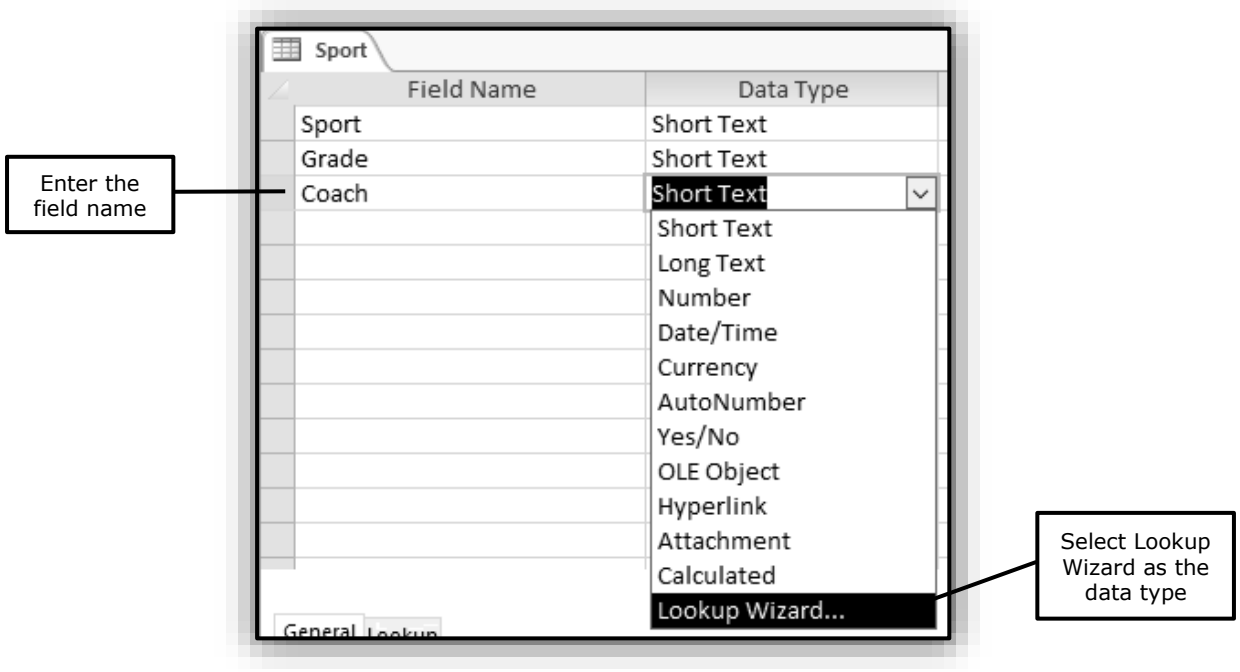
Fields / Columns

Create, Modify and Delete a Lookup in a Field / Column

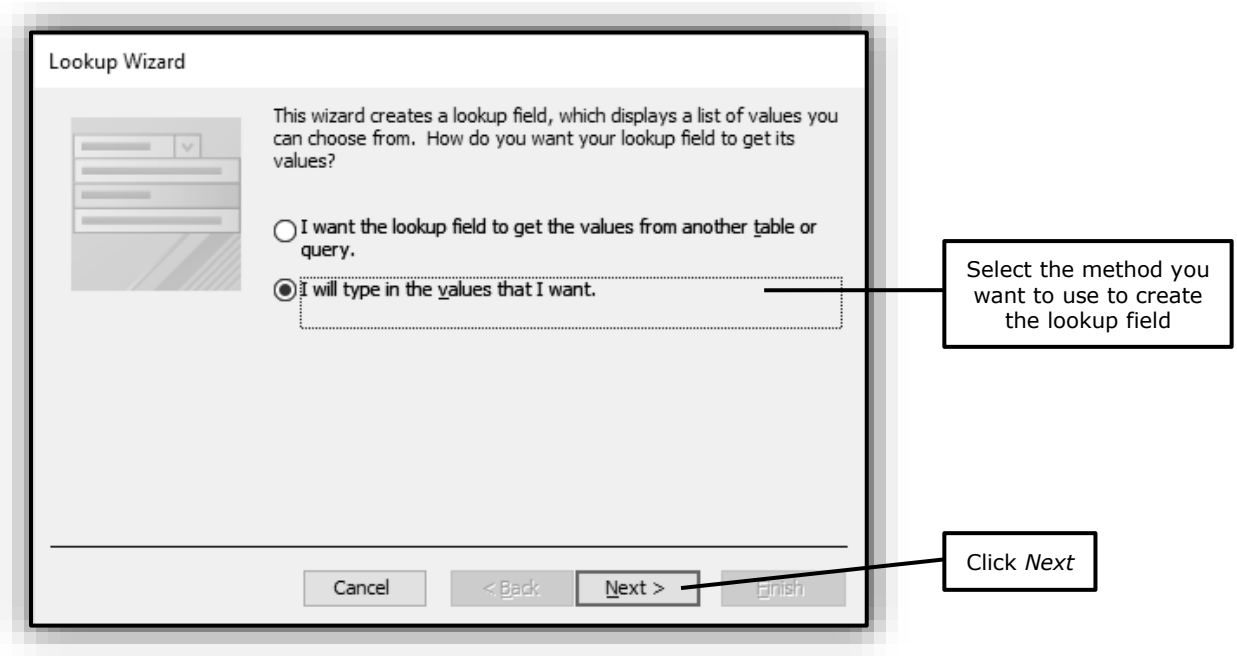
For information that is repeated in multiple tables, a lookup field can be created. Lookup fields prevent keying errors when data is entered into a table and it also saves time in that each entry does not have to be typed, but simply selected. In other words, it helps improve the efficiency of the data entry process for your database. You can create a lookup field in either Design or Datasheet view.

Create

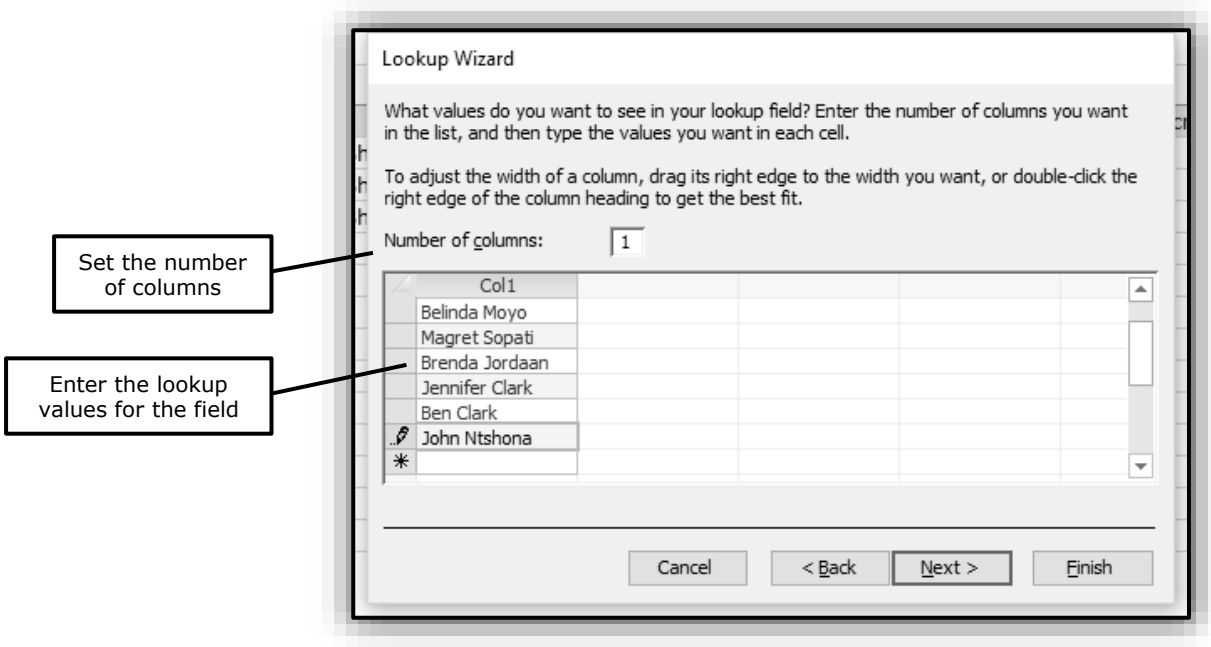
1. Open the **Ubora School.accdb** database.
2. Open the **Sport** table where you will create a lookup field in *Design* view. You can also follow these steps while creating a new table.
3. Locate the field with the name **Coach** in the *Field Name* box.
4. Change the data type by clicking the drop-down list and selecting *Lookup Wizard*.



5. The *Lookup Wizard* dialog box will open.
6. Select the *I will type in the values that I want* option button.
7. Click *Next*.



8. Ensure the *Number of columns* is set to 1.
9. Enter the following entries in the column (Press *Tab* to move from one entry to the next):
   Janice Griffith
   Marike Strydom
   Belinda Moyo
   Magret Sopati
   Brenda Jordaan
   Jennifer Clark
   Ben Clark
   John Ntshona



10. Click *Next*.
11. If you want to make sure that only entries from the list can be used, select the *Limit To List* check box.
12. The label will be the same as the field name. The name can be changed if desired.