N4 Database knowledge re-cap

## Lesson 1

# What is a database?

A database is an organised collection of information or data. Putting information into a computer database gives you far more flexibility in organising, displaying and printing that information. Information in a database is organised into **Fields, Records and Files**.

**Fields**

# 

**File**

# Advantages of computerised databases

A computerised database doesn’t take up lots of room and can be easily edited without having to re-print things. Computerised databases are much quicker to search for a certain piece of information than looking through paper records. It is also easier to sort a computerised database e.g. by a person’s surname, than a paper based system.

# Database Structure

## Database Entities

A database contains one or more tables. Each one of these tables holds information about particular people, objects or things – called entities.

Some examples of database tables might be:

* a customer table
* an employee
* a movies table

**Entities**

* a smartphone table
* a car table

**The definition of an entity is a person, place, thing or concept about which data can be collected.** In a database there is a table for each entity.

## Attributes

Each entity will be made up from a number of **attributes** which represent that entity. For example, in the customer entity:

### Customer Entity

* First name
* Surname

**Attributes**

* Address
* Phone number
* Date of birth

Each of the bullet points above is known as an attribute. These attributes are what the customer entity is made up from.

Another example could be a movies entity:

### Movies Entity

* Movie title
* Release date

**Attributes**

* Director
* How much made at box office
* Movie length

**An attribute describes the facts, details or characteristics of an entity**.

In a database there is a field for each attribute.

## Database Attribute Types & Sizes

|  |  |
| --- | --- |
| **Text** | * Text fields will store characters or words. * Can be a combination of text and numbers e.g. a postcode or an address with a street number (PH33 7BP or 14 High Street) * A **telephone number** would be a **text field**. This is because it starts with a 0. |
| **Number** | * Number field types can store whole numbers (integers) such as 8,99,236 or numbers with decimal places (real numbers) such as 76.33, 2.11, 4.5. |
| **Date** | * This field type will store dates. These can be stored in different formats. You will decide when you design the table what format you would like to choose. |
| **Time** | * A time field stores the time of day, normally in 24 hour format e.g. 15:30 |
| **Boolean** | * A Boolean field is either true/ false or yes/ no. |

## Attribute Size

When you create a database you want to put limits on each attribute to ensure it does not take up too much **storage space**. For example, a person’s forename could be limited to 20 characters in size.

# Field validation

Validation is done to ensure data entered is allowable & sensible. You need to know about the following different types of validation.

|  |  |
| --- | --- |
| Range Check | Ensures the data entered in the field is between a lower and upper limit.  e.g.  **Cost >0 AND <100**  **Time >1 min AND<5 mins**  Date before 1 Jan 2001: **<#01/01/2001#**  Format for a date in MS Access  Date between 1 Jan ‘01 AND 31 Dec ‘01:    **<#31/12/2001# And >#01/01/2001#**  Format for a time in MS Access  Time later than 9am **>#09:00:00#**  Time between 9am AND 7pm **<#19:00:00# And >#09:00:00#** |
| Restricted Choice | Gives the user a list of options to choose from.  Prevents user typing errors. |
| Length Check | Ensures the data entered in the field has a restricted number of characters;  e.g. PIN = 4 chars  Password >= 6 chars  Comments < 200 chars  Or a combination of the above |
| Presence Check | Ensures the field is not left blank |

# Questions

1. State three advantages of using a computerised database compared to a paper based database.
2. In your own words explain the terms
   * Table
   * Field
   * File
3. Write down suitable attribute types for the following:
   * Name
   * Age
   * Date of Birth
   * Car registration number
   * Qualified driver
4. What type of data type should be used to store a telephone number? Why?
5. Look at the following information from a school database:

**Forename:** John

**Surname:** Smith

**House Number:** 5

**Address:** Riverside Road

**Town:** Fort William

**Postcode:** PH33 7QR

**Phone No:** 01399 772221

**Date of Birth:** 26/09/2003

**Facebook:** www.facebook.com/johnsmith

Gets bus to school: Yes

Copy the following table into your jotter and complete it using the information above. Add sufficient rows for all the attributes, fill in attribute names and then state what data type each attribute is.

|  |  |
| --- | --- |
| Attribute Name | Data Type |
|  |  |
|  |  |

1. Explain why a restricted choice validation might be used on a database field.
2. In the Drivers Licence database, all licence holders must be at least 17 years of age. Write down the field validation used (use todays date and month).
3. Staff in Fresco can work flexi hours, but must start between 10 am and 11.30am. Write down the field validation for the Starting Time field in Fresco’s employee database.

Extension Questions

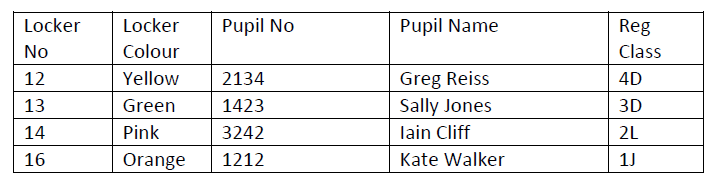
1. A school stores the following information for each member of staff:

First name, Middle name, Surname, Address (street, postcode), Date of Birth, Contact phone number, Numbers of days absent.

In your jotter, produce a table showing Attributes and Data Types for the Pupil entity.

|  |  |
| --- | --- |
| Attribute Name | Data Type |
|  |  |
|  |  |

1. The pupils at Slains Academy use lockers in which to keep their books, P.E. kit etc. The janitor keeps the following record:



* + 1. Write down a suitable entity name for this database.
    2. List the attributes, their data types and suitable attribute sizes for this entity.

1. A cycling club holds the following information about members:

* First name
* Surname
* Date of Birth
* Address
* Mobile number
* Membership Fees Paid
* Personal Best Time

Copy and complete the following table:

|  |  |
| --- | --- |
| Entity: | |
| Attribute Name | Data Type Attribute Size |
|  |  |
|  |  |

1. Poundstore has a database of all its branches. The following details are stored in it.

**Branch number, Branch name, Telephone number, Rent, Opening hours, Date opened.**

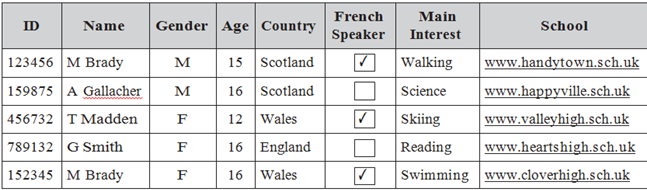
Each branch must have a Branch number, the Rent must be less than £250, the opening hours must be between 9am and 7 pm and the date opened must be after

12 December 2015.

Using this information, copy out and complete this table, adding extra rows as needed:

|  |  |  |
| --- | --- | --- |
| Entity: | | |
| Attribute Name | Data Type | Validation |
|  |  |  |

1. French schools use the following database to help pupils find pen pals in Britain.



1. Write down a suitable entity name for this table.
2. State the most appropriate data type for the French Speaker field.
3. State the most appropriate data type for the Gender field.
4. State what type of data validation should be used on the Country field.

N5 Course

## Lesson 2

The N5 Information Systems Design and Development (ISDD) unit covers the following stages:

* Database Structure
* Analysis
* Design
* Implementation
* Testing
* Evaluation

# Database Structure

## Flat File Databases

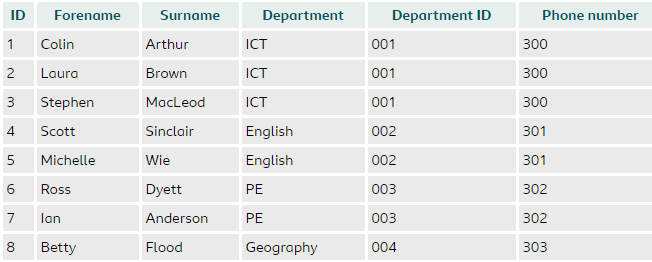
Flat file databases consist of one table. A flat file database is useful when storing a small number of records.

Flat-file databases have several problems:

* Data can be duplicated in a row of a table.
* Duplicating data increases the chance of human error when typing in the same data several times. It can then be hard to be sure which entry is correct.

Example:

The table below holds data on *teachers* in a school. It also contains information about the *department* they work in.



In this table the **Department** and **Department ID** represent the same thing – they **duplicate** the same information. Each teacher record only needs **one** of these fields.

## Relational Databases

To avoid duplicating data in a database table we create two tables, instead of one, link these tables and split the duplicated data between the two tables.

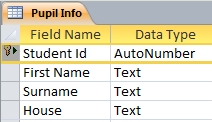
**A database which contains two or more linked tables is called a relational database**.

Relational databasesuse **Primary** and **Foreign** Keys to link the tables.

### Primary keys

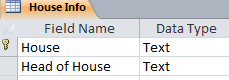
A primary key is a field used to uniquely identify every record in a database. Sometime primary keys are obvious, for example a car license plate could be used to uniquely identify cars. Often, primary keys are less obvious, and it is necessary to create a unique value – often an ID number.

In this school database, pupils are identified by their unique Student ID number – it’s the primary key to the Pupil Info table:



Student Id is the primary key in the Student Info table

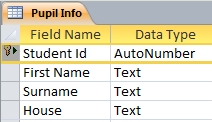
Information about each House is stored in the House Info table. The primary key for this is the House name:

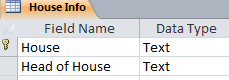


House is the primary key in the House Info table

### Foreign Keys

A foreign key in a table is a primary key (from another table) that links the two tables together. In the school database, the House key is the Primary key in the House Info table and the Foreign key in the Pupil Info table:

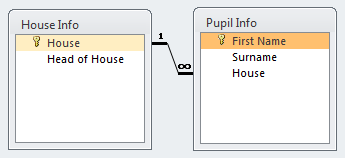




House is the Foreign key in the Pupil Info table

House is the Primary key in the House Info table

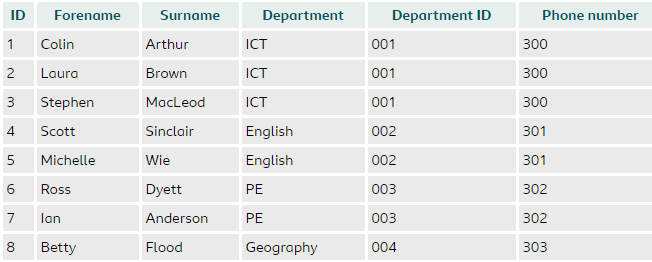
Because the two tables are linked by a common field, the foreign key, we say there is **a relationship** between the tables. This relationship is shown like this:



Foreign key

Primary key

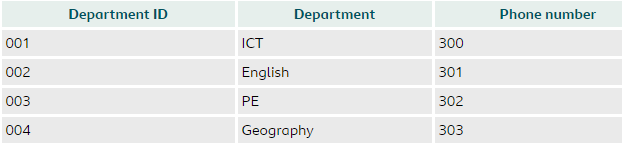
## Advantages of Relational Databases

Relational databases are used to **avoid duplicating data in a record** and **reduce the chance of human error** when entering information in a database. Look again at the teachers’ database from before:

It’s possible to split the data into two different tables and to link them together using a field that is relevant to both tables.

The first table could store details about each department. The Department ID is the **primary key.**

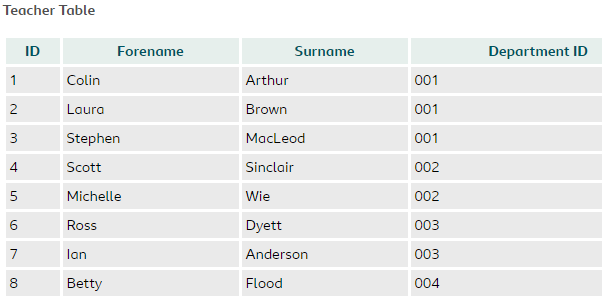
Department Table



Department ID is the **primary key** in the Department table

The second table could store details on **each member of staff**. The teacher’s ID is the **primary key.** The Department ID is the Foreign key and links the two tables together.

TeacherTable



Department ID is the **foreign key** in the Teacher table

ID is the **primary key** in the Teacher table

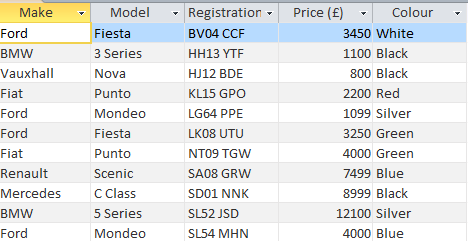
Using 2 linked tables reduces the likelihood of errors. By using linked tables in the Teachers database example:

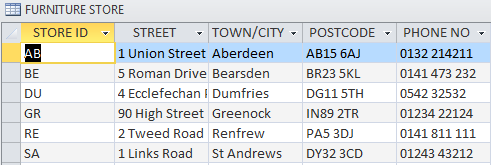
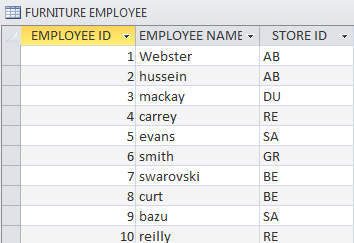
* A new department can be added without having to add staff information alongside it.
* If a member of staff were deleted, such as Betty Flood, you would no longer lose the information held about her department.
* If the ICT phone number had to change, it would only have to change once in the department table.

Questions

1. What is a relational Database
2. What is a primary key?
3. What is a foreign key?
4. Clyde Vets stores all its client information in a flat file database. Here is an extract:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| First Name | Surname | Address | Town | Pet Name | Pet Type |
| Hardeep | Singh | 65 Iona Way | Greenock | Foggy | Cat |
| Hardeep | Singh | 65 Iona Way | Greenock | Gladys | Gerbil |
| Hardeep | Singh | 65 Iona Way | Greenock | Speedy | Tortoise |
| Sally | Chan | 142 Main Street | Greenock | Misty | Cat |
| Elaine | Bryce | 29 Clyde Drive | Greenock | Slinky | Tortoise |
| Cameron | Gray | 17 Shuttle Street | Greenock | Rover | Dog |

1. Identify the duplicate information in the extract of the database shown.
2. What problems could be encountered by using a flat file database?
3. Here is an extract of some the data held in Arnold Brown’s database of all cars in their local showroom:
   1. What field would you use as the primary key for this table?
   2. Explain your choice.
4. Here is an extract of 2 tables from the Sofa Workhouse database:



1. Identify the Primary key in each table.
2. Identify the Foreign key used.
3. Explain the relationship between the tables (how they are joined).

Extension Questions

1. Here is an extract of 2 tables from the Clyde Cycling Club's membership database:
2. Identify the primary and foreign keys used in these tables.
3. Explain the relationship between the tables.



1. Rex R Us stores details about used cars for sale in each of its branches in a flat file database. Here is an extract of the Cars table:



1. Redesign this as a relational database, with two tables, showing the fields in each table.
2. Identify the Primary key in each table.
3. Identify the Foreign key.

Analysis

## Lesson 3

During the analysis stage of database development, the following requirements should be identified:

### End-user requirements:

The **end users** are the people who are going to be using the database.

Their **requirements** are the tasks they expect to be able to do using the database.

For example, if creating a database for a doctor's surgery, it would be necessary to speak to the employees who would make use of the database.

Developers could ask that each user group lists the tasks that they wish to be able to complete. In this example, two different end user groups exist, reception staff and medical staff.

Reception staff report that they must be able to:

* enrol patients
* search for basic patient data
* output appointment letters
* output a summary of patient details

Medical staff report that they must be able to:

* add medical notes
* generate onward referrals
* access confidential medical records
* output prescriptions

### Functional requirements:

These are the **tasks** that the system has to perform – usually in the form of searching and sorting (queries).

They specify the **information** that the system has to contain to be able to carry out its functions.

These requirements will help:

* clarify the design of the database - tables/entities & fields/attributes
* identify the features to be implemented on the database
* evaluate whether the system is fit for purpose after development is complete

In the example of the doctor's surgery, the functional requirements could include the following:

A table should store basic patient details including fields for:

* patient ID
* first name
* surname
* date of birth
* address
* contact number
* next appointment time
* next appointment date

The basic patient data table should be linked to a medical records table.

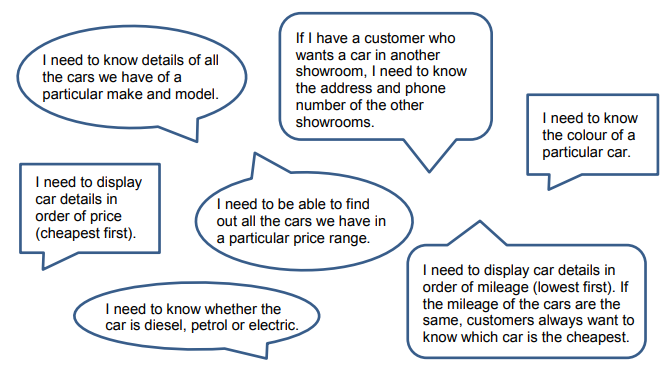
The medical records table should include the following:

* Patient ID
* Medical record ID
* Diagnosed illness
* Medication
* Blood type

Simple and complex queries should exist to allow for searching and sorting on both tables.

Example

A used-car dealership has six showrooms in different locations across Scotland. It wants to create a relational database to store details of cars owned by the company and details of each of their showrooms. The database will allow sales staff to view details of specific cars and to see which showroom the cars are located in. The developers have asked some of the sales staff what they would like to see in the database. Here are a few of the comments they made.



### End-user requirements

1. Sales staff should be able to display details of cars, by performing a number of different searches using:

* car make and model
* car colour
* range of car prices
* type of fuel used

1. Search results should display:

* car make
* car model
* car price
* car fuel
* car mileage
* branch address
* branch telephone number

1. Users should be able to sort the search results in ascending order of mileage and ascending order of price.

### Functional requirements

1. The relational database will have two tables:

* one for car details and one for branch details:
* each table will require a suitable primary key field.
* a foreign key will be used to link the two tables.

1. Additional fields will be needed for:

car make

car model

car colour

car fuel — diesel, petrol or electric

car price

car mileage

branch address

branch telephone number

1. Simple and complex queries will be used to search the database.
2. A complex sort will be used to order the query results.

Design

## The law

You must consider the **legal implications** when designing a database. For National 5, you are expected to be able to describe and give examples of the General Data Protection Regulation (GDPR). This law applies to all personal data held by businesses and organisations. It protects individuals’ personal data from being used without their permission (e.g. their e-mail address on mailing lists or their names and addresses passed on to other companies to be used in mail shots and advertisements); ensures organisations who hold person data keep it securely (e.g. password protected or encrypted), ensures organisations keep personal data accurate and up-to-date (e.g. the annual pupil data check in schools); and that organisations destroy data when they no longer need it.

The GDPR states that data must be:

* Processed lawfully, fairly and in a transparent manner in relation to individuals
* Used for the declared purpose only
* Limited to the data needed for the declared purpose
* Accurate
* Not kept for longer than necessary
* Held securely

## Entity Relationship Diagrams

An **entity-relationship** **(ER)** diagram is a graphical representation of the **entities** in a system.

It is used to show the **relationship** that exists between two or more **entities**.

### Entities

In database design, entities represent what will become tables during implementation.

The term 'entity' is used to describe the following:

* a person
* a place
* an object
* a thing

Entities are made up of many attributes.

### Attributes

Attributes are the specific characteristics of an entity. Attributes will become fields during implementation.

An entity to store information on a country may include the following attributes:

* name
* population
* continent
* language
* currency

## Relationships and Entity Relationship Diagrams

In a relational database, data is stored across several tables which are linked together using primary and foreign keys. The relationship that exists between different entities(tables) is shown in an Entity Relationship (ER) diagram.

For National 5, you must be able to describe and give examples of ER Diagrams with 2 entities, showing:

• entity name,

• attributes

• relationship (one to many) between entities

A **one-to-many relationship** exists when **one** entity can be present in **many** different instances of another entity. One-to-many relationships are the most common relationships in correctly implemented relational database management systems.

One-to-many relationships can be shown in several ways:



Example

A relational database is used by a travel agency to store details of **Scottish holiday resorts** and the **hotels** that are in each resort.

The resort and hotel details have been arranged into two entities:

* Resort
* Hotel

The attributes of each entity are:

Resort entity Hotel entity

resortID hotelRef

town hotelName

resortType phoneNumber

trainStation resortID

starRating

seasonStartDate

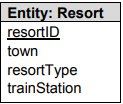
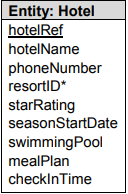
swimmingPool

mealPlan

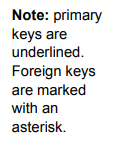
checkInTime

The entities and their attributes are usually shown in diagram form.

The diagrams below illustrate the Hotel and Resort **entities** and their **attributes**. The relationship between the two entities is also shown in the diagram:



Is the location of



This shows that **one Resort** is the locationof **many hotels.**

This is a **one-to-many relationship.**

The diagram below is an alternative way to illustrate the relationship between Hotel and Resort entities:

## 

Questions

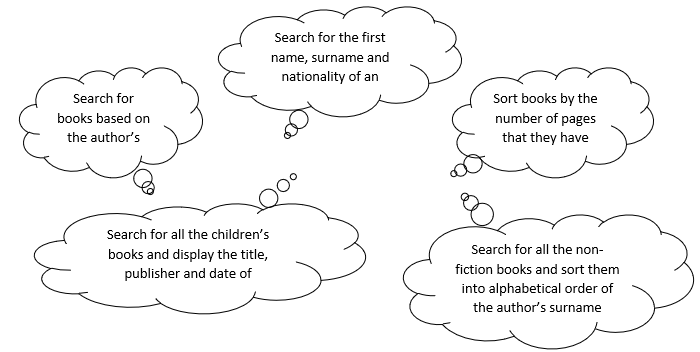
1. Curl up and Dye hair salon stores details of its customers on a database.
2. State one principle Curl up and Dye must comply with in terms of the General Data Protection Regulation.
3. Explain why compliance with this principle is important to customers.
4. For each of the following tasks identify:

The *Functional* *Requirements* (including tables, fields and keys)

The *End-User* *Requirements* (including details of queries)

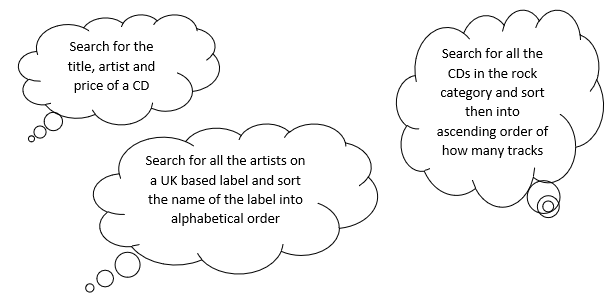
1. Clydeview Library

Clydeview Library wants a new relational database to store details of books and authors. The developers of the database have asked the librarian to list some of the tasks that she would like complete using the new database.



1. Online Music Store

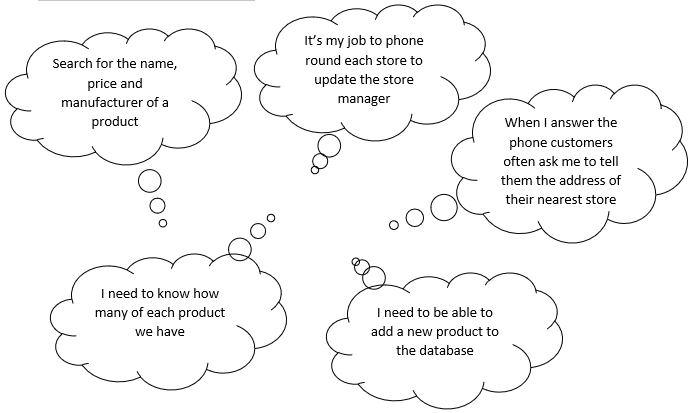
An online music store wants a new relational database to store details of CDs and Record Labels. The developers of the database have asked customers of the music store to describe some of the tasks that they would expect to carry out using the new database.



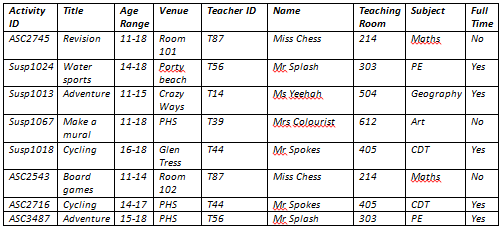
1. East Face Clothing

The East Face Clothing Company has stores across Scotland. They want to use a relational database to store details about their products and their stores. The database should allow Head Office staff to search for particular products in order to check stock levels and find out contact details for each of its stores.

The developers have asked employees at Head Office to list some of the things that they would like to be able to do with the database.



1. Lockerbie Academy has a flat file database which shows details of activities and the staff who run them:



The decision is taken to change the database into a relational database, containing 2 tables.

1. Draw the Entity Relationship (ER) diagram for the new database design.
2. Show the attributes for each entity.
3. Show the relationship between the two entities.

## Data Dictionaries

## Lesson 4

Data dictionaries are created during the design phase to define the structure of a database.

Data dictionaries contain the following **metadata** - information about the data in the database

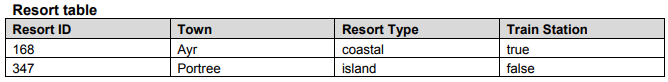
* Entity name
* Attribute name
* Attribute size (field size)
* Primary & foreign key
* Attribute type:
* text,
* number,
* date,
* time,
* Boolean
* Attribute size
* Validation
* Presence check
* Restricted choice
* Field length
* Range
* Validation rules for each attribute

For National 5, you must be able to describe and give examples of Data Dictionaries, specifying the relevant metadata from the list above.

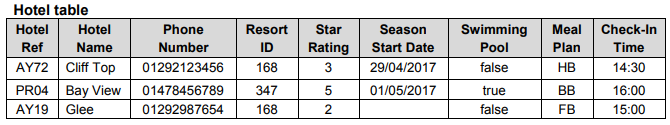
Example

Using our example of Scottish holiday resorts and hotels, the resort and hotel details are arranged in two separate entities. Each entity will be a separate table in the database.

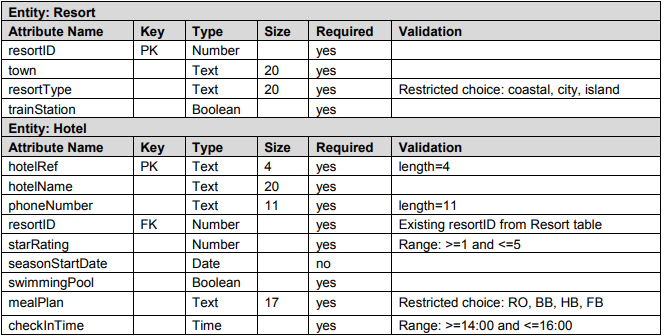
The resort entity:



The hotel entity:



Here is the completed data dictionary:



# Solutions to Queries

For National 5, you must be able to design a solution to a query which will interrogate several table and more than one field. Your query should use search criteria to find specific information and use sort order to display the information in a specific order.

Example

Here are the two tables from our Resort database:

****

1. Design a query to list the town name and the train station details of all resorts that have a train station.

|  |  |
| --- | --- |
| Field(s) | town, trainStation |
| Tables(s) | Resort  trainStation is a Boolean (true/false) |
| Search criteria | trainStation=true |
| Sort order |  |

1. Design a query to list the hotel name and phone number, together with the star rating and swimming pool details for all hotels with a swimming pool that have a rating of at least 4 stars.

|  |  |
| --- | --- |
| Field(s) | hotelName, phoneNumber, starRating, swimmingPool |
| Table(s) | Hotel |
| Search criteria | swimmingPool = true AND starRating >= 4 |

1. Design a query to list the hotel name and phone number, together with the town and train station details of any hotel in Ayr.

|  |  |
| --- | --- |
| Field(s) | hotelName, phoneNumber, town, trainStation |
| Table(s) | Hotel, Resort |
| Search criteria | town = "Ayr" |

1. Design a query to list the town name, resort type and star rating of all hotels that have a 5 star rating. These details should be listed in alphabetical order of town.

|  |  |
| --- | --- |
| Field(s) | town, resortType, starRating |
| Table(s) | Resort, Hotel |
| Search criteria | starRating = 5 |
| Sort order | town ASC |

1. Design a query to list the hotel name and its star rating, together with the town, resort type and check-in time, of all hotels that allow check in before 15:00. These details should be displayed so that the hotel with the highest rating is listed first; hotels with the same star rating should be listed in alphabetical order of town name.

|  |  |
| --- | --- |
| Field(s) | hotelName, starRating, town, resortType, checkInTime |
| Table(s) | Hotel, Resort |
| Search criteria | checkInTime < 15:00 |
| Sort order | starRating DESC, town ASC |

Implementation, Testing and Evaluation

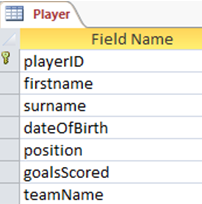
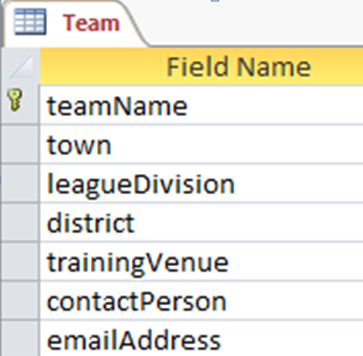
## Lesson 5

# SQL

SQL stands for Structured Query Language. SQL is a special purpose programming language for storing, manipulating and retrieving data in relational databases. Although most database systems use SQL, there can be a number of differences between different dialects or versions. However, the standard SQL commands such as SELECT, INSERT, UPDATE and DELETE are common to them all.

We will use the following relational database to exemplify the SQL commands:

The Scottish Handball League use a relational database to store details of teams and players in two separate tables called Team and Player. The structure of these tables is shown below:



A sample record stored in each table is shown below:

|  |  |
| --- | --- |
| **Team table** | |
| **teamName** | Clyde Flyers |
| **town** | Greenock |
| **leagueDivision** | 1 |
| **district** | West |
| **trainingVenue** | Burnside Sports Centre |
| **contactPerson** | Chris Black |
| **emailPerson** | [clydeflyers@handball.mail.uk](mailto:clydeflyers@handball.mail.uk) |

|  |  |
| --- | --- |
| **Player table** |  |
| **playerID** | 810JE |
| **firstname** | Jack |
| **surname** | Edwards |
| **dateOfBirth** | 03/05/1994 |
| **position** | Left back |
| **goalsScored** | 37 |
| **teamName** | Clyde Flyers |

## Searching

The **SELECT** statement is used to decide which **fields** should be displayed. The SELECT statement is followed by the fields, separated by commas.

The **FROM** clause states the names of the database **table(s)** that are needed in the query.

The **WHERE** clause states the **criteria** that must be met. This clause is followed by the field

name, an operator (<, >, =) and the information inside inverted commas if it is text.

**SELECT** *fieldName1, fieldName2, fieldName3*

**FROM** *tableName*

**WHERE** *fieldName = data;*

**Logical operators** (AND, OR) can be used to create complex criteria.

Examples

1. To search the database to display the town, contact person, e-mail address and district for all teams in the west district, you would write the following SQL.

**SELECT** town, contactPerson, emailAddress, district

**FROM** Team

**WHERE** district = "West";

1. Complete the SQL to search the database to display the surname, position, and division of all players who either play in the right wing or centre position.

**SELECT** surname, position, division

**FROM** \_\_\_\_\_\_\_\_\_\_

**WHERE** position = "right wing"

**OR** position = "\_\_\_\_\_\_\_\_";

## EQUI-JOIN between tables

If a search involves displaying data found in two linked tables the SQL clause must state the link. An **EQUI-JOIN** is added to the **WHERE** statement stating that the primary and foreign key values in both tables must match.

**SELECT** *fieldName1, fieldName2, fieldName3*

**FROM** *tableName1, tableName2*

**WHERE** *tableName1.fieldNamePK = tableName2.fieldNameFK*

***AND*** *fieldName = data;*

1. To search the database to display the full name and town of any players who play for a handball team based in Paisley, you would write the following SQL.

**SELECT** firstname, surname, town

**FROM** Team, Player

**WHERE** Team.teamName = Player.teamName

**AND** town = "Paisley";

1. Complete the SQL to search the database to display the full name, number of goals scored and the team name of any players who have scored at least 20 goals for Lothian Flames.

As the teamName field appears in both tables, you have to specify which table you want to use, eg Team.teamName or Player.teamName.

**SELECT** firstname, surname, goalsScored, Team.teamName

**FROM** Team, Player

**WHERE** Team.teamName = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**AND** goalsScored >= \_\_

**AND** Team.teamName = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## Sorting

The **ORDER BY** clause decides how the output of the search should be sorted. ORDER BY is followed by the name of the field and then whether it is ascending (ASC) or descending (DESC) order.

**SELECT** *fieldName1, fieldName2, fieldName3*

**FROM** *tableName*

**WHERE** *fieldName = data*

**ORDER BY** *fieldName ascending or descending;*

1. Complete the SQL to display the full name, team name and goals scored for any player who has scored fewer than 30 goals this season, so that the player with the most goals is listed first.

**SELECT** firstname, surname, Player.teamName, goalsScored

**FROM** Player

**WHERE** goalScored < 30

**ORDER BY** goalScored DESC;

1. Complete the SQL to search the database to display the full name, position, and team name of all goalkeepers who play in the league (listing in alphabetical order of surname; players with the same surname should be listed in alphabetical order of first name).

**SELECT** firstname, surname, position, Player.teamName

**FROM** Player

**WHERE** position = "goalkeeper"

**ORDER BY** \_\_\_\_\_\_\_\_\_\_\_\_\_\_, \_\_\_\_\_\_\_\_\_\_\_

## Adding records

The **INSERT INTO** statement is used to add records to a table. The statement is followed by the table name and then the **VALUES** statement, followed by the data in brackets separated by commas.

**INSERT INTO** *tableName (fieldName1, fieldName2)*

**VALUES** *(value1, value2);*

You must ensure that the *order of the values is the same as the order of the fields.*

1. Details of the newest team to join the league are shown below.

|  |  |
| --- | --- |
| **Team name** | Dundee Dynamos |
| **Town** | Dundee |
| **Contact person** | Paul McLaughlin |
| **E-mail address** | [dynamos@dundeehandball.gmail.com](mailto:dynamos@dundeehandball.gmail.com) |
| **Division** | Second |
| **District** | North |
| **Training venue** | DISC |

When adding a complete record to a database, you could write the following SQL statement using only the values. There must be values for every field and they must be in the same order as the field names in the table.

**INSERT INTO** Team

**VALUES** ("Dundee Dynamos", "Dundee", "Paul McLaughlin", ["dynamos@dundeehandball.gamil.com",](mailto:dynamos@dundeehandball.gamil.com) "Second", "North", "DISC");

When adding only partial record data to a table, both the field names and their associated values must be identified in the SQL statement.

1. A new player has joined the team. The available details about the player are shown below.

|  |  |
| --- | --- |
| **Player ID** | 419AC |
| **First name** | Anatol |
| **Surname** | Czaja |
| **Team name** | Harris Hurricanes |

Complete the SQL to add these details to the database.

**INSERT INTO** Player (playerID, firstname, surname, teamName)

**VALUES** ("419AC", "\_\_\_\_\_\_\_\_", "\_\_\_\_\_\_\_", \_\_\_\_\_\_\_\_\_\_\_\_\_\_);

## Editing records

The **UPDATE** statement is used to alter records in a table. The statement is followed by the name of the table, a **SET** clause and the **WHERE** clause which states what criteria must be met.

**UPDATE** *tableName*

**SET** *fieldName to updated value*

**WHERE** *criteria to be met;*

The player whose ID is 726HB has transferred to a new team and now plays for the Dundee Demons.

To update the correct record of the database you would write the following SQL.

**UPDATE** Player

**SET** Player.teamName = "Dundee Demons"

**WHERE** playerID = "726HB";

1. The contact details for the Airdrie Lions have changed. The team’s contact person is now Lynne Jack and the team e-mail address is now [airdrie@lionshandball.com](mailto:airdrie@lionshandball.com).

To update the correct record of the database, you would write the following SQL.

**UPDATE** Team

**SET** contactPerson = "Lynne Jack", emailAddress = ["airdrie@lionshandball.com"](mailto:airdrie@lionshandball.com)

**WHERE** Team.teamName = "Airdrie Lions";

## Deleting records

The **DELETE FROM** statement is used to delete records in a table. The statement is followed by the name of the table and the **WHERE** clause which states what criteria must be met.

**DELETE FROM** *tableName*

**WHERE** *criteria to be met;*

1. The Borders Bandits have been knocked out of the league.

To remove the correct record from the database, you would write the following SQL.

**DELETE FROM** Team

**WHERE** Team.teamName = "Borders Bandits";

1. Jack Roberts no longer plays in the handball league.

Complete the SQL to delete his record from the database.

**DELETE FROM** Player

**WHERE** firstname = "Jack" **AND** surname = \_\_\_\_\_\_\_\_

Questions

1. Write an SQL query to search the database to display the playerID, Firstname and Surname for all players who are members of the Clyde Flyers team.
2. Write an SQL query to search the database to display the playerID, Firstname and Surname for all players who are members of the Clyde Flyers or Glasgow Rocks teams.
3. Write an SQL query to search the database to display the full name, playerID, town and District of any players who play for a handball team in the West district.
4. Write the SQL that would give the same solution for the query, but this time displaying the Player.teamName field.
5. Write the SQL to search the database to display the full name, team name and goals scored for any player who has scored more than 10 goals this season, so that the player with the least goals is listed first, you would write the following SQL.
6. Write the SQL to search the database to display the full name, number of goals scored and the team name of any players who have scored at least 20 goals for Lothian Flames, displayed in alphabetical order by surname.
7. Write the SQL to add the following team into the database:

|  |  |
| --- | --- |
| **Team name** | Edinburgh Sparks |
| **Town** | Edinburgh |
| **Contact person** | Chris Cunningham |
| **E-mail address** | sparks@edinburghhandball.hotmail.co.uk |
| **Division** | Second |
| **District** | East |
| **Training venue** | Lothian Sports Centre |

1. Write the SQL to add the following details into the database:

|  |  |
| --- | --- |
| **Player ID** | 685DC |
| **First name** | Kim |
| **Surname** | Khan |
| **Team name** | Edinburgh Sparks |

1. The contact person at Edinburgh Sparks has changed to Grant Jones.

Write the SQL to update the details in the database.

1. Write the SQL to delete Edinburgh Sparks from the Team database.

Testing & Evaluation

At National 5 level, you must describe and give an example(s) of appropriate test(s) that would be run to prove that SQL queries produce the expected results. You should document expected results and actual results. A comparison can then be made, with testing passed if the expected result and actual result is the same.

Example

Consider the **Customer** table shown below.



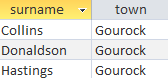
A query is required to display the full name and town of all customers who live in Gourock. The details should be listed in alphabetical order of customer surname.

## Query Testing

An appropriate test would be to run the SQL query and compare the expected output to the actual output.

Expected Results: Actual Results:

|  |  |  |
| --- | --- | --- |
| Ryan | Collins | Gourock |
| Grant | Donaldson | Gourock |
| Rowan | Hastings | Gourock |



## Query Evaluation

To evaluate the tests of an SQL query, you must evaluate whether the SQL

1. is fit for purpose
2. produces accurate output.

*Evaluating the Gourock example above:*

The query is fit for purpose because it displays details of the three customers who live in Gourock and has arranged the details in ascending order of surname.

However, the query output is **not accurate** because the answer table only shows details of surname and town; the forenames of the customers are missing from the answer table.

As the output is not accurate, the design of the SQL should be re-visited and the query re-run to fix the problem.