

Database Design and Development

What is a database?

A database is used to store information. Data is often a company's best commodity, which is why so much of our information is being harvested and sold.

Companies and organisations use databases because:

- they can be searched and sorted efficiently
- a number of people can access and use the same information simultaneously

A well-designed, efficient database allows companies to use our data as effectively as possible.

Database Structure

A database contains **entities** - each entity contain records

One **record** is all data stored about one person or thing

The records contains **attributes** - an attribute is a single piece of information.



hairdresser	clientid	clientfirstname	clientlastname	phonenumber
1928	10290	Wen	Qiu	0141 496 0536
2019	10291	Peta	Mulisdottir	0151 496 0838
1928	11766	Michael	Waters	07700 900556
1928	12654	Faseeha	al-Allam	07700 900569
2019	20533	Egisto	Mario	0131 496 0294
2210	32100	Emily	Sieff	020 7946 0126
2210	36172	Phillip	Roach	0131 496 0734

Database Structure

Entity

Attribute

The screenshot displays the Microsoft Access interface. On the left, the 'All Access Objects' pane shows the 'Tables' section with 'client' and 'hairstresser' listed. The 'client' table is selected. The main window shows the 'client' table's data. Red circles and arrows are used for annotations: one circle around the 'client' table name in the left pane with an arrow pointing to the word 'Entity'; another circle around the 'clientlastname' column header with an arrow pointing to the word 'Attribute'; and a third circle around the entire row for 'Emily' (clientid 32100) with an arrow pointing to the word 'Record'.

hairstresser	clientid	clientfirstname	clientlastname	phonenumber
1928	10290	Wen	Qiu	0141 496 0536
2019	10291	Peta	Mulisdottir	0151 496 0838
1928	11766	Michael	Waters	07700 900556
1928	12654	Faseeha	al-Allam	07700 900569
2019	20533	Egisto	Mario	0131 496 0294
2210	32100	Emily	Sieff	020 7946 0126
2210	36172	Phillip	Roach	0131 496 0734

Flat File Databases

If a database only has one entity then it can be referred to as a **flat-file database**.

client

clientid	clientfirstname	clientlastname	phonenumber	firstname	lastname	contactnumber	salon
10290	Wen	Qiu	0141 496 0536	Phillip	Christie	07700 900142	Cuts & Co
11766	Michael	Waters	07700 900556	Phillip	Christie	07700 900142	Cuts & Co
12654	Faseeha	al-Allam	07700 900569	Phillip	Christie	07700 900142	Cuts & Co
10291	Peta	Mulisdottir	0151 496 0838	Sharon	Watt	07700 900582	On The Corner
20533	Egisto	Mario	0131 496 0294	Sharon	Watt	07700 900582	On The Corner
36172	Phillip	Roach	0131 496 0734	Huda	Quhshi	07700 900477	West Style
32100	Emily	Sieff	0207 946 0126	Huda	Quhshi	07700 900477	West Style

Disadvantages of Flat File Databases

In flat file databases, some information is stored more than once (e.g. the same phone number for multiple members of staff). This results in an increased file size for the database.

Using flat file databases can lead to three very specific problems called anomalies. These anomalies compromise the **integrity** of the data stored.

As a result, **relational databases** are used instead of flat file databases.

Insert Anomaly

Insert anomaly: you cannot add a new department without also adding a member of staff at the same time - there is no way to add a new hairdresser without also adding a client.

client

clientid	clientfirstname	clientlastname	phonenumber	firstname	lastname	contactnumber	salon
10290	Wen	Qiu	0141 496 0536	Phillip	Christie	07700 900142	Cuts & Co
11766	Michael	Waters	07700 900556	Phillip	Christie	07700 900142	Cuts & Co
12654	Faseeha	al-Allam	07700 900569	Phillip	Christie	07700 900142	Cuts & Co
10291	Peta	Mulisdottir	0151 496 0838	Sharon	Watt	07700 900582	On The Corner
20533	Egisto	Mario	0131 496 0294	Sharon	Watt	07700 900582	On The Corner
36172	Phillip	Roach	0131 496 0734	Huda	Quhshi	07700 900477	West Style
32100	Emily	Sieff	0207 946 0126	Huda	Quhshi	07700 900477	West Style

Delete Anomaly

Delete anomaly: you cannot delete data from the table without having to delete the entire record, e.g. if we remove Sharon Watt from the table, we would lose all data about her two clients too, meaning we lose data unnecessarily.

client

clientid	clientfirstname	clientlastname	phonenumber	firstname	lastname	contactnumber	salon
10290	Wen	Qiu	0141 496 0536	Phillip	Christie	07700 900142	Cuts & Co
11766	Michael	Waters	07700 900556	Phillip	Christie	07700 900142	Cuts & Co
12654	Faseeha	al-Allam	07700 900569	Phillip	Christie	07700 900142	Cuts & Co
10291	Peta	Mulisdottir	0151 496 0838	Sharon	Watt	07700 900582	On The Corner
20533	Egisto	Mario	0131 496 0294	Sharon	Watt	07700 900582	On The Corner
36172	Phillip	Roach	0131 496 0734	Huda	Quhshi	07700 900477	West Style
32100	Emily	Sieff	0207 946 0126	Huda	Quhshi	07700 900477	West Style

Update Anomaly

Update anomaly: if data for a salon changed, it would need to be updated in multiple records. If the change only happened in one of the two records, then an update anomaly would have taken place.

client

clientid	clientfirstname	clientlastname	phonenumber	firstname	lastname	contactnumber	salon
10290	Wen	Qiu	0141 496 0536	Phillip	Christie	07700 900142	Cuts & Co
11766	Michael	Waters	07700 900556	Phillip	Christie	07700 900142	Cuts & Co
12654	Faseeha	al-Allam	07700 900569	Phillip	Christie	07700 900142	Cuts & Co
10291	Peta	Mulisdottir	0151 496 0838	Sharon	Watt	07700 900582	On The Corner
20533	Egisto	Mario	0131 496 0294	Sharon	Watt	07700 900582	On The Corner
36172	Phillip	Roach	0131 496 0734	Huda	Quhshi	07700 900477	West Style
32100	Emily	Sieff	0207 946 0126	Huda	Quhshi	07700 900477	West Style

Relational Databases

A **relational database** is a database which contains more than one entity. The entities are linked together using **relationships**.

hairdresser

hairdresserid	firstname	lastname	contactnumber	salon
2210	Huda	Quhshi	07700 900477	West Style
2019	Sharon	Watt	07700 900582	On The Corner
1928	Phillip	Christie	07700 900142	Cuts & Co

client

hairdresserid	clientid	clientfirstname	clientlastname	phonenummer
1928	10290	Wen	Qiu	0141 496 0536
1928	11766	Michael	Waters	07700 900556
1928	12654	Faseeha	al-Allam	07700 900569
2019	10291	Peta	Mulisdottir	0151 496 0838
2019	20533	Egisto	Mario	0131 496 0294
2210	36172	Phillip	Roach	0131 496 0734
2210	32100	Emily	Sieff	0207 946 0126

Topics

Database Design & Development

Analysis

N5 Computing Science

Database Design & Development

Implementation

N5 Computing Science

Database Design & Development

Design

N5 Computing Science

Database Design & Development

Testing and Evaluation

N5 Computing Science

Analysis

End-User Requirements

End users are the people who will use the database and they are unlikely to have any technical knowledge of how the database works.

End-user requirements are the tasks users expect to be able to do using the database.

End-User Requirements Example

In a database used to store information about the hairdressers and clients of a salon, example end-user requirements might be:



I need to be able to add new clients that require an appointment



I need to find clients phone numbers so that I can contact them about their appointments

As the owner of 3 salons, I need to find the hairdressers that work at each salon



Functional Requirements

Functional requirements are processes and activities that the databases has to carry out.

Processes include **adding, removing, updating, searching or sorting.**

Functional Requirements Example

Functional requirements should not be generic and usually relate very closely to end-user requirements, so examples of functional requirements for a database used to store salon information might be:

- Insert new clients to the client entity
- Search for hairdresser details for each salon
- Display contact details (phone number) of clients

Design

Design Definitions

Primary Key: a primary key is the attribute that uniquely identifies a record within an entity. The primary key is a required attribute, so it should always have presence check validation.

Foreign Key: a foreign key is the primary key of another entity, and is used to link two tables together.

Field Types

When creating entities in a database, you need to define the type of information that will be held in each field. At National 5 you need to know the following data types:

Text: any data that is represented using a string of text. This includes values with letters and numbers (e.g. post codes) and numbers that start with a zero (e.g. phone numbers)

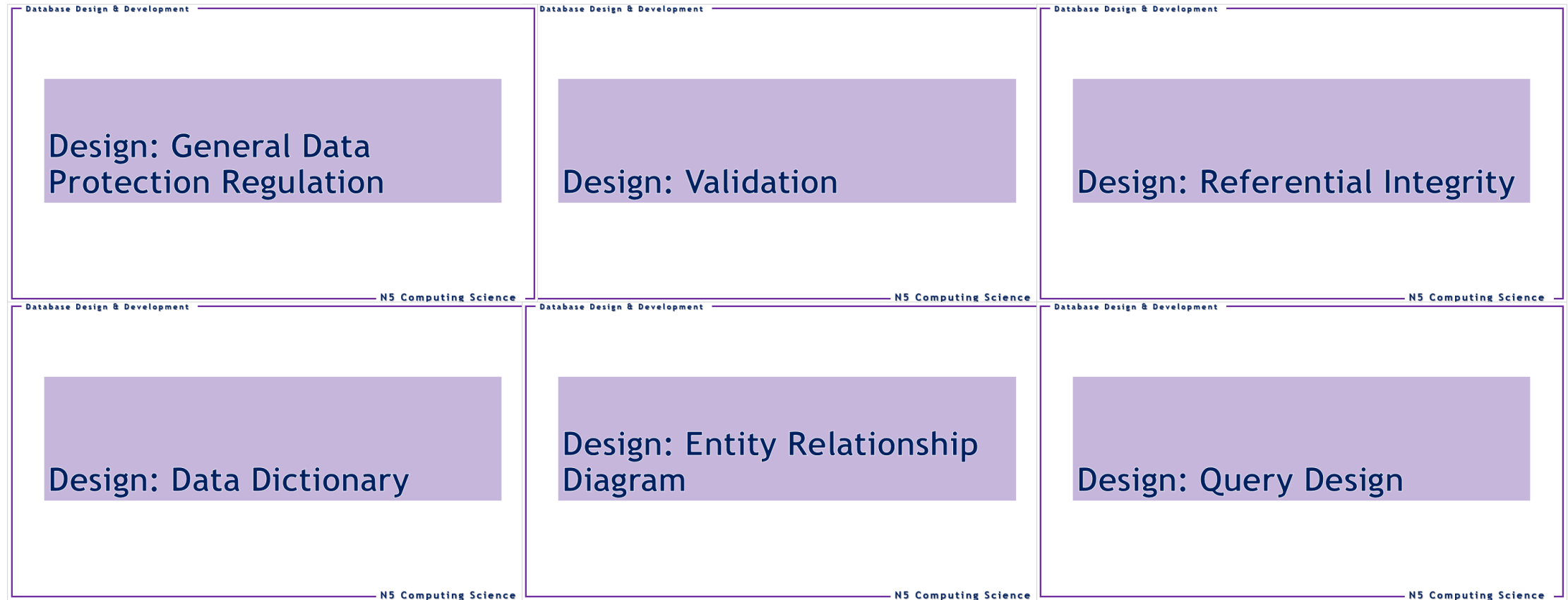
Number: any integer or real data values

Date: any date, in the format DD/MM/YYYY

Time: any time, in the format HH:MM

Boolean: a true or false value i.e. a checkbox

Design Contents



Design: General Data Protection Regulation

Definition

The General Data Protection Regulation (GDPR) is a set of rules designed to give EU citizens greater control over their personal data. The EU GDPR will be brought into UK law and will still apply in the UK even after Brexit.

The regulation replaces existing data protection laws and updates them so that they are relevant in the internet age.

The regulation requires that those that hold data are transparent about how they gather and use that data.

GDPR: What Is It and How Might It Affect You?



Definitions Within GDPR

Data subject: the individual whose data is being collected.

Data controller: the organisation that is using personal data and managing how that data is processed.



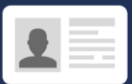







Data processor: a third-party who processes the data for the controller.

Privacy notice: the information provided by the data controller that tells the data subject how their information will be processed and how long it will be kept.

What is Personal Data?

Any information about you that can be used to identify you is **Personal Data**. You can also be individually identified by a combination of data.

Sensitive Personal Data is also protected under GDPR, and often has even tighter rules than standard Personal Data.

Personal Data	Sensitive Personal Data
 Names	 Health Data
 Location Data	 General Data
 Identification Numbers	 Biometric Data
 IP Addresses	 Racial or Ethnic Data
 Cookie Data	 Political Opinions
 RFID Tags	 Sexual Orientation

Rights of the Individual

The regulation gives individuals 8 specific rights about how their data is processed.

1. Right to be informed
2. Right to access
3. Right to rectification
4. Right to erasure
5. Right to restrict processing
6. Right to data portability
7. Right to object
8. Rights related to automated decision making and profiling

Requirements of the Data Controllers

The rights of data subjects mean that there are requirements placed on organisations about how data is handled.

1. processed lawfully, fairly and in a transparent manner in relation to individuals
2. used for the declared purpose only
3. limited to the data needed for the declared purpose
4. accurate
5. not kept for longer than necessary
6. held securely

Design: Validation

Definition

Validation is used to check that the data entered for an attribute is what is expected. There are four types of validation that can be applied to an attribute:

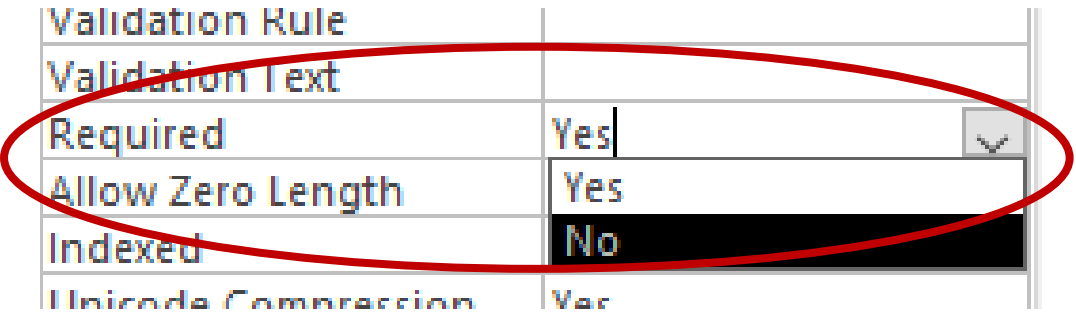
- Presence check
- Length check
- Range check
- Restricted choice



Presence Check

A presence check makes sure the person entering data cannot leave this attribute blank. For example, when signing up for a new online account, you cannot leave the email address field empty.

Microsoft Access: implement presence check validation in the Design View of Access. Set the Required property to 'Yes'.



Validation Rule	Validation Text
Required	Yes
Allow Zero Length	Yes
Indexed	No
Unicode Compression	Yes

Length Check

The length validation rule makes sure the minimum and/or maximum number of characters have been entered for attributes with a 'Text' data type.

Microsoft Access: implement Length Check validation in the Design View of Access. Set the Validation Rule property to an appropriate rule.

For example, `Len([FieldName])=5`, replacing `FieldName` with the name of the attribute that requires the validation.

General Lookup	
Field Size	255
Format	
Input Mask	
Caption	
Default Value	
Validation Rule	<code>Len([First name]) >= 3</code>
Validation Text	
Required	Yes
Allow Zero Length	Yes
...	...

Length Check

Length validation can check for:

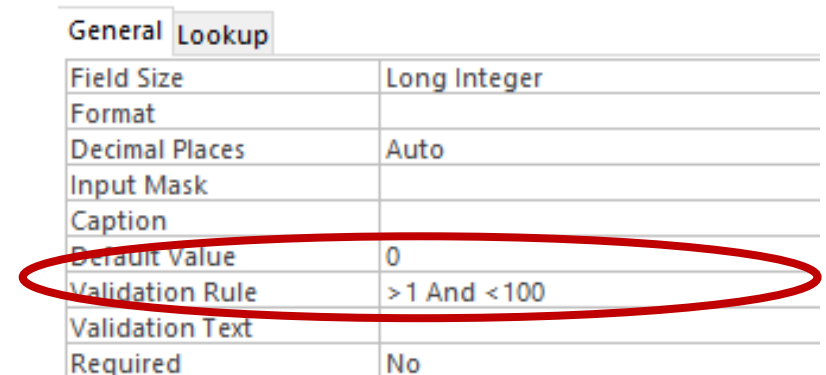
- an exact number of characters, e.g. that a phone number is 11 digits long
- a minimum number of characters, e.g. that your password is at least 8 characters long
- a maximum number of characters, e.g. the maximum length of a Tweet is 280 characters
- a mix of both, e.g. your username must be more than 8 characters and less than 16 characters long

Range Check

The value of an attribute with a Number data type can be limited to values between a maximum and a minimum, giving a range of possible values. For example between 0 and 10 or between -25 and 25.

Microsoft Access: implement Range Check validation in the Design View of Access. Set the Validation Rule property to an appropriate rule.

When validating times, you must surround the values with a # symbol, e.g. `>=#16:00:00#` and `<=#18:00:00#`



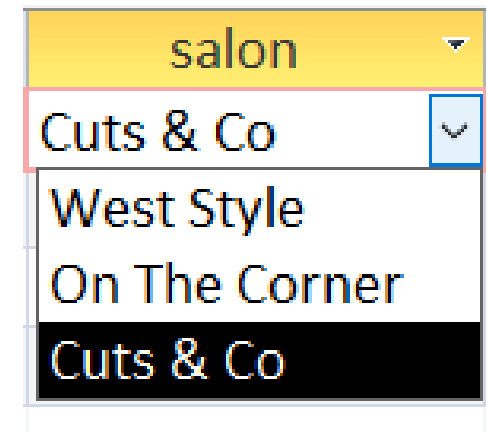
The screenshot shows the 'Design View' of a table in Microsoft Access. The 'Lookup' tab is selected. The 'Validation Rule' property is highlighted with a red oval and set to '> 1 And < 100'. The 'Default Value' is set to '0'. The 'Field Size' is 'Long Integer', 'Format' is 'Auto', 'Input Mask' is empty, 'Caption' is empty, 'Validation Text' is empty, and 'Required' is 'No'.

General Lookup	
Field Size	Long Integer
Format	
Decimal Places	Auto
Input Mask	
Caption	
Default Value	0
Validation Rule	> 1 And < 100
Validation Text	
Required	No

Restricted Choice

A restricted choice validation rule limits the values for an attribute to a selection of predefined options. For example, jackets available in sizes S, M, L, XL, XXL, these are the only allowed values for the attribute.

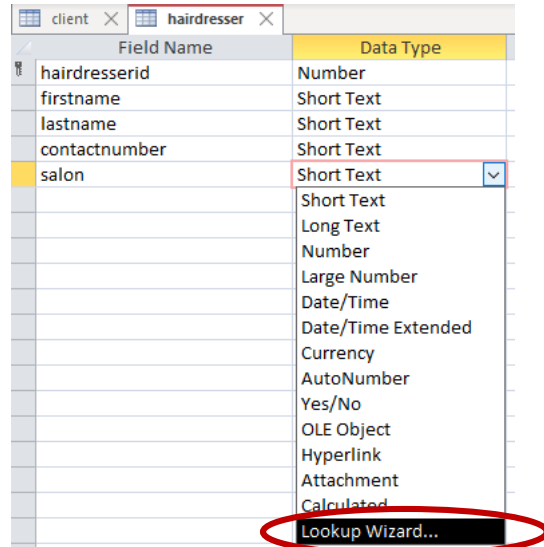
Restricted choice helps to ensure that data integrity is maintained, as users cannot make spelling mistakes when adding information.



salon ▼

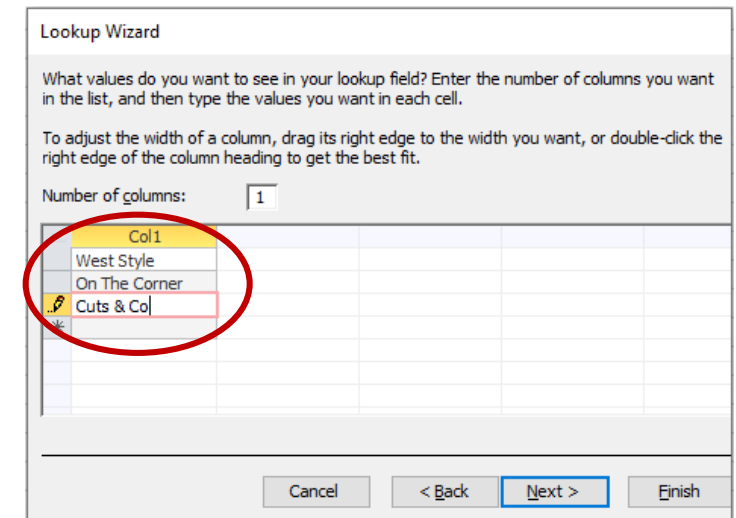
- Cuts & Co ▼
- West Style
- On The Corner
- Cuts & Co

Restricted Choice



Microsoft Access: implement Restricted Choice validation in the Design View of Access. Set the Data Type to 'Lookup Wizard'. Select 'I will type in the values that I want'.

Enter the values allowed in restricted choice validation. This will mean that the user will have a drop-down when adding information instead of having to type.

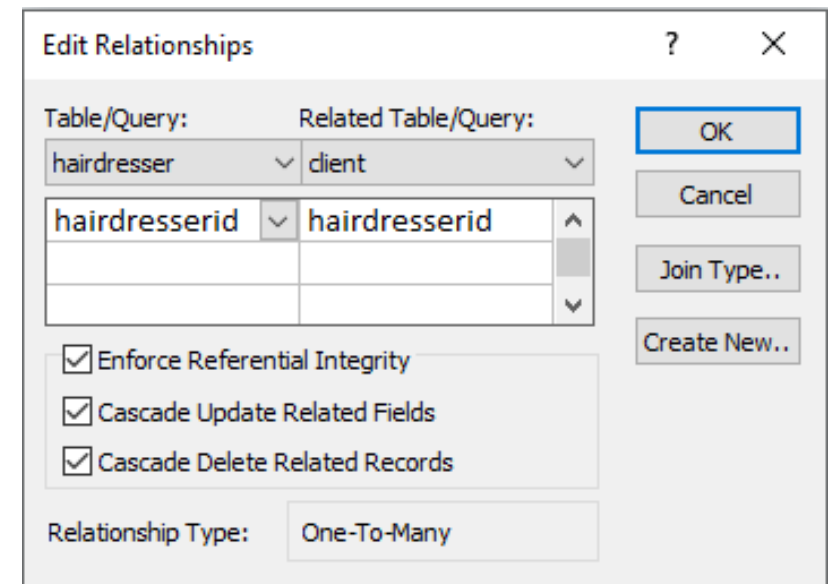


Design: Referential Integrity

Definition

Referential Integrity is process of **only** allowing foreign key values that have matching primary key values in a related table.

Referential Integrity is a version of restricted choice where the values are restricted to those entered in another table. This is used to ensure that a value entered in for the foreign key has a matching value as a primary key in the related table.



Referential Integrity Example

Referential integrity means that the relationship between two tables should always be consistent.

In the hairdresser/client database, the hairdresserid is the primary key of the hairdresser table and it is also a foreign key in the client table.

hairdresser

hairdresserid	firstname	lastname	contactnumber	salon
2210	Huda	Quhshi	07700 900477	West Style
2019	Sharon	Watt	07700 900582	On The Corner
1928	Phillip	Christie	07700 900142	Cuts & Co

client

hairdresserid	clientid	clientfirstname	clientlastname	phonenumber
1928	10290	Wen	Qiu	0141 496 0536
1928	11766	Michael	Waters	07700 900556
1928	12654	Faseeha	al-Allam	07700 900569
2019	10291	Peta	Mulisdottir	0151 496 0838
2019	20533	Egisto	Mario	0131 496 0294
2210	36172	Phillip	Roach	0131 496 0734
2210	32100	Emily	Sieff	0207 946 0126

Referential Integrity Example

The client table is impacted if Huda Quhshi is removed from the hairdresser table.

As hairdresserid is a foreign key in the client table, there is no longer a corresponding primary key value of 2210 for hairdresserid in the hairdresser table. The tables will look like this.

hairdresser

hairdresserid	firstname	lastname	contactnumber	salon
2019	Sharon	Watt	07700 900582	On The Corner
1928	Phillip	Christie	07700 900142	Cuts & Co

client

hairdresserid	clientid	clientfirstname	clientlastname	phonenumber
1928	10290	Wen	Qiu	0141 496 0536
1928	11766	Michael	Waters	07700 900556
1928	12654	Faseeha	al-Allam	07700 900569
2019	10291	Peta	Mulisdottir	0151 496 0838
2019	20533	Egisto	Mario	0131 496 0294
2210	36172	Phillip	Roach	0131 496 0734
2210	32100	Emily	Sieff	0207 946 0126

?

Referential Integrity Example

Deleting the row for Huda Quhshi creates inconsistent rows in the client table - it breaks the link between the two tables.

Enforcing referential integrity means that there cannot be foreign key values which do not have a corresponding primary key value in the linked table.

- It prevents breaking the link between primary and foreign keys in related tables.
- It prevents new rows being added in a table where the foreign key does not have a matching primary key value in the linked table.

Cascade Deletes

If referential integrity is enforced then removing a row which has the primary key value should also delete all the rows in the linked table with the related foreign key. This process is called **cascade delete**.

hairdresser

hairdresserid	firstname	lastname	contactnumber	salon
2019	Sharon	Watt	07700 900582	On The Corner
1928	Phillip	Christie	07700 900142	Cuts & Co

client

hairdresserid	clientid	clientfirstname	clientlastname	phonenumber
1928	10290	Wen	Qiu	0141 496 0536
1928	11766	Michael	Waters	07700 900556
1928	12654	Faseeha	al-Allam	07700 900569
2019	10291	Peta	Mulisdottir	0151 496 0838
2019	20533	Egisto	Mario	0131 496 0294

So, deleting the row for Huda Quhshi in the hairdresser table will also delete all the related rows in the client table.

Cascade Updates

If there is an update to the primary key value, then it is important that all the linked foreign keys are also updated. This process is called **cascade update** - any changes to a primary key value are also copied to all the related foreign key values.

hairdresser

hairdresserid	firstname	lastname	contactnumber	salon
3982	Sharon	Watt	07700 900582	On The Corner
1928	Phillip	Christie	07700 900142	Cuts & Co

client

hairdresserid	clientid	clientfirstname	clientlastname	phonenummer
1928	10290	Wen	Qiu	0141 496 0536
1928	11766	Michael	Waters	07700 900556
1928	12654	Faseeha	al-Allam	07700 900569
3982	10291	Peta	Mulisdottir	0151 496 0838
3982	20533	Egisto	Mario	0131 496 0294

If the primary key for Sharon Watt is changed from 2019 to 3982 in hairdresser, then all the related foreign key values in client are also updated to 3982.

Design: Data Dictionary

Definition

Data dictionaries is the detailed design for one or more entities, and defines the structure of the database. A data dictionary contains metadata, which is **data about data**.

A data dictionary is laid out as a table containing the definition of each attribute and entity:

- entity names
- attribute names
- primary and foreign
- data types
- attribute size
- validation

Data Dictionary Example

The next slide shows the data dictionary for these entities.

hairdresser

hairdresserid	firstname	lastname	contactnumber	salon
2210	Huda	Quhshi	07700 900477	West Style
2019	Sharon	Watt	07700 900582	On The Corner
1928	Phillip	Christie	07700 900142	Cuts & Co

client

hairdresserid	clientid	clientfirstname	clientlastname	phonenummer
1928	10290	Wen	Qiu	0141 496 0536
1928	11766	Michael	Waters	07700 900556
1928	12654	Faseeha	al-Allam	07700 900569
2019	10291	Peta	Mulisdottir	0151 496 0838
2019	20533	Egisto	Mario	0131 496 0294
2210	36172	Phillip	Roach	0131 496 0734
2210	32100	Emily	Sieff	0207 946 0126

Data Dictionary Example

Entity name: hairdresser					
Attribute name	Key	Type	Size	Required	Validation
hairdresserid	PK	Number	4	Y	range ≥ 1000 AND ≤ 9999
firstname		Text		Y	
lastname		Text		Y	
contactnumber		Text		Y	
salon		Text		Y	restricted choice: West Style, On The Corner and Cuts & Co

Entity name: client					
Attribute name	Key	Type	Size	Required	Validation
hairdresserid	FK	Number	4	Y	existing hairdresserid from hairdresser table
clientid	PK	Number	5	Y	
clientfirstname		Text		Y	length > 3
clientlastname		Text		Y	
phonenumber		Text		Y	

Data Dictionary Example

Entity name: hairdresser					
Attribute name	Key	Type	Size	Required	Validation
hairdresserid	PK	Number	4	Y	range ≥ 1000 AND ≤ 9999
firstname		Text		Y	
lastname		Text		Y	
contactnumber		Text		Y	
salon		Text		Y	restricted choice: West Style, On The Corner and Cuts & Co

Range check validation

Restricted choice validation

Presence check validation

Entity name: client					
Attribute name	Key	Type	Size	Required	Validation
hairdresserid	FK	Number	4	Y	existing hairdresserid from hairdresser table
clientid	PK	Number	5	Y	
clientfirstname		Text		Y	length > 3
clientlastname		Text		Y	
phonenumber		Text		Y	

Referential integrity

Length check validation

Design: Entity Relationship Diagram

Definition

An Entity Relationship Diagram is a graphical representation used to illustrate the relationship that exists between two or more entities.

Entity Relationship Diagrams show:

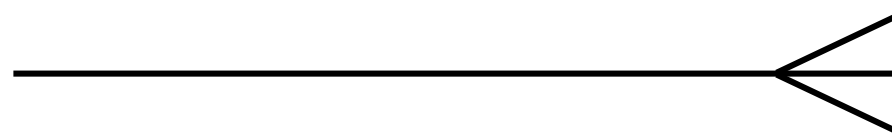
- attributes
- relationship cardinality
- relationship names

Cardinality

Cardinality refers to the relationship of data in one entity with respect to another entity. A **one-to-many** relationship exists when one entity can be present in many different instances of another entity.

A one-to-many relationship is shown with an arrow with crow's feet. The single line shows the 'one' side of the relationship, while the crows feet show the 'many' side of the relationship

'one' side of the
relationship



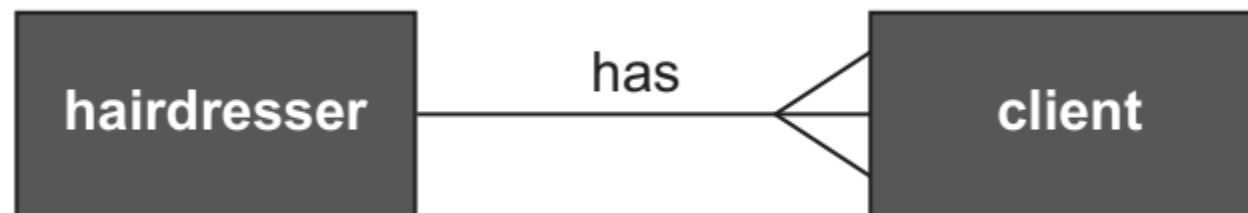
'many' side of
the relationship

Relationship Names

Relationships between entities requires a short phrase to help to describe the relationship. For example “one user **posts** many updates” or “one company **employs** many people”.

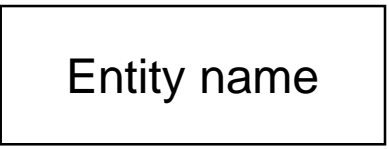
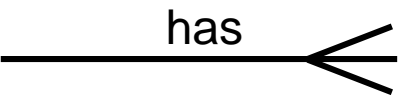



Often, if you cannot think of an appropriate description, “has” will work instead.

Here is a simple ERD showing the relationship name and cardinality:

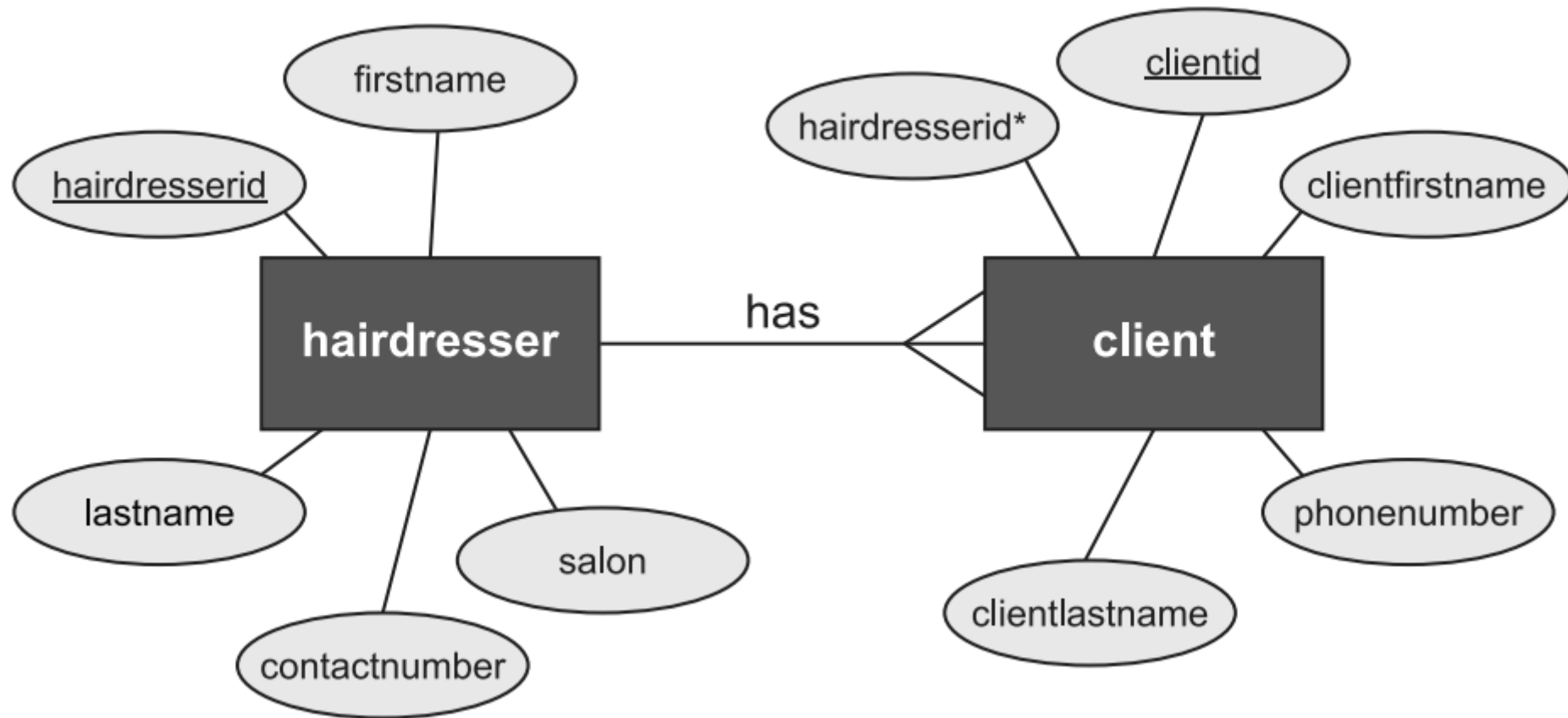


ERD Symbols

ERDs illustrates entities, relationships and attributes. Entity relationship diagrams make use of the following symbols.

	A rectangle is used to represent each entity. The name of the entity set is entered inside the rectangle to identify it. The name should be singular (e.g. person rather than people).
	The relationship between the two entity sets. A short phrase is written above the line to describe the relationship
	Attributes can be added to the ERD as ovals. The name of the attribute is entered inside the oval.
	If an attribute is the primary key then its name is underlined inside the oval.
	If an attribute is the foreign key then an asterix is put beside its name.

ERD Example



Design: Query Design

Query Design

Query design is used for queries that may be implemented later. Query design doesn't use SQL.

The design of a query requires that you define:

- tables to be used in the query
- fields to be used in the query
- search criteria
- sort order

This is usually used to design **SELECT** queries to display information from entities.

Query Design

To represent the design of a query, the following layout is usually used.

Table(s)	
Field(s)	
Search criteria	
Sort order	

Query Design Example

The next slide shows the design for a query that will display clientfirstname and phonenumber for clients where the hairdresserid is 2210, sorted by clientfirstname descending.

hairdresser

hairdresserid	firstname	lastname	contactnumber	salon
2210	Huda	Quhshi	07700 900477	West Style
2019	Sharon	Watt	07700 900582	On The Corner
1928	Phillip	Christie	07700 900142	Cuts & Co

client

hairdresserid	clientid	clientfirstname	clientlastname	phonenumber
1928	10290	Wen	Qiu	0141 496 0536
1928	11766	Michael	Waters	07700 900556
1928	12654	Faseeha	al-Allam	07700 900569
2019	10291	Peta	Mulisdottir	0151 496 0838
2019	20533	Egisto	Mario	0131 496 0294
2210	36172	Phillip	Roach	0131 496 0734
2210	32100	Emily	Sieff	0207 946 0126

Query Design Example

Design to display clientfirstname and phonenumber for clients where the hairdresserid is 2210, sorted by clientfirstname descending.

Field(s)	clientfirstname, phonenumber
Table(s)	client
Search criteria	hairdresserid = 2210
Sort order	clientfirstname descending

Implementation

Queries

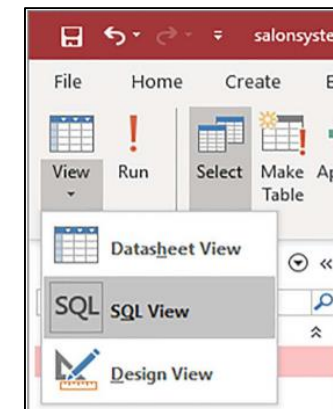
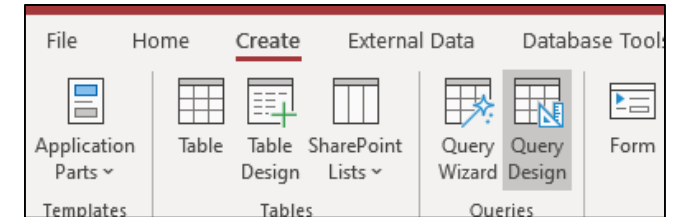
To add, display, change or delete data from a database, we use a special programming language called **SQL** (sometimes pronounced SEQUEL).

There are four queries that you need to know for National 5:

- SELECT
- INSERT
- UPDATE
- DELETE

Creating Queries in Microsoft Access

1. Select the Create ribbon and then click Query Design
2. Close the table selection pop up that appears
3. Select SQL View

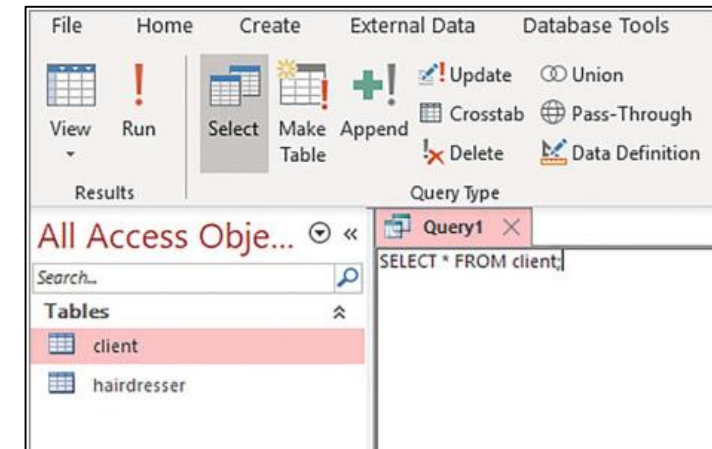


Creating Queries in Microsoft Access

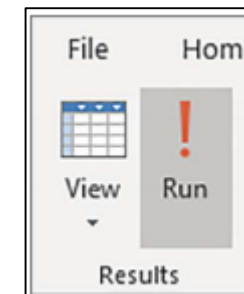
2. H

3. H

4. Write your SQL in the SQL Editor



5. Once written, execute the query by clicking the Run button



SELECT Query Structure

The SELECT query is used to display information that is stored in the database. The SELECT query has up to four clauses:

SELECT	Mandatory	Defines which attributes are going to be displayed in the output.
FROM	Mandatory	Specifies which entity/entities to get the data from.
WHERE	Optional	Identifies the data required by specifying a condition that the outputted data must meet.
ORDER BY	Optional	Defines the attribute(s) that should be ordered, and how they should be ordered (ascending or descending).

SELECT Query Example

```
SELECT clientfirstname, phonenumber  
FROM client  
WHERE hairdresserid = 2210  
ORDER BY clientfirstname desc;
```

Output:

clientfirstname	phonenumber
Phillip	0131 496 0734
Emily	0207 946 0126

SELECT Query Example

We can also easily create a SELECT query from a design.

Field(s)	clientfirstname, phonenumber
Table(s)	client
Search criteria	hairstresserid = 2210
Sort order	clientfirstname descending



```
SELECT clientfirstname, phonenumber
```

```
FROM client
```

```
WHERE hairstresserid = 2210
```

```
ORDER BY clientfirstname desc;
```

SELECT Query Structure

The asterisk (*) represents all attributes, so `SELECT *` would display every attribute.

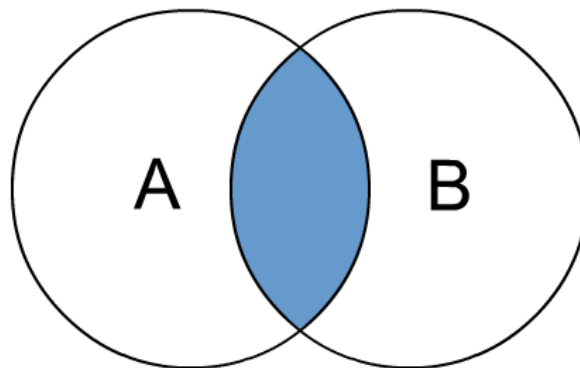
Ascending and descending are written shorthand in SQL queries as 'asc' and 'desc' respectively.

All SQL queries, including SELECT queries, end with a semi-colon.

SELECT Using EQUI-JOIN

Sometimes you need to combine data from two entities to create the desired output. This is achieved using an **EQUI-JOIN**.

An EQUI-JOIN only displays data from records if there are matching values in both tables, i.e. if the value of the primary key in one table equals the value of the foreign key in the linked table.



EQUI-JOIN Example

The structure of a SELECT query using an EQUI-JOIN is the same as a basic SELECT statement.

To implement an EQUI-JOIN, you need an extra condition in the WHERE clause to check that the **value** of the primary key in one entity equals the **value** of the foreign key in the linked entity, e.g.

```
WHERE hairdresser.hairdresserid = client.hairdresserid;
```

EQUI-JOIN Example

```
SELECT clientfirstname, clientlastname, salon  
FROM client, hairdresser  
WHERE hairdresser.hairdresserid = client.hairdresserid;
```

Output:

clientfirstname	clientlastname	salon
Wen	Qiu	Cuts & Co
Michael	Waters	Cuts & Co
Faseeha	al-Allam	Cuts & Co
Peta	Mulisdottir	On The Corner
Egisto	Mario	On The Corner
Phillip	Roach	West Style
Emily	Sieff	West Style

INSERT Query Structure

The INSERT query is used to add new records into the database. The INSERT query has two clauses:

INSERT INTO	Mandatory	Defines which entity the record will be entered into, and the attributes that will have data entered.
VALUES	Mandatory	Specifies the data which will be added to the database.

The order of values in the VALUES clause **must match** the order of the attributes in the INSERT INTO clause or the wrong data will be added to the wrong attributes.

INSERT Query Example

```
INSERT INTO client (hairstresserid, clientid,  
clientfirstname, clientlastname, phonenumber)  
VALUES (2019, 38921, "Rebecca", "Stewart",  
"01317649127");
```

Result:

client

hairstresserid	clientid	clientfirstname	clientlastname	phonenumber
1928	10290	Wen	Qiu	0141 496 0536
1928	11766	Michael	Waters	07700 900556
1928	12654	Faseeha	al-Allam	07700 900569
2019	10291	Peta	Mulisdottir	0151 496 0838
2019	20533	Egisto	Mario	0131 496 0294
2019	38921	Rebecca	Stewart	01317649127
2210	36172	Phillip	Roach	0131 496 0734
2210	32100	Emily	Sieff	0207 946 0126

UPDATE Query Structure

The UPDATE query is used to modify information that is stored in the database. The UPDATE query has three clauses:

UPDATE	Mandatory	Defines which entity is going to be updated.
SET	Mandatory	Specifies the attribute(s) that will be modified, and the new value. Separate multiple values with a comma.
WHERE	Optional	Identifies the records to be updated by specifying a condition that the data must meet to be changed.

The WHERE clause is **very important** – if you miss it out then you will update every record in the entity. Save a copy of your database before running an UPDATE query just in case!

UPDATE Query Example

```
UPDATE client  
SET phonenumber = "01318726418"  
WHERE clientid = 11766;
```

Result: client

hairstresserid	clientid	clientfirstname	clientlastname	phonenumber
1928	10290	Wen	Qiu	0141 496 0536
1928	11766	Michael	Waters	01318726418
1928	12654	Faseeha	al-Allam	07700 900569
2019	10291	Peta	Mulisdottir	0151 496 0838
2019	20533	Egisto	Mario	0131 496 0294
2210	36172	Phillip	Roach	0131 496 0734
2210	32100	Emily	Sieff	0207 946 0126

DELETE Query Structure

The DELETE query is used to remove information from the database. The DELETE query has three clauses:

DELETE	Mandatory	States that a record(s) will be deleted.
FROM	Mandatory	Specifies which entity/entities to remove the data from.
WHERE	Optional	Identifies the data to be deleted by specifying a condition that the data must meet.

The WHERE clause is **very important** - if you miss it out then you will delete every record in the entity. Save a copy of your database before running a DELETE query just in case!

DELETE Query Example

```
DELETE  
FROM client  
WHERE hairdresserid = 2210;
```

Result:

client

hairdresserid	clientid	clientfirstname	clientlastname	phonenummer
1928	10290	Wen	Qiu	0141 496 0536
1928	11766	Michael	Waters	07700 900556
1928	12654	Faseeha	al-Allam	07700 900569
2019	10291	Peta	Mulisdottir	0151 496 0838
2019	20533	Egisto	Mario	0131 496 0294

Testing and Evaluation

Testing

Any queries that are written have to be tested to make sure that they are **accurate**.

To test a query:

1. Write down the expected output of running the query
2. Compare the actual output to the expected impact

If the expected output matches the actual output, then the query is found to be **accurate**.

Testing Example

A query is required to display the first name and last name of all hairdressers from the West Style salon.

hairdresser

hairdresserid	firstname	lastname	contactnumber	salon
2210	Huda	Quhshi	07700 900477	West Style
2019	Sharon	Watt	07700 900582	On The Corner
1928	Phillip	Christie	07700 900142	Cuts & Co

The expected output is shown here:

firstname	lastname
Huda	Quhshi

Testing Example

A query is created to display this information:

```
SELECT hairdresserid, firstname, lastname  
FROM hairdresser  
WHERE salon = "West Style";
```

This query generates the following output

hairdresserid	firstname	lastname
2210	Huda	Quhshi

Testing Example

Expected output

firstname	lastname
Huda	Quhshi

Actual output

hairdresserid	firstname	lastname
2210	Huda	Quhshi

Comparing the actual output with the expected output shows they are nearly the same, but not exactly. The actual output contains the required data but also has an additional column not asked for.

This means the query is **not accurate**, and fails the testing phase.

Evaluation

Any queries that are written also have to be evaluated to make sure that they are **fit for purpose**.

When evaluating fitness for purpose, you are answering the question “does this query do what it was meant to do?”.

You can also decide if the database is fit for purpose if it meets the end-user and functional requirements gathered during the analysis phase.

Evaluation Example

A query is required to display the first name and last name of all hairdressers from the West Style salon.

hairdresser

hairdresserid	firstname	lastname	contactnumber	salon
2210	Huda	Quhshi	07700 900477	West Style
2019	Sharon	Watt	07700 900582	On The Corner
1928	Phillip	Christie	07700 900142	Cuts & Co

The expected output is shown here:

firstname	lastname
Huda	Quhshi

Evaluation Example

A query is created to display this information:

```
SELECT hairdresserid, firstname, lastname  
FROM hairdresser  
WHERE salon = "West Style";
```

This query generates the following output

hairdresserid	firstname	lastname
2210	Huda	Quhshi

Evaluation Example

Expected output

firstname	lastname
Huda	Quhshi

Actual output

hairstresserid	firstname	lastname
2210	Huda	Quhshi

Comparing the actual output with the expected output shows they are nearly the same, but not exactly. The actual output contains an additional column, but the required information is present.

The means the query is **fit for purpose**, and passes the evaluation phase.