Higher

Computing Science

Database Design and Development

Design

Name:____________________________

Contents

Revision

What is a database?

A **database** is used to store information. Databases can contain thousands of pieces of information, stored in a variety of formats

Databases are used as they can be searched and sorted very efficiently and they can allow a number of people to use the same information simultaneously.

Database Structure

Databases contain tables. Each **table** contains records. One **record** is all the data stored about one person or one object.

The records contain fields; a **field** is one single piece of information.

All the records in a table must have the same fields. Fields can have many different data types.

Linking Tables

Databases can contain more than one table and these tables can be linked using **primary keys** and **foreign keys**.

Types of Computerised Database

There are two types of computerised database

- Flat file Database
- Relational Database

Flat File Database

A flat file is **not** the most efficient way to store data.

Member ID	Initial	Surname	Title	Postcode	Tele	Dog Name	Gender	DofB	Breed	Origin	Life Expectancy
1	A	Fish	Mrs	CV35QW	02476111111	Bongo	M	21/08/09	Poodle	China	5
1	A	Fish	Mrs	CV35QW	02476111111	Jiccup	F	08/08/08	Poodle	China	5
1	A	Fish	Mrs	CV53QW	02476111111	Rizla	F	09/09/10	Poodle	China	5
1	A	Fish	Mrs	CV35QW	02476111111	Gov	F	11/01/11	Alsatian	Germany	10
2	C	Here	Mrs	CV27RF	01788222222						
3	D	Lapidated	Mr	CV14RR	02476333333	Manic	M	11/01/11	Poodle	China	5
3	D	Lapidated	Mr	CV14RR	02476333333	Blip	F	02/02/11	Spaniel	France	7
4	V	Ray	Ms	CV12YY	02476444444						
5	Y	Nott	Mr	CV24TT	01788555555	Ruff	M	08/08/10	Poodle	China	5
5	Y	Nott	Mr	CV24TT	01788555555	Addi	M	10/02/10	Poodle	China	5
5	Y	Nott	Mr	CV24TT	01788555555	Catnip	F	10/03/99	Poodle	China	5
5	Y	Nott	Mr	CV24TT	01788555555	Emmi	F	11/03/11	Poodle	China	5
5	Y	Nott	Mr	CV24TT	01788555555	Gov	F	11/01/11	Alsatian	Germany	10
……	……	……	……	……	……	……	……	……	……	……	……
……	……	……	……	……	……	……	……	……	……	……	……

The unnecessary duplication of data values not only wastes memory, it can lead to **modification errors.**

As a result, flat file databases lead to **inconsistencies.**

Example 1

When member 5, Y Nott, moves house, the club secretary will need to change the postcode five times.

*The repeated modification of the same data item can lead to inconsistencies. This type of **modification error** occurs when repeated data is **updated**.*

Example 2

When the details of D Lapidated, are removed from the database, the details of both of his dogs will also be removed.

*If this is the only member with a spaniel, the club will lose the only record containing information about the breed. This type of **modification error** occurs when data is **deleted**.*

Example 3

As the data is organised at present, there is no way to add details of a new breed without first adding details of a member who owns a dog of that breed.

*This type of **modification error** occurs when there is an attempt to **insert** new data.*

Relational Database

A **relational database** is a database which contains more than one table. The tables are linked together by using primary and foreign keys.

Relational databases avoid the issues caused by flat file databases.

The design of a relational database is mainly concerned with three things:

-Entities

-Attributes

-Relationships

Entities and Attributes

An **entity** is a person, place, thing, event or concept of interest to the business or organisation about which data is to be stored.

For example, in a school, possible entities might be Student, Teacher, Class and Subject.

Any system can be represented as a collection of one or more ‘objects’, ‘things’ or **entities**

A specific example of an entity is called an instance or **entity occurrence**.

For example, John Smith, Mary McLeod and Omar Shaheed are all entity occurrences found in the Student entity; English, Computing and Chemistry are all entity occurrences within the Subject entity.

An entity is described by its **attributes**. Each attribute is a characteristic of the entity.

For example, **attributes** of the Student entity would include studentID, firstname, surname and dateOfBirth.

Primary Key

All entities **must** have a **primary key**.

A **primary key** is one or more attributes whose values are used to uniquely identify an individual record in the database

To distinguish primary keys from other attributes it is normal to use underlining.

<u>Customer Number</u>	Surname	Address	Balance Owning
5567	Jones	Cross Road	2.50
2913	Anderson	River Lane	0.00
4890	Murray	West Street	1.50
1622	Jones	Mill Lane	3.00
……	……	……	……

Compound Key

In some entities, no **one** attribute can be used as a primary key.

Property ID	Seller ID	Buyer ID	Buyer Name	Viewing Date
P101	Smith	1282	Jones	17/04/14
P101	Smith	1982	Peters	19//04/14
P101	Smith	2983	Patel	29/05/14
P101	Smith	1282	Jones	29/05/14
P106	Parker	1282	Jones	30/05/14
P106	Parker	7225	Mitchell	23/04/14

There is no single attribute here that can be used to uniquely identify each record in the entity.

One solution is to use a combination of two or more attributes to create a unique identifier.

<u>Property ID</u>	Seller ID	<u>Buyer ID</u>	Buyer Name	<u>Viewing Date</u>
P101	Smith	1282	Jones	17/04/14
P101	Smith	1982	Peters	19//04/14
P101	Smith	2983	Patel	29/05/14
P101	Smith	1282	Jones	29/05/14
P106	Parker	1282	Jones	30/05/14
P106	Parker	7225	Mitchell	23/04/14

The combination of *PropertyID*, *BuyingID* and *Viewing Date* creates a unique identifier for each record in the entity.

This is known as a **compound key.**

Foreign Key

A **foreign key** is an attribute in one entity that is the **primary key** of another entity.

Foreign keys are used to **link** entities in a relational database. Some complex entities have more than one foreign key.

Product ID	Product Name	Category	Unit Price	Supplier Code
P001	Corn flakes	Grocery	1.39	OB1
P002	Corn flakes	Grocery	1.95	KG1
P003	Sardines	Grocery	0.48	JW1
P004	Tea bags	Grocery	2.47	OB1
P005	Carrots	Fruit Veg	0.73	OB1
P006	Milk	Dairy	0.86	OB1
……	……	……	……	……

Supplier Code	Supplier	Telephone	Email
OB1	Own Brand	01715217348	ownbrand@super,com
KG1	Kelloggs	01234567891	kelloggs@cereals.co.uk
JW1	John West	01812136453	johnwest@oceans.com
……	……	……	……

This example shows how information about products and suppliers is presented.

- The primary key of the Product entity is **Product ID**.
- The primary key of the Supplier entity is **Supplier Code**
- **Supplier Code** in the Product entity is the **foreign key** which links the Product entity to the Supplier entity

Practise Questions – Compound Key

Task 1

The Player table below is used to store details of footballers who play in a local junior league. Determine whether values stored in the fields of the table would be unique and give a reason in each case.

Player			
Field	**Type**	**Unique?**	**Reason**
fullName	Text		
position	Text		
shirtNumber	Number		
injured	Boolean		
team	Text		

Suggest a compound key that could be used as the primary key of the Player table.

Compound key:

Task 2

The House table below is used to store details of houses for sale in an Estate Agency. Determine whether values stored in the fields of the table would be unique and give a reason in each case.

House			
Field	**Type**	**Unique?**	**Reason**
number	Number		
street	Text		
colour	Text		
postcode	Text		
price	Number		

Suggest a compound key that could be used as the primary key of the House table.

Compound key:

Task 3

The Receipt table below is used to store details of all receipts printed in a supermarket.

Receipt				
customerID	**date**	**time**	**totalCost(£)**	**staffID**
1036271	11/08/15	10:53	23.94	205

Suggest a compound key that could be used as the primary key of the Receipt table.

Compound key:

Task 4

The Match table below is used to store details of scores for matches in an ice hockey league.

Match					
homeTeam	**awayTeam**	**date**	**homeScore**	**awayScore**	**refID**
Blue Bells	Wolverines	09/09/14	3	2	08

Suggest a compound key that could be used as the primary key of the Match table.

Compound key:

Task 5

The Phone table below is used to store details of mobile phones available for monthly contract customers.

Phone						
make	**model**	**colour**	**weight(g)**	**4G?**	**network**	**cost(£)**
Samsung	Galaxy S5	Blue	145	Yes	EE	26.99

Suggest a compound key that could be used as the primary key of the Match table.

Compound key:

Task 6

The Product table below is used to store details of items for sale at the UGame website.

Product						
make	**model**	**price (£)**	**colour**	**bluetooth?**	**NumberOfReviews**	**starRating**
SennHeiser	Game On	189.99	Black	No	23	4

Suggest a compound key that could be used as the primary key of the Product table.

Compound key:

Relationships

A relationship is a significant real world association between entities. It is a natural association between one or more entities. For example, Students learn Subjects and Teachers educate Students.

It describes how the two entities are related and can be thought of a connection between them.

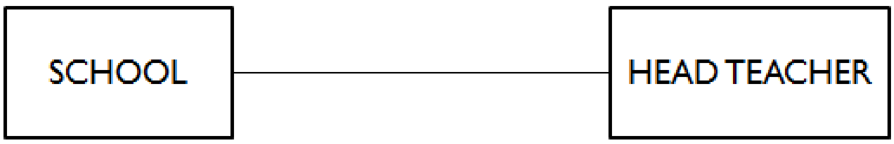
Cardinality

The cardinality of a relationship defines the number of participants in the relationship. It states the number of entity occurrences in one entity that are associated with one occurrence of the related entity. Cardinality can be:

- One-to-one
- One-to-many
- Many-to-many

One-to-one

In this type of relationship, every instance in one entity is linked with **exactly** one instance in another entity.

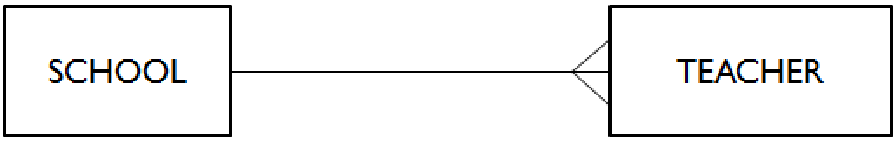


This entity relationship diagram shows a one-to-one relationship *(straight line).*

Each school has exactly one head teacher; each head teacher belongs to exactly one school.

One-to-many

In this type of relationship, <u>every</u> instance in one entity is linked with **several** instances in another entity.



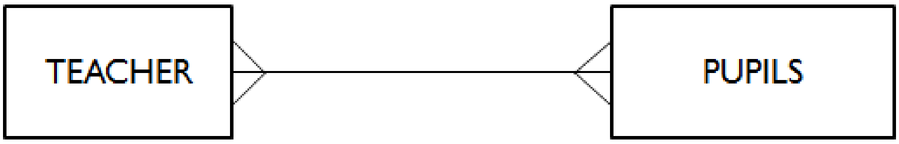
This entity relationship diagram shows a one-to-many relationship *(crow's feet at the many end).*

Each school can have several teachers but each teacher belongs to just one school.

Many-to-many

In this type of relationship, <u>every</u> instance in one entity is linked with **several** instances in another entity.

In turn, each instance in the second entity is linked to many instances in the first entity.



This entity relationship diagram shows a many-to-many relationship *(crows feet at both ends)*

Each pupil is taught by many teachers and each of those teachers will teach many pupils.

Practise Questions – Relationships/Cardinality

Task 1

SurfScotland is a blog used by members to share information about surfing in Scotland. A relational database is used to store details of members and blog posts in two related tables, Member and Post.

Member Table

MemberID	Lastname	Firstname	Email
0001	davies	jim	jimbo31@scotmail.co.uk
0002	mckay	ann	mckaya218@hotmail.com
0003	roberts	carol	croberts123@teachers.com
0004	singh	hardeep	singh832@scotmail.co.uk

Post Table

PostID	Title	Date	MemberID
0001	Welcome to the SurfScotland blog	01/082016	0001
0002	Belhaven Bay Dunbar	08/08/2016	0001
0003	Coldingham Bay Scottish Borders	13/08/2016	0001
0004	Hebridean Surf Lewis	15/08/2016	0002
0005	Broch Open Surf Competition	15/08/2016	0004

(a) State the cardinality that exists between the Member and Post entities.

(b) Describe the type of relationship that exists between the Member and Post entities.

Task 2

ScotBank uses a relational database to store information about customers and the different types of accounts that they have.

AccountType Table

AccountType	Account
01	current
02	savings
03	mortgage
04	loan

Customer Table

CustomerID	Lastname	Firstname	AccountType
0001	davies	jim	01
0001	davies	jim	02
0002	mckay	ann	01
0002	mckay	ann	03
0003	roberts	carol	02
0003	roberts	carol	03
0003	roberts	carol	04
0004	singh	hardeep	01
0004	singh	hardeep	02
0004	singh	hardeep	03

(a) State the cardinality that exists between the Customer and AccountType entities.

(b) Describe the type of relationship that exists between these entities.

Task 3

The RetroClothing website uses a relational database to store details of items of clothing for sale and the brand of each item.

Item Table

ItemID	Description	Size	Era	BrandID
0001	Red swim suit	10	1950s	003
0002	Floral dungarees playsuit	10	1990s	002
0003	Brown swing coat	16 - 18	1960s	005
0004	Circle skirt black white polka dot	12 - 14	1950s	004
0005	Floral print hostess dress	10	1970s	005

Brand Table

BrandID	Brand	Nationality
001	Valentino	Italian
002	Mary Quant	British
003	Rose Marie Reid	US
004	Kiki Byrne	Nowegian
005	Susan Small	British

Founder Table

BrandID	Firstname	Surname	DOB
001	Valentino	Garavani	11/05/1932
002	Mary	Quant	11/02/1934
003	Rose	Yancey	12/09/1906
004	Olaug	Grinaker	18/04/1937
005	Leslie	Jones	21/07/1904

(a) State the cardinality that exists between the Item and Brand entities.

(b) State the cardinality that exists between the Brand and Founded entities.

(c) Describe the type of relationship that exists between each pair of entities.

Task 4

GeoCity is a website that is used by primary children to learn about world geography. A relational database is used to store details of Continents, Countries and their Capital Cities in two related tables.

CapitalCity Table

CapitalID	Capital	Population
001	Sophia	1300000
002	Yaounde	750000
003	Reykjavik	84000
004	Riga	9000000
005	Saint Georges	30000

Continent Table

Continent	Area(km^2)	Population
Europe	10,180,000	741,447,158
Africa	30,370,000	1,225,080,510
America	42,549,000	1,001,559,000
Antarctica	14,000,000	1,106
Australia	8,600,000	35,000,000
Asia	44,579,000	4,462,676,731

Country Table

CountryCode	Country	Continent	CapitalID
BG	Bulgaria	Europe	001
CAM	Cameroon	Africa	002
IC	Iceland	Europe	003
LV	Latvia	Europe	004
WG	Grenada	Americas	005

(a) State the cardinality that exists between the Continent and Country entities.

(b) State the cardinality that exists between the Country and CapitalCity entities.

(c) Describe the type of relationship that exists between each pair of entities.

Data Modelling

A **model** is a representation of a real world object or system.

For example:

- a wooden model of a new ship
- a computer model of global warming
- a miniature model of a new bridge

Models are used to check or **test** a system.

Depending on the system, a model allows engineers to check that the system will work once it is implemented or allow scientists to understand a system before it develops further.

Models are quick and cheap to build and fixing errors at an early stage in a development saves a lot of money and resources in the long run.

When developers are creating a database system, **data models** are created. These are used to check the design of the system before it is implemented.

The models describe the structures and relationships needed and allow the developers to better understand the system.

Data modelling always includes –

- an **Entity Occurrence Diagram**
- an **Entity Relationship Diagram (ERD)**
- a **Data Dictionary (DD)**

Entity Occurrence Diagram

An entity-occurrence diagram illustrates the relationships between the entity occurrences of one entity, with the entity occurrences within a related entity. The creation of an entity-occurrence diagram helps to identify the cardinality of the relationship that exists between the two entities.

In an entity-occurrence diagram, each entity is shown as a tall oval. Inside each entity, each entity occurrence is represented by the value of its identifier and each relationship is shown by drawing a line between associated entity occurrences.

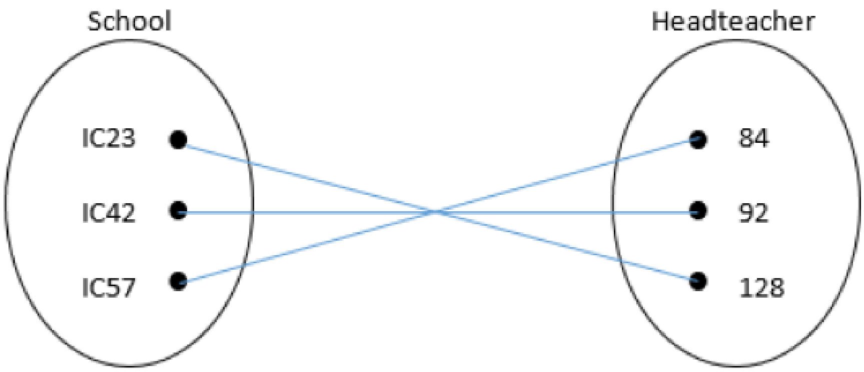
Examples of entity-occurrence diagrams are shown below.

Entity-occurrence diagram example 1: one-to-one relationship

The following table indicates which School is managed by which Head teacher and which Head teacher manages which School.

School	Headteacher
IC42	92
IC57	84
IC23	128

Here is the matching entity-occurrence diagram.



From this entity-occurrence diagram, we can see that each occurrence in the School entity has an association with one, and only one, entity occurrence in the Head teacher entity. Similarly, each occurrence in the Head teacher entity has an association with one, and only one, occurrence in the School entity.

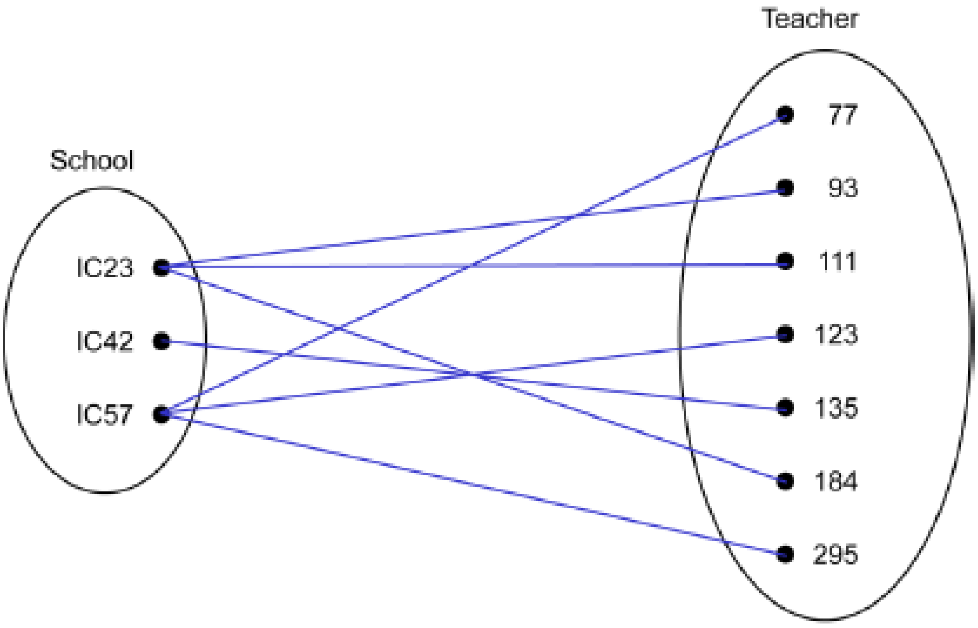
This confirms that there is a one-to-one relationship between the School and Head teacher entities.

Entity-occurrence diagram example 2: one-to-many relationship

The following table indicates which School employs which Teachers and which Teachers are employed by which School.

School	Teacher
IC42	135
IC57	123
IC23	111
IC23	184
IC57	77
IC57	295
IC23	93

Here is the matching entity-occurrence diagram.



From this entity-occurrence diagram, we can see that each occurrence in the School entity has an association with one or more entity occurrences in the Teacher entity. We can also see that each occurrence in the Teacher entity has an association with one, and only one, occurrence in the School entity.

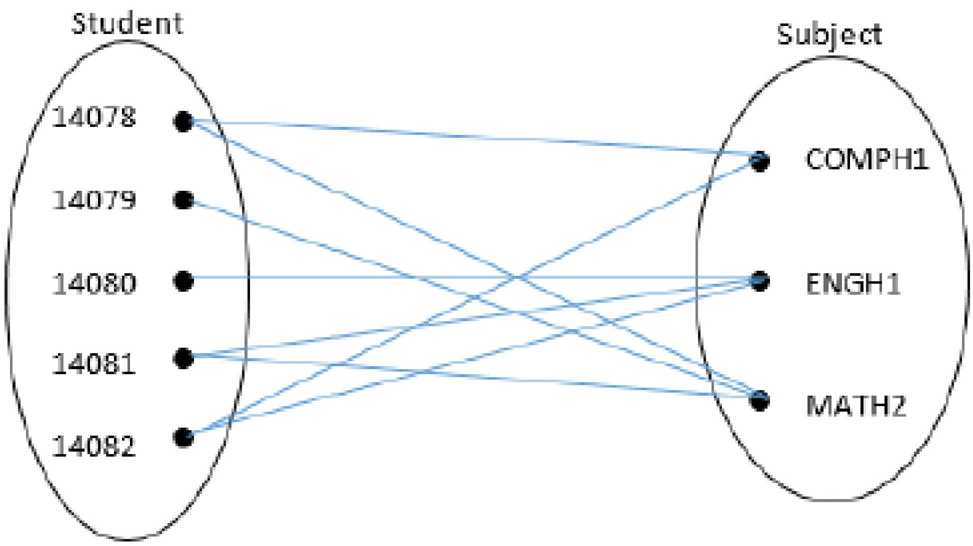
This confirms that there is a one-to-many relationship between the School and Teacher entities.

Entity-occurrence diagram example 3: many-to-many relationship

The following table indicates which Students study which Subjects and which Subjects are studied by which Students.

Student	Subject
14078	COMPH1
14079	MATH2
14080	ENGH1
14078	MATH2
14081	ENGH1
14082	ENGH1
14082	COMPH1
14081	MATH2

Here is the matching entity-occurrence diagram.

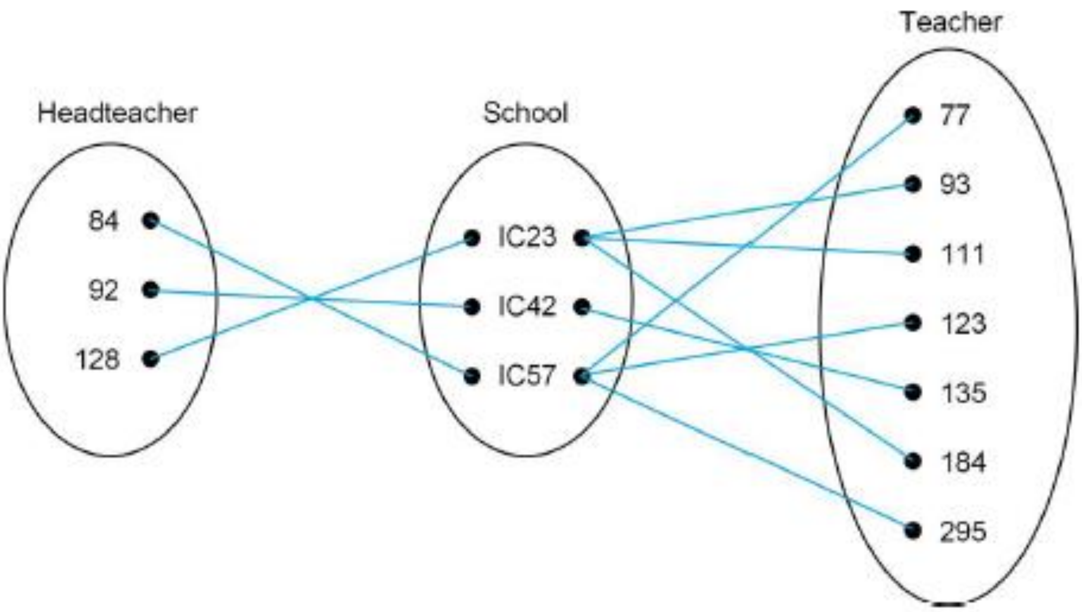


From this entity-occurrence diagram, we can see that each occurrence in the Student entity has an association with one or more entity occurrences in the Subject entity. We can also see that each occurrence in the Subject entity has an association with one or more occurrences in the Student entity.

This confirms that there is a many-to-many relationship between the Student and Subject entities.

Entity-occurrence diagram example 4: multiple entities

An entity-occurrence diagram may be used to illustrate the relationship between multiple entities as shown below:

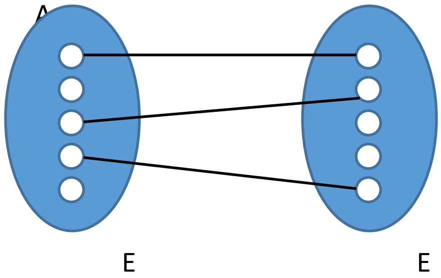


Practise Questions – Entity Occurrence Diagrams

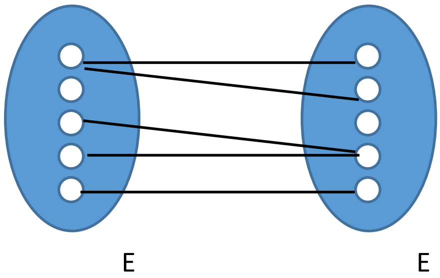
Task 1

Which entity occurrence diagram represents a one-to-one relationship?

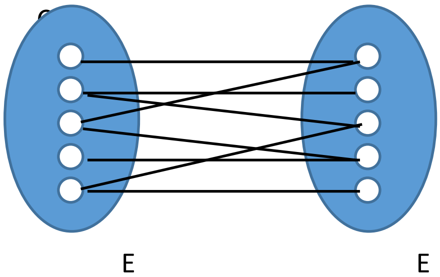
A



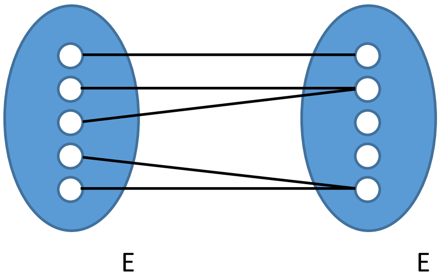
B



C



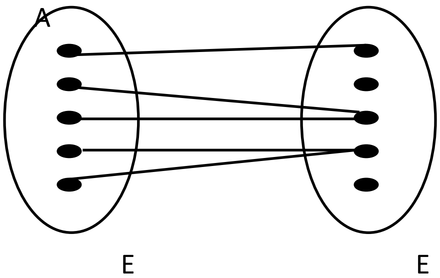
D



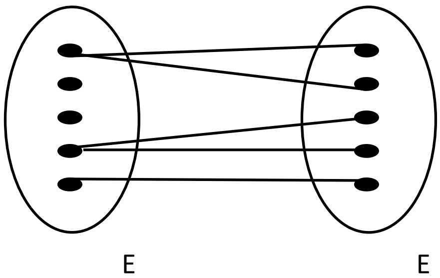
Task 2

Which entity occurrence diagram represents a many-to-many relationship?

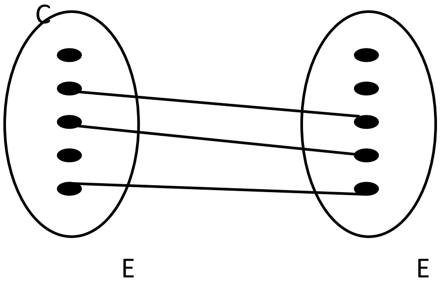
A



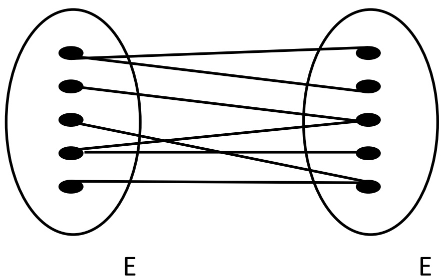
B



C

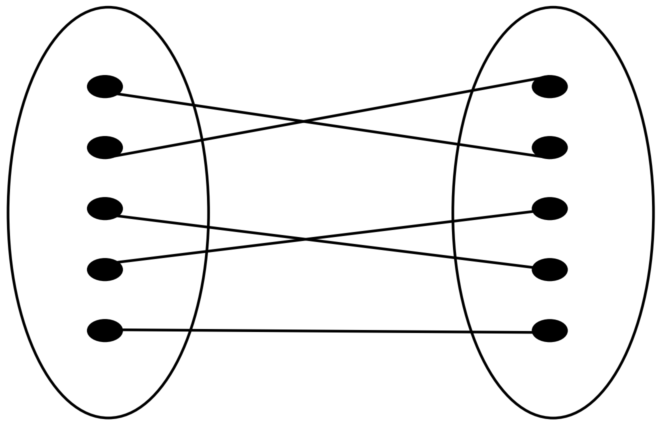


D



Task 3

What type of relationship is illustrated in the following entity occurrence diagram?



Entity 1

Entity 2

Answer:

Task 4

(a) Draw an entity occurrence diagram to represent the relationship between the Customer and Rental entities. Sample data stored in these entities has been provided below.

Customer			
CustCode	First	Name	Phone
1001	Anne	Jones	01111111111
1002	Sam	McKay	02222121212
1003	Jim	Shaw	01213132123

Rental		
ID	CustCode	Registration
1	1001	AB12 JKL
2	1003	BA32 MKL
3	1001	CD41 PLM
4	1002	AB22 MNB
5	1002	BA32 MKL
6	1001	AB22 MNB

(b) State the cardinality of the relationship between these two entities.

Task 5

(a) Draw an entity occurrence diagram to represent the relationship between the Rental and Vehicle entities. Sample data stored in the Vehicle entity has been provided below; sample data in the Rental entity was provided in Task 4 above.

Vehicle		
Registration	Make	Model
AB12 JKL	Ford	Taurus
AB22 MNB	Vauxhall	Corsa
BA32 MKL	Ford	Focus
CD41 PLM	Fiat	500L

(b) State the cardinality of the relationship between these two entities.

Entity Relationship Diagrams (ERD)

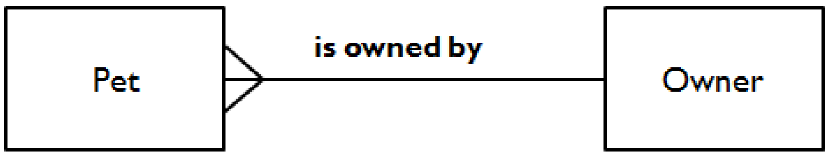
An entity-relationship diagram is a graphical representation of the entities in a system. It is used to summarise the relationship that exists between two or more entities. An entity-relationship diagram indicates:

- the name of each entity in the system
- the name of the relationship between two entities
- the cardinality of the relationship between two entities
- if required, the name of each attribute can be shown

Entities are shown as rectangles:

Relationships are shown as labelled lines:

The many end of a relationship is indicated using crow's feet.



This relationship states that each pet has one owner and each owner has many pets.

Creating ER Diagrams

When creating an entity relationship diagram, you should:

1. list each pair of entities and the relationship between them (start with the word 'each')
2. position the entities on the page
3. draw the relationships between the entities
4. resolve any many-to-many relationships

Example 1

An airline can fly many flights, but each flight is flown by only one airline.

What would the entity relationship diagram look like for this example?

The entities and relationships mentioned are:

- Each airline field many flights
- Each flight is flown by one airline

Example 2

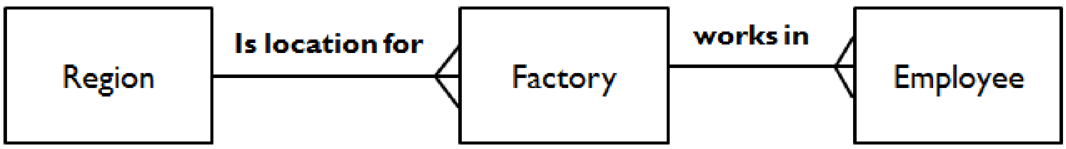
The MakeIt Corporation operates many factories and each of those factories is located in a particular region.

Each region is the location of many of the factories. Each factory employs many employees, but each of these employees is employed by only one of the factories.

What would the entity relationship diagram look like for this example?

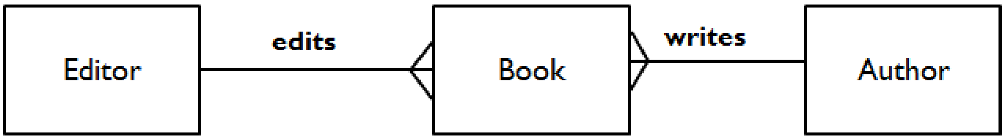
The entities and relationships mentioned are:

- Each factory is location in one region
- Each region is the location for many factories
- Each factory employs many employees
- Each employee works in one factory



Reading ER diagrams

Example 1

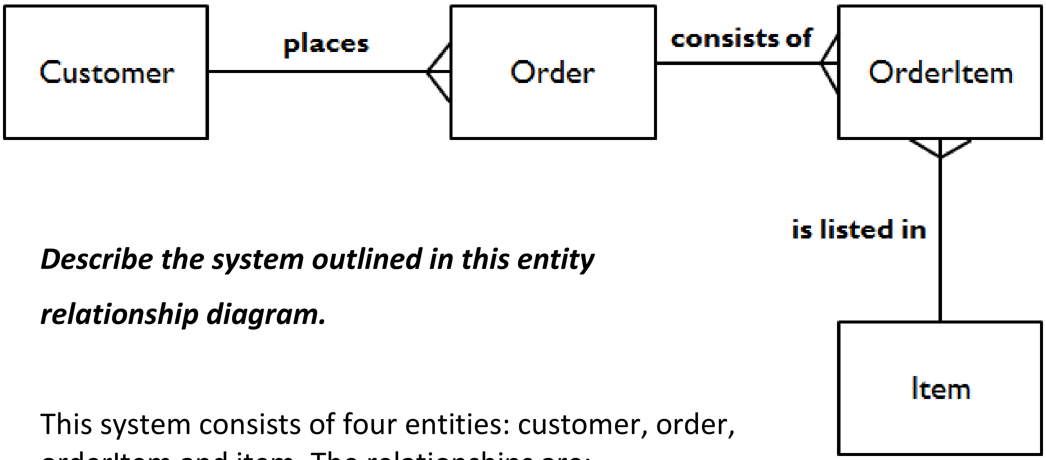


Describe the system outlined in the entity relationship diagram above

The system consists of three entities: editor, book and author. The relationships are:

- Each editor edits many books
- Each book has one editor
- Each book has one author
- Each author writes many books

Example 2



Describe the system outlined in this entity relationship diagram.

This system consists of four entities: customer, order, orderItem and item. The relationships are:

- Each customer can place many orders
- Each order is placed by one customer
- Each order consists of many order items
- Each order item is associated with one particular order
- Each order item refers to one specific item
- Each item can appear in many order items

Resolving Many-To-Many Relationships

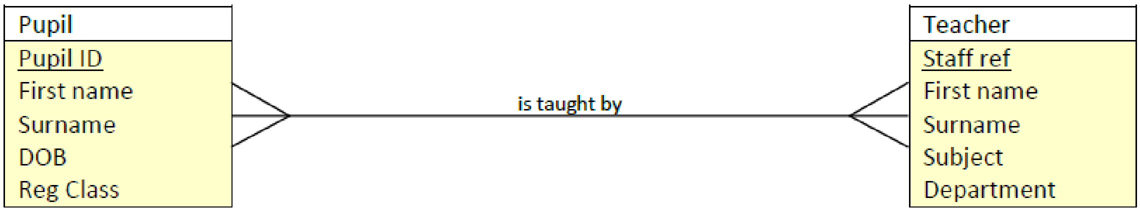
A **many-to-many** relationship **cannot** be implemented in a database system.

Instead, a third entity must be introduced.

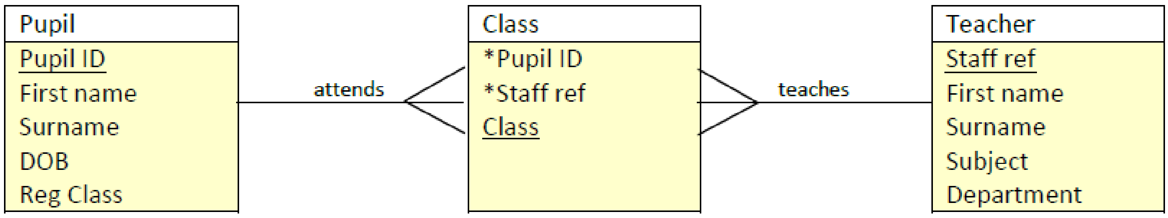
This new entity replaces the many-to-many relationship with **two** separate **one-to-many** relationships.

The new entity includes a **foreign key** from **each** of the other entities.

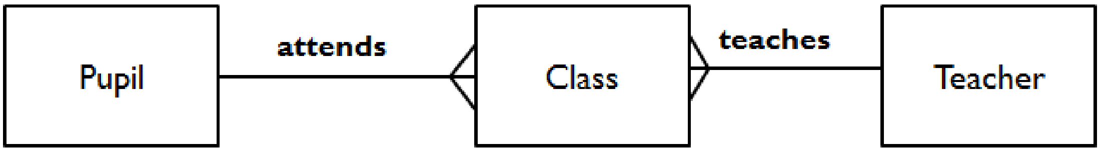
The relationship between Pupil and Teacher is a many-to-many relationship**.**



This many-to-many relationship can be resolved by introducing a new entity:



We can illustrate the relationships between the entities Pupil and Teacher in an Entity Relationship Diagram:



Practise Questions – Entity Relationship Diagrams

1. Draw an entity relationship diagram to model the following set of conditions relating to Students, Courses and Lecturers.
   - Each student is on only one course
   - Each course must have one or more students
   - Each course is taught by one or more lecturers
   - Each lecturer teaches on one or more courses

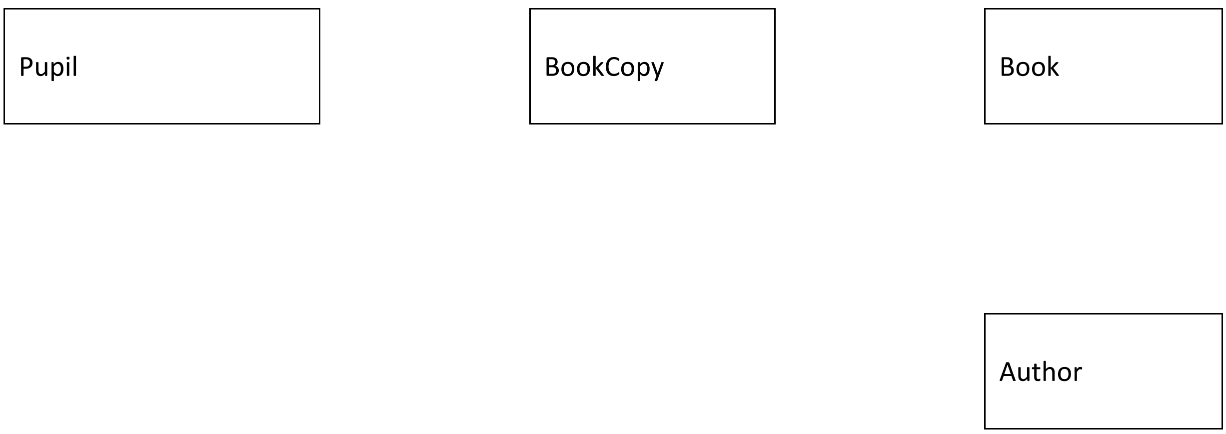
   Your entity relationship diagram should indicate:
   - the name of each relationship
   - the cardinality of each relationship

2. A relational database is used to store details in a school library. The following design rules are applied:

- Pupils can take out up to six books at a time
- There may be more than one copy of a particular book
- Books are usually written by only one author but sometimes that are written jointly by more than one author

Complete the entity relationship diagram below to model the library system. You should indicate:

- the cardinality of the relationship between each pair of entities
- the name of each relationship in the diagram

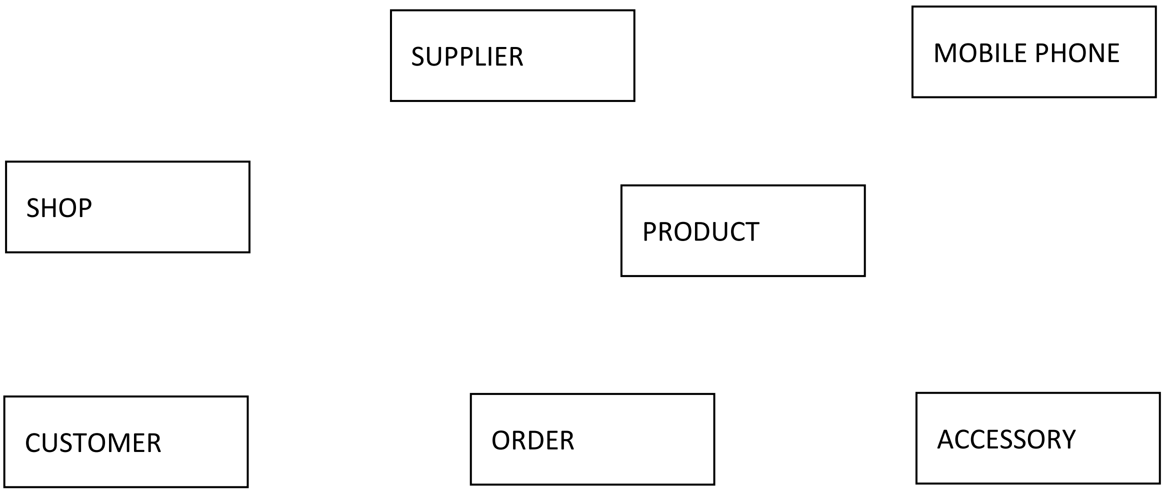


3. A chain of shops sells a range of mobile phones and accessories sourced from a variety of mobile phone companies. Customers place orders for phones and accessories with a shop which then supplies the customer from stock or orders the items from one if its suppliers.

After analysing the system, the following relationships have been identified:

- The organisation has many SHOPs in different towns and cities
- Each SHOP sells many PRODUCTs
- Each PRODUCT can be sold in many SHOPs
- PRODUCTs can be either a PHONE or an ACCESSORY
- PRODUCTs can be sourced from many SUPPLIERs
- SUPPLIERs supply many PRODUCTs
- CUSTOMERs place many different ORDERs over time
- Each ORDER can only come from one CUSTOMER
- One ORDER may be for many PRODUCTS
- Each PRODUCT may appear on different ORDERs

Complete the entity relationship diagram to model these entities. You should indicate the name and cardinality of the relationship between each pair of entities.



4. A company is made up of departments. Each employee works in just one of those departments and each department has many employees working in it. A department can be working on many different projects at any time. A project might need many different departments to work on it. Each project requires many resources to complete it, such as computer time or a specialist contractor's time. Each resource might be required on many different projects.

   (a) Use this description to identify the entities and relationships in the system. The first few have been started for you.

   Each EMPLOYEE works in only one DEPARTMENT

   Each DEPARTMENT employs many EMPLOYEEs

   (b) Draw an entity relationship diagram for the business. Remember to indicate the name and cardinality of the relationship between each pair of entities.

Data Dictionary

A **data dictionary** is used to record details about the database. It provides a description of the constraints or rules that apply to each of the attributes of each entity in the system.

A data dictionary is simply a large table that stores **metadata** – in other words, it stores data about data.

The structures needed to store data in the database are **planned** using a data dictionary and an **entity relationship diagram**.

Entity	Attribute	PK/ FK	Data Type/ Size	Unique	Required	Validation	Format
Customer	CustomerID	PK	Number	Y	Y	>=1 and <=9999	0000
	Firstname		Text (15)	N	Y		
	Surname		Text (20)	N	Y		
	Street		Text (30)	N	Y		
	Town		Text (20)	N	Y		
	Postcode		Text (8)	N	Y		
	Ski Level		Text (12)	N	Y	Choose one of Beginner, Intermediate or Expert)	
Resort	Resort	PK	Text (20)	Y	Y		
	Resort Manager		Text (35)	N	Y		
	Resort Address		Text (50)	Y	Y		
	Spa		Boolean	N	Y		
	Crèche		Boolean	N	Y		

- Each row provides details about **one attribute** in the system
- Each column **specifies a rule** or restriction that applies to the attributes.

Each row of the data dictionary is completed by stating the appropriate value or rule for each column

Where no rules apply, the column value is left empty.

- **PK/FK** (Is the field a primary key / foreign key – a field can be both)
- **Data Types** (Text, Number, Boolean, Date/time)
- **Size** (Maximum number of characters /digits)
- **Required** (Mandatory or Optional. Primary key must be required)
- **Validation** (Presence/Length/Range checks, restricted choice)

Entity	Attribute	PK/FK	Data Type	Required	Required	Validation

Practise Questions – Data Dictionary

Task 1

A publishing company uses a relational database to store details about books and customer orders in four separate entities. Details of the entities and attributes used are shown below (primary keys have been underlined and foreign keys are indicated using an asterisk *).

Customer Entity	Order Entity	Book Entity	Author Entity
<u>Customer Number</u> Customer Name Customer Address Customer Town Customer Postcode	<u>Order Reference</u> Order Date ISBN * Quantity Customer Number *	<u>ISBN</u> Book Title Category Price Author Code *	<u>Author Code</u> Author Name Author Address Author Town Author Postcode

A sample customer order and sample book details are shown below.

Customer Order: **HYR847GB**

Customer Number:	782
Customer Name:	Inverclyde Books
Customer Address:	52 High Street
Customer Town:	Gourock
Customer Postcode:	PA19 1UX

Order Date	ISBN	Quantity
12/03/2014	0901714564X	9
16/03/2013	7289192000S	15
27/03/2013	0901714564X	20
04/04/2013	3781972928N	12
11/04/2013	1217292921B	9

Note: Category can be …

- Children
- Crime
- Historical
- Large Print
- Non-Fiction
- Romance

Book

ISBN:	0901714564X
Book Title:	Weather Time
Category:	Non-Fiction
Price:	£15.99
Author Code:	87281
Author Name:	Julie Adams
Address:	15 West Street
Town:	Greenock
Postcode:	PA19 7XE

- Quantity must be at least 1 but can no more than 24
- The cheapest book available costs £0.99 and the dearest costs £38.95

Create a data dictionary to record details of each of the entities in this relational database system.

Attribute Name	**Key**	**Type**	**Size**	**Reqd**	**Validation**
Entity:					
Entity:					
Entity:					
Entity:					

Task 2

Now Rentals DVD club uses a relational database to store details of members, DVDs and rentals. A sample report produced by the system is shown below.

MEMBER RECORD

Member Number: 241

Surname: Smith
Address: 23 Jones Road
Town Walforth
Phone: 07771234567
Joined: 12/01/2010

Priority Code:

DVD ID	**DVD Title**	**Date Due Back**	**Late?**	**Certificate**	**Certificate Description**
3236	Wheelies	11/05/14	True	U	Universal
6512	Harrytron	12/06/14	False	PG	Parental guidance
4419	The Brainbox	30/07/14	False	PG	Parental guidance

The database stores the rental details in four entities: MEMBER, RENTAL, DVD and CERTIFICATE.

The attributes in each of those entities are listed below (primary keys are underlined and foreign keys are indicated using an asterisk *).

MEMBER	RENTAL	DVD	CERTIFICATE
<u>Member ID</u> Surname Address Town Phone Number Date Joined Priority Code	<u>Rental Number</u> Member ID * DVD ID * Date Due Back Late?	<u>DVD ID</u> DVD Title Certificate *	<u>Certificate</u> Description

Create a data dictionary to record details of each of the entities in this relational database system.

Certificate can be …

- U (Universal)
- PG (Parental Guidance)
- 12 (suitable for 12 years and over)
- 15 (Suitable only for 15 years and over)
- 18 (Suitable only for adults)

- Members can opt to pay an additional fee to become a Priority member
- All members who take this option are allocated a 4-digit Priority Code between 1111 and 9999

Attribute Name	Key	Type	Size	Reqd	Validation
Entity:					
Entity:					
Entity:					
Entity:					

Query Design

The design of the SQL query should indicate:

- any field(s)/attributes or computated values required
- the table(s)/entities needed to provide all of the details required
- any search criteria to be applied
- what grouping is needed (if appropriate)
- the field(s) used to sort the data and the type(s) of sort required

Planning the design of a query before creating the SQL code is good practice.

This gives the developer time to think carefully about the fields that are required, which in turns, helps them to identify the table or tables that will be needed.

It also allows developers to consider the purpose of the query (search and/or sort), together with any required search criteria and/or sort order.

Planning ahead helps to reduce the errors that developers may otherwise encounter when working with the SQL code.

Example

A travel agency uses a relational database to enable their employees to view details of hotels in Scottish holiday resorts and make bookings for customers. The details are stored in four separate tables called Hotel, Resort, Booking and Customer.

The structure of these tables is shown below:

Hotel

Field Name
hotelRef
hotelName
resortID
starRating
seasonStartDate
swimmingPool
mealPlan
checkInTime
pricePersonNight

Resort

Field Name
resortID
resortName
resortType
trainStation

Booking

Field Name
bookingNo
customer#
hotelRef
startDate
numberNights
numberInParty

Customer

Field Name
customer#
firstname
surname
address
town
postcode

Example 1: design a query to display the name, swimming pool details, resort and resort type of any hotel in a coastal resort that starts with the letter 'A'.

Field(s) and calculation(s)	hotelName, swimmingPool, resortName, resortType
Table(s)	Hotel, Resort
Search criteria	resortType = "coastal" and resortName like "A*"
Grouping	
Sort order	

Example 2: design a query to display a customer's full name, booking number, start date, hotel name and resort name for all customers who have an 'h' as the second letter of their surname. List these details in alphabetical order of surname; listing customers with the same surname in order of the earliest holiday first.

Field(s) and calculation(s)	firstname, surname, bookingNo, startDate, hotelName, resortName
Table(s)	Customer, Booking, Hotel, Resort
Search criteria	surname LIKE "?h*"
Grouping	
Sort order	surname ASC, startDate ASC

Example 3: design a query that uses a readable heading to display the cheapest and dearest price per night.

Field(s) and calculation(s)	Dearest price per night = MAX(pricePersonNight), Cheapest price per night = MIN(pricePersonNight)
Table(s)	Hotel
Search criteria	
Grouping	
Sort order	

Example 4: design a query to display the average number of nights booked.

Field(s) and calculation(s)	AVG(numberNights)
Table(s)	Booking
Search criteria	
Grouping	
Sort order	

Example 5: design a query to display a list of the different types of resort, together with the number of resorts in each of those categories.

Field(s) and calculation(s)	resortType, COUNT(*)
Table(s)	Resort
Search criteria	
Grouping	resortType
Sort order	

Example 6: design a query to display the number of bookings for hotels in coastal resorts. Show the resort type and use a readable heading for the results returned by the aggregate function.

Field(s) and calculation(s)	resortType, Number of Hotels = COUNT(*)
Table(s)	Resort, Hotel, Booking
Search criteria	resortType = "coastal"
Grouping	resortType
Sort order	

Example 7: design a query to display a list of each type of meal plan, together with the number of bookings made for each of those meal plans. List the details from the least popular meal plan to the most popular.

Field(s) and calculation(s)	mealPlan, COUNT(*)
Table(s)	Hotel, Booking
Search criteria	
Grouping	mealPlan
Sort order	COUNT(*) ASC

Example 8: design a query that uses a readable heading to display the total number of people booked into a hotel in July.

Field(s) and calculation(s)	People booked in July = SUM(numberInParty)
Table(s)	Booking
Search criteria	startDate LIKE "*/07/*"
Grouping	
Sort order	

Example 9: design a SELECT query to display the hotel name and the improved rating, if all hotels in Ayr gain an extra star (use a readable heading to display the improved ratings).

Field(s) and calculation(s)	hotelName, Improved rating = starRating + 1
Table(s)	Hotel, Resort
Search criteria	resortName = “Ayr”
Grouping	
Sort order	

Example 10: design a query to display the surname, booking number, number of nights, number in party, price per night and the total cost of each booking (with a readable column heading). Display the dearest booking first.

Field(s) and calculation(s)	surname, bookingNo, numberNights, numberInParty, pricePersonNight, Total Cost = (numberNights * numberInParty * pricePersonNight)
Table(s)	Customer, Booking, Hotel
Search criteria	
Grouping	
Sort order	numberNights * numberInParty * pricePersonNight DESC

Practise Questions – Query Design

Task 1

The `Countries` database stores details of countries and cities in two separate tables called `Country` and `City`. The structure of the tables is shown below.

Country
Field Name
countryName
countryCode
capital
area
totalPopulation

City
Field Name
cityID
cityName
countryCode
population
longitude
latitude

Design SELECT queries to perform each of the following tasks. Each design should indicate:

- any field(s) or computed values required
- the table(s) or query(queries) needed to provide the details required
- any search criteria to be applied
- what grouping is needed (if appropriate)
- the field(s) used to sort the data and the type(s) of sort required

1. Search the database to display the name, capital city and total population of the country with the largest total population.

Query1(1)

Field(s) and calculation(s)	
Table(s) and query(queries)	
Search criteria	
Grouping	
Sort order	

Query1(2)

Field(s) and calculation(s)	
Table(s) and query(queries)	
Search criteria	
Grouping	
Sort order	

2. Search the database to display the name, country and population of any city that has a population which is at least 5,000,000 more than the average population of all the cities.

Query2(1)

Field(s) and calculation(s)	
Table(s) and query(queries)	
Search criteria	
Grouping	
Sort order	

Query2(2)

Field(s) and calculation(s)	
Table(s) and query(queries)	
Search criteria	
Grouping	
Sort order	

3. Search the database to display the name of any city that is further north than Reykjavik (the latitude of these cities is greater than the latitude of Reykjavik). The query should show the name of each relevant city, its latitude and country name. The city that is furthest north should be listed first.

Query3(1)

Field(s) and calculation(s)	
Table(s) and query(queries)	
Search criteria	
Grouping	
Sort order	

Query3(2)

Field(s) and calculation(s)	
Table(s) and query(queries)	
Search criteria	
Grouping	
Sort order	

4. Search the database to display the name and population of any city in the United Kingdom that has a population which is more than the average population of all the cities in Bolivia. Arrange these details from smallest to largest population.

Query4(1)

Field(s) and calculation(s)	
Table(s) and query(queries)	
Search criteria	
Grouping	
Sort order	

Query4(2)

Field(s) and calculation(s)	
Table(s) and query(queries)	
Search criteria	
Grouping	
Sort order	

5. Search the database to display the total number of countries with an area less than 1% of that of the country with the largest area.

Query5(1)

Field(s) and calculation(s)	
Table(s) and query(queries)	
Search criteria	
Grouping	
Sort order	

Query5(2)

Field(s) and calculation(s)	
Table(s) and query(queries)	
Search criteria	
Grouping	
Sort order	

Task 2

The `SimpleBreaks` database stores details of countries and cities in two separate tables called `Holiday` and `Hotel`. The structure of the tables is shown below.

Holiday

Field Name
Title
Destination
Country
dateOfDeparture
Nights
hotelRef

Hotel

Field Name
hotelRef
hotelName
City
starRating
pricePerNight
kilometresFromAirport

Design SELECT queries to perform each of the following tasks. Each design should indicate:

- any field(s) or computed values required
- the table(s) or query(queries) needed to provide the details required
- any search criteria to be applied
- what grouping is needed (if appropriate)
- the field(s) used to sort the data and the type(s) of sort required

1. Search the database to display the name, destination, country and distance from the airport of the hotel that is furthest from the airport.

Query1(1)

Field(s) and calculation(s)	
Table(s) and query(queries)	
Search criteria	
Grouping	
Sort order	

Query1(2)

Field(s) and calculation(s)	
Table(s) and query(queries)	
Search criteria	
Grouping	
Sort order	

2. Search the database to display the name and star rating of any hotel with a rating that is poorer that the average star rating of all the holidays that have the word ‘Break’ or ‘Package’ in their title. The hotel with the highest star rating should be listed first; hotels with the same star rating should be listed in alphabetical order of hotel name.

Query2(1)

Field(s) and calculation(s)	
Table(s) and query(queries)	
Search criteria	
Grouping	
Sort order	

Query2(2)

Field(s) and calculation(s)	
Table(s) and query(queries)	
Search criteria	
Grouping	
Sort order	

3. Search the database to display the name, city and price per night of any hotel which is dearer that the dearest hotel in Edinburgh. List the hotel details with the dearest hotel first; hotels with the same price should be listed in alphabetical order of city.

Query3(1)

Field(s) and calculation(s)	
Table(s) and query(queries)	
Search criteria	
Grouping	
Sort order	

Query3(2)

Field(s) and calculation(s)	
Table(s) and query(queries)	
Search criteria	
Grouping	
Sort order	

4. Search the database to display the number of holidays that have the same star rating as that of the 'Der Wald' hotel.

Query4(1)

Field(s) and calculation(s)	
Table(s) and query(queries)	
Search criteria	
Grouping	
Sort order	

Query4(2)

Field(s) and calculation(s)	
Table(s) and query(queries)	
Search criteria	
Grouping	
Sort order	

5. Search the database to display the title, city and distance from the airport of any holiday to Lisbon that is closer to the airport than the average distance from the airport of all the hotels in Spain.

Query5(1)

Field(s) and calculation(s)	
Table(s) and query(queries)	
Search criteria	
Grouping	
Sort order	

Query5(2)

Field(s) and calculation(s)	
Table(s) and query(queries)	
Search criteria	
Grouping	
Sort order	

6. Search the database to display the title, departure date and duration of any holiday that has the same duration as the longest holiday to a city with the letter 'o' as the second character of the city name. Arrange these details so that the most recent holiday is listed first.

Query6(1)

Field(s) and calculation(s)	
Table(s) and query(queries)	
Search criteria	
Grouping	
Sort order	

Query6(2)

Field(s) and calculation(s)	
Table(s) and query(queries)	
Search criteria	
Grouping	
Sort order	