Advanced Higher
Computing Science

Database Development

Database Structure

Name:_____

DATABASE DESIGN

DATA DICTIONARY

A data dictionary is a document that describes the structure of the database.

Entity Name	Attribute Name	PK/FK	Data Type / Size	Unique	Required	Validation
Department	deptID	PK	VARCHAR(5)	Y	Y	Length =5
	deptName		VARCHAR(30)	Y	Y	
	location		VARCHAR(5)	N	Y	Restricted Choice: North, South, East, West
Employee	employeeNumber	PK	INTEGER	Y	Y	
	firstName		VARCHAR(20)	N	Y	
	surname		VARCHAR(20)	N	Y	
	salary		FLOAT	N	Y	Range: 0.00 to 100,000.00
	startDate		DATE	N	Y	
deptEmployees	deptID	PK/FK	VARCHAR(5)			
	employeeNumber	PK/FK	INTEGER			

The data dictionary provides information on the following for the entire database:

Entity Name	The name of the table
Attribute Name	The name of the field
PK/FK	PK for a primary key and FK for a foreign key.
Data Type/Size	The type of data that the field can contain. For text fields, the number of characters is also specified.
Unique:	Specifies whether more than one record can have the same value.
Required:	Specifies whether a field must be filled in – presence check.
Validation:	Details any validation constraints to be checked.

Data Types

Attributes (fields) must be allocated appropriate data types and sizes.

Listed below are the main data types recognised by MySQL.

VARCHAR(n)	Character string. Variable length. Maximum length n
INT(size)	-2147483648 to 2147483647 normal. 0 to 4294967295 UNSIGNED*. The maximum number of digits may be specified in parenthesis
FLOAT(size,d)	A small number with a floating decimal point. The maximum number of digits may be specified in the size parameter. The maximum number of digits to the right of the decimal point is specified in the d parameter
DATE()	A date. Format: YYYY-MM-DD
TIME()	A time. Format: HH:MI:SS
YEAR()	A year in two-digit or four-digit format.
DATETIME()	*A date and time combination. Format: YYYY-MM-DD HH:MI:SS
CHARACTER(n)	Character string. Fixed-length n

The greyed out data types at the bottom of the list are not part of the Advanced Higher course but they may be useful to know about.

The full list of data types available can be found by visiting:

http://www.w3schools.com/sql/sql_datatypes.asp

ENTITY OCCURRENCE DIAGRAMS

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.

The **cardinality** of entity relationships can be identified as being:

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

They also help to identify the participation of an entity within a relationship.

Relationship Participation

In an entity relationship, the participation of an entity can be either **mandatory** or **optional**.

Mandatory
In a mandatory relationship, every instance of one entity must participate in a relationship with another entity.

Optional
In an optional relationship, any instance of one entity might participate in a relationship with another entity, but this is not compulsory.

This means that for any given relationship, there are four possible combinations of participation:

- Mandatory – Mandatory
- Optional – Mandatory
- Mandatory – Optional
- Optional – Optional

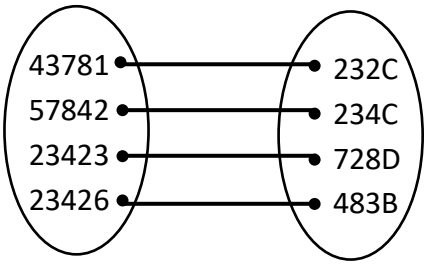
One to one – relationship participation

A one-to-one relationship specifies that for one entity there is only one other corresponding entity

The examples below show the options for allocating lockers to pupils in a school.

Mandatory – Mandatory

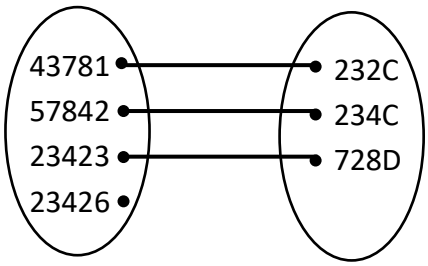
Pupil Number	Locker Number
43781	232C
57842	234C
23423	728D
23426	483B



Each pupil **must** have a locker and each locker **must** be allocated to a pupil.

Optional – Mandatory

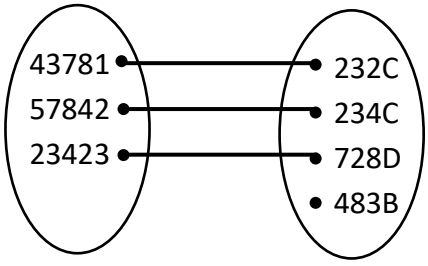
Pupil Number	Locker Number
43781	232C
57842	234C
23423	728D
23426	



Each pupil **may** have a locker but each locker **must** be allocated to a pupil.

Mandatory – Optional

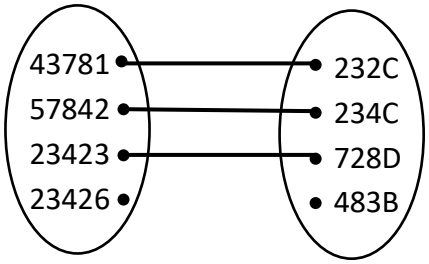
Pupil Number	Locker Number
43781	232C
57842	234C
23423	728D
	483B



Each pupil **must** have a locker but each locker **may** be allocated to a pupil.

Optional – Optional

Pupil Number	Locker Number
43781	232C
57842	234C
23423	728D
23426	
	483B



Each pupil **may** have a locker and each locker **may** be allocated to a pupil.

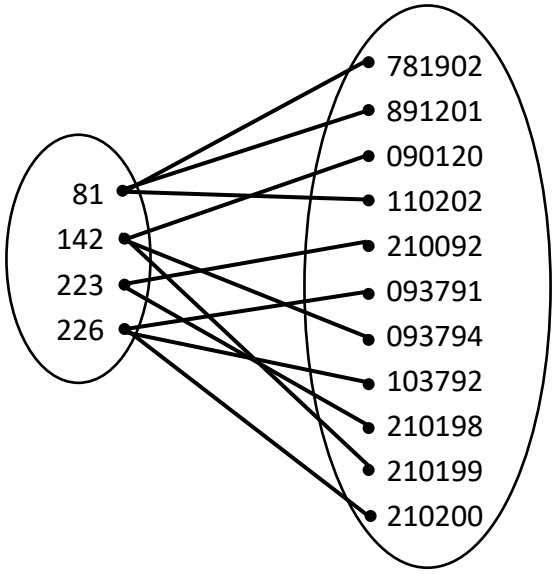
One to many – relationship participation

A one-to-many relationship specifies that for one entity there can be many corresponding entities.

The examples below show the options for allocating patients to doctors in a GP surgery.

Mandatory – Mandatory

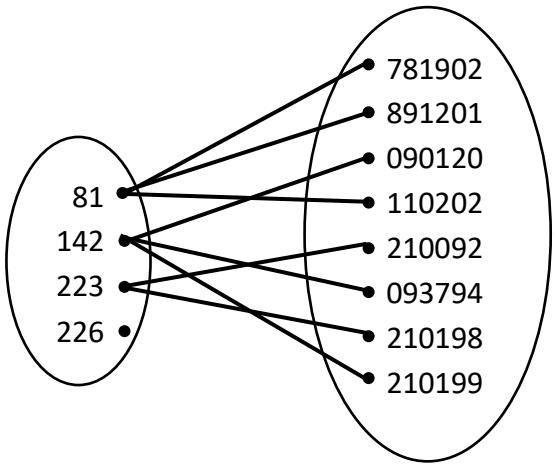
Doctor Number	Patient Number
81	781902
81	891201
142	090120
81	110202
223	210092
226	093791
142	093794
226	103792
223	210198
142	210199
226	210200



Each doctor **must** have at least one patient and each patient **must** be allocated to a doctor.

Optional – Mandatory

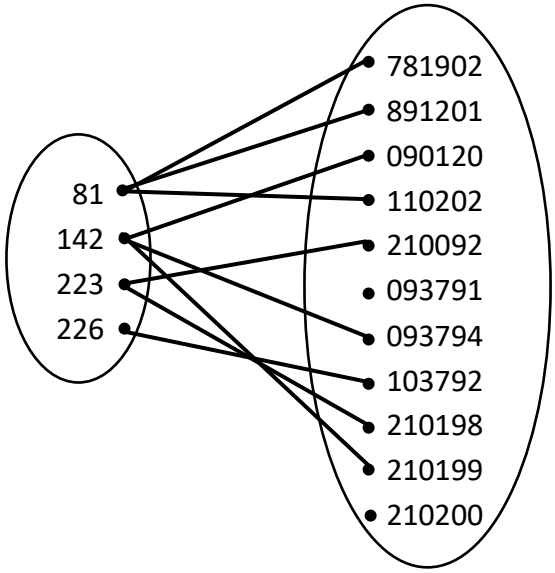
Doctor Number	Patient Number
81	781902
81	891201
142	090120
81	110202
223	210092
142	093794
223	210198
142	210199
226	



Each doctor **may** have patients but each patient **must** be allocated to a doctor.

Mandatory – Optional

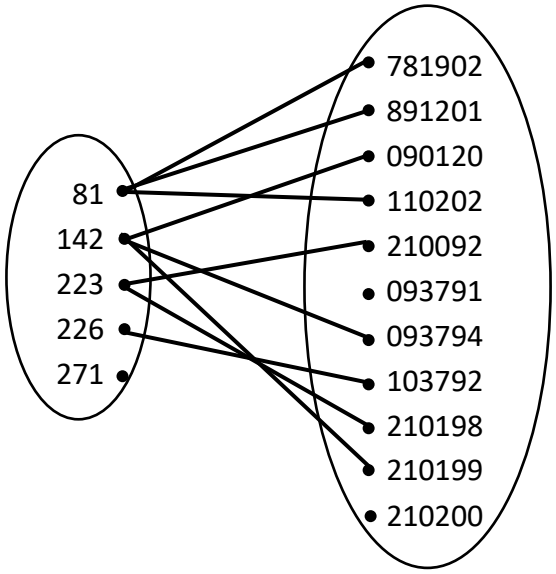
Doctor Number	Patient Number
81	781902
81	891201
142	090120
81	110202
223	210092
	093791
142	093794
226	103792
223	210198
142	210199
	210200



Each doctor **must** have at least one patient but a patient **may** be allocated to a doctor.

Optional – Optional

Doctor Number	Patient Number
81	781902
81	891201
142	090120
81	110202
223	210092
	093791
142	093794
226	103792
223	210198
142	210199
	210200
271	



Each doctor **may** have patients and each patient **may** be allocated to a doctor.

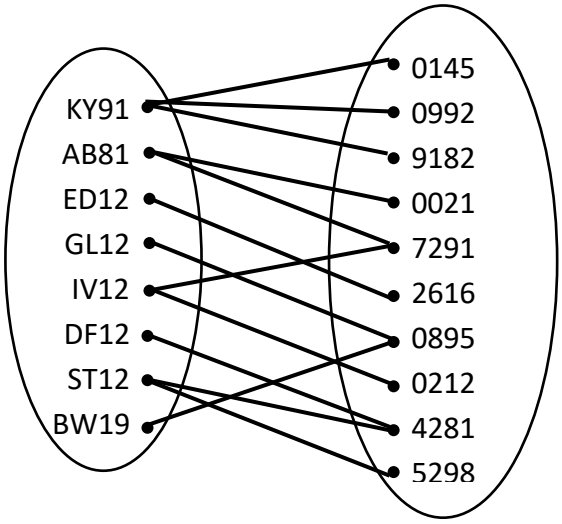
Many to many – relationship participation

A many-to-many relationship specifies that for one entity there can be many corresponding entities and vice versa.

The examples below show the options for netball players to netball teams.

Mandatory – Mandatory

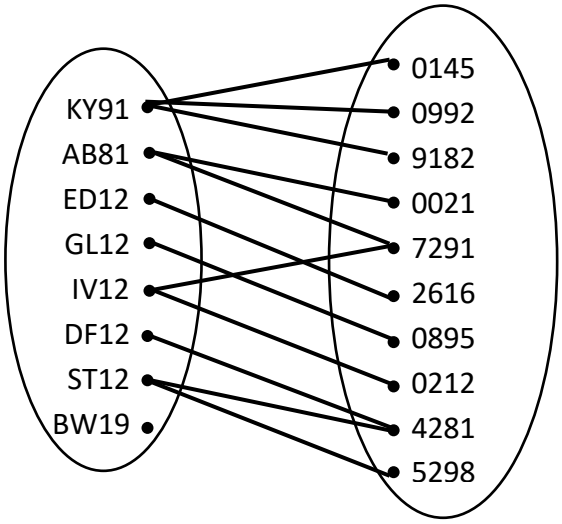
Team ID	Player ID
KY91	0145
KY91	0992
KY91	9182
AB81	0021
AB81	7291
ED12	2616
GL12	0895
IV12	0212
IV12	7291
DF12	4281
ST12	4281
ST12	5298
BW19	0895



Each team **must** have at least one player and each player **must** have at least one team. Teams can have many players and player can play for more than one team.

Optional – Mandatory

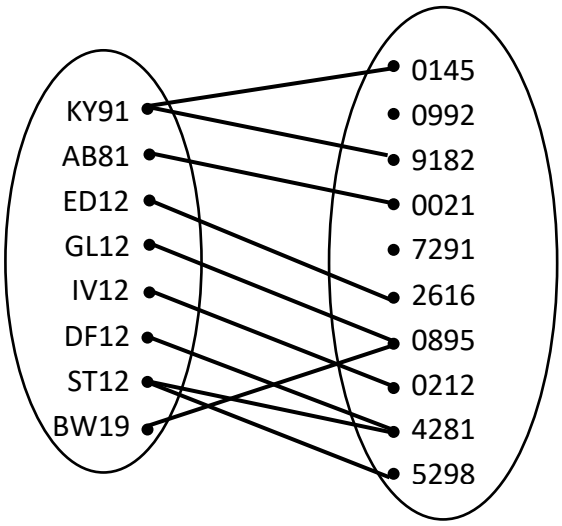
Team ID	Player ID
KY91	0145
KY91	0992
KY91	9182
AB81	0021
AB81	7291
ED12	2616
GL12	0895
IV12	0212
IV12	7291
DF12	4281
ST12	4281
ST12	5298
BW19	



Each team **may** have players allocated but each player **must** belong to at least one team. Teams can have many players and player can play for more than one team.

Mandatory – Optional

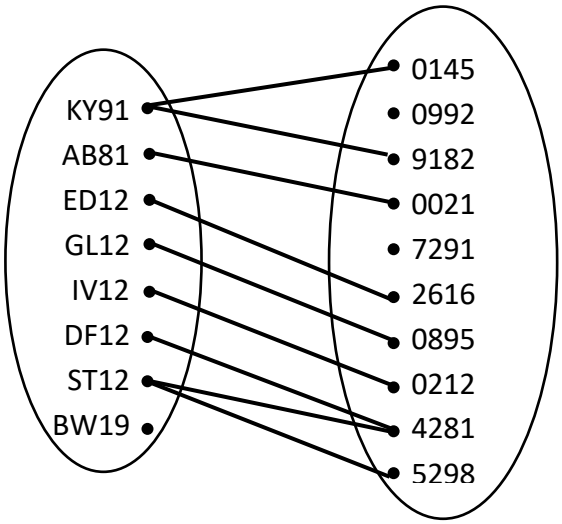
Team ID	Player ID
KY91	0145
	0992
KY91	9182
AB81	0021
	7291
ED12	2616
GL12	0895
IV12	0212
DF12	4281
ST12	4281
ST12	5298
BW19	0895



Each team **must** have at least one player but each player **may** belong to a team. Teams can have many players and player can play for more than one team.

Optional – Optional

Team ID	Player ID
KY91	0145
	0992
KY91	9182
AB81	0021
	7291
ED12	2616
GL12	0895
IV12	0212
DF12	4281
ST12	4281
ST12	5298
BW19	



Each team **may** have players allocated and each player **may** belong to a team. Teams can have many players and player can play for more than one team.

ENTITY RELATIONSHIP DIAGRAMS

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 type of entity in a relationship (strong/weak)
- the name of the relationship between two entities (label)
- the cardinality of the relationship between two entities (1-1; 1-M; M:N)
- the participation of the relationships between two entities (mandatory/optional)
- if required, the name of each attribute can be shown

Entity Type

An entity can be either strong or weak depending on the attribute it uses as its primary key.

Weak entity

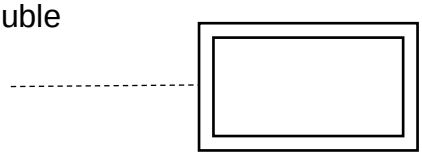
A weak entity is an entity which **uses the primary key of a related entity set as part of its primary key**.

Weak entities are often created when a repeating group is removed to produce a separate entity with a **compound key**.

Weak entities can be implemented in a database system but they **cannot exist without a related strong entity**.

This is because the strong entity provides part of the primary key of the weak entity, therefore if the strong entity does not exist then the related weak entity cannot exist.

Weak entities are shown in an ER diagram using a double rectangle rather than a single one.



A dotted line is used to connect a weak entity to the strong entity it relies on.

Strong entity

Any entity which has a single attribute primary key is a strong entity. A strong entity is independent of any other entity in the database.

When strong entities are implemented in a database system, they can exist without relying on any other entity.

Strong entities are shown in an ER diagram using a single rectangle.

Example:

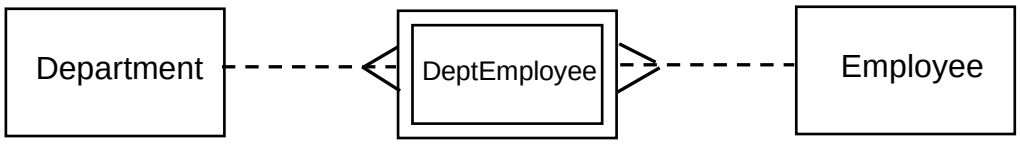
Take the data dictionary example from page 4.

Department	**Employee**	**deptEmployee**
deptID	employeeNumber	deptID (FK)
deptName	firstName	employeeNumber (FK)
location	surname	
	salary	
	startDate	

Department and Employee are **strong entities** in this case because they have their own primary key.

DeptEmployee is a **weak entity** as its primary key relies on at least one primary key from another entity.

The Entity Relationship diagram for this example would look like this:



Surrogate Key

A surrogate key is a primary key that has been made up to provide a unique value where there is no other obvious primary key.

In a pupils entity that stores the following information, there is no obvious primary key or even a combination that would provide a unique record.

Pupils
firstname
surname
age
house

Adding a **pupilID** attribute is the only option here to provide us with a primary key.

Pupils
pupilID
firstname
surname
age
house

PupilID would be a **surrogate key** because we have introduced it to create a primary key.

Surrogate Key and Weak Entities

It is common practice to remove weak entities by adding a surrogate key.

If a surrogate key is introduced as a single attribute primary key then the entity changes to a strong entity because it no longer relies on any other entity for part of its key.

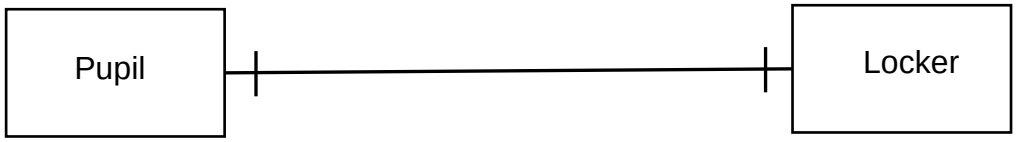
Department	**Employee**	**deptEmployee**
deptID	employeeNumber	referenceNumber
deptName	firstName	deptID (FK)
location	surname	employeeNumber (FK)
	salary	
	startDate	

By introducing referenceNumber as a unique primary key, the deptEmployee entity changes from a weak entity to a strong entity.

Relationship Participation

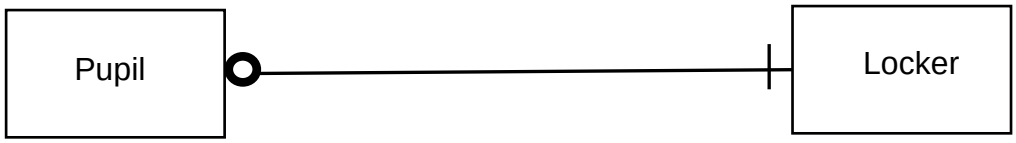
To show whether an entity has mandatory or optional participation in a relationship, hollow and solid circles are used on the ER diagram.

Mandatory – Mandatory



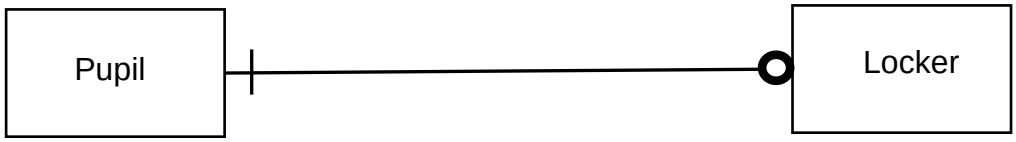
Pupil **must** have a locker and locker **must** be allocated to a pupil

Mandatory – Optional



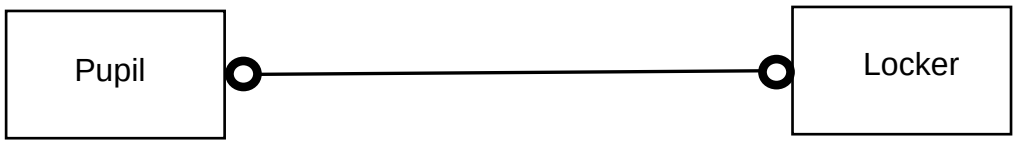
Pupil **must** have a locker but locker **may** be allocated to a pupil

Optional – Mandatory



Pupil **may** have a locker but locker **must** be allocated to a pupil

Optional – Optional



Pupil **may** have a locker and locker **may** be allocated to a pupil

Complete Example

A relational database is used by a travel agency to store details of Scottish holiday resorts and hotels in each resort. The database also stores details of customers and their hotel bookings. The booking, customer, resort and hotel details are arranged in four separate entities.

Details of the attributes stored in each entity have been provided below.

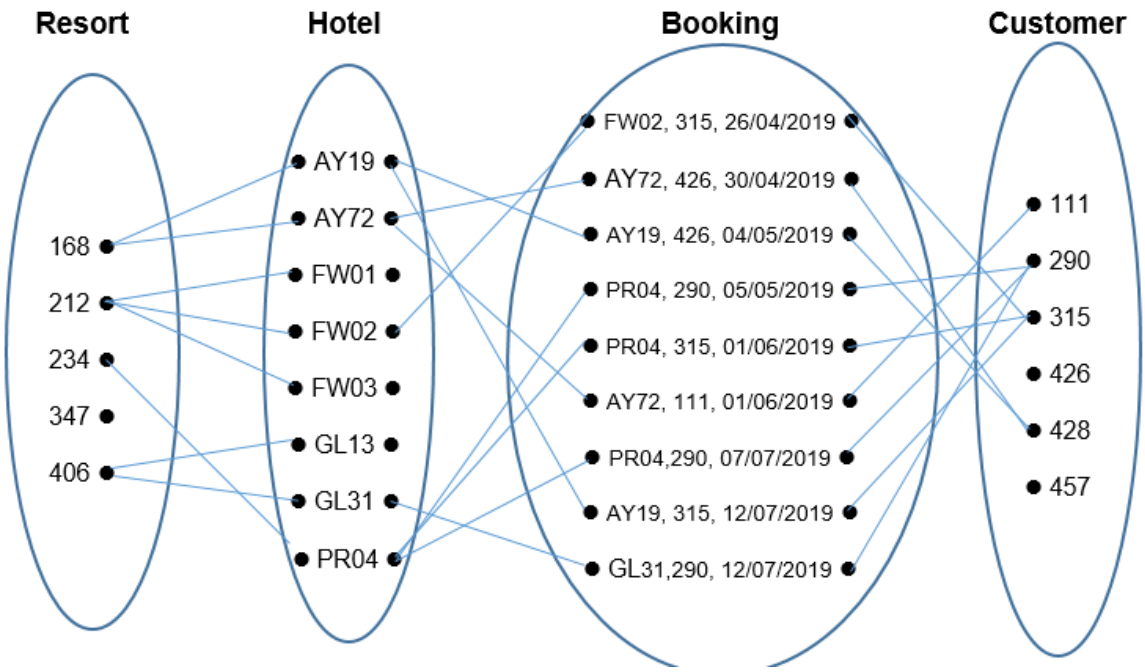
Resort	Hotel	Customer	Booking
resortID resortName resortType	hotelRef hotelName resortID * starRating seasonStartDate mealPlan checkInTime pricePersonNight	customerNo firstname surname address town postcode	hotelRef * customerNo * startDate numberOfNights numberInParty

Strong and Weak Entities

From the list of attributes, we can see that Resort, Hotel and Customer area all ***strong*** entities while Booking is a ***weak*** entity.

Relationship Participation

An entity occurrence diagram indicating the relationships between the entities is shown below.



Entity Relationship Diagram

A complete Entity Relationship diagram at Advanced Higher level should show all of the features from Higher:

- Entity Name
- Attributes
- Relationship name (label)
- Cardinality (1 to 1, 1 to many, many to many)

plus:

- Strong/Weak Entities (single/double rectangle)
- Relationship Participation (solid/hollow circle)