

|  |
| --- |
| DAT601 Milestone 1 |
| Conceptual Database Design |
| Name: Rhylei Tremlett  ID: 13513865  Date: 3/05/2023 |

Table of Contents

[Introduction 3](#_Toc134042731)

[Data Modelling Overview 3](#_Toc134042732)

[What is Data Modelling 3](#_Toc134042733)

[Entity Relationship Diagrams 3](#_Toc134042734)

[Conceptual Model 3](#_Toc134042735)

[Logical Model 3](#_Toc134042736)

[Physical Model 3](#_Toc134042737)

[Chen’s Notation 4](#_Toc134042738)

[Entities 4](#_Toc134042739)

[Strong Entity 4](#_Toc134042740)

[Weak Entity 4](#_Toc134042741)

[Associative Entity 4](#_Toc134042742)

[Attributes 4](#_Toc134042743)

[Candidate Key 5](#_Toc134042744)

[Weak Key 5](#_Toc134042745)

[Multivalued Attribute 5](#_Toc134042746)

[Derived Attribute 5](#_Toc134042747)

[Composite Attribute 6](#_Toc134042748)

[Relationships 6](#_Toc134042749)

[Strong Relationship 6](#_Toc134042750)

[Weak Relationship 6](#_Toc134042751)

[Cardinality 7](#_Toc134042752)

[One-to-One 7](#_Toc134042753)

[One-to-Many 7](#_Toc134042754)

[Many-to-Many 7](#_Toc134042755)

[Participation 8](#_Toc134042756)

[Partial 8](#_Toc134042757)

[Total 8](#_Toc134042758)

[Optionality 8](#_Toc134042759)

[Mandatory 8](#_Toc134042760)

[Optional 8](#_Toc134042761)

[Extended Notation 9](#_Toc134042762)

[Generalisation 9](#_Toc134042763)

[Specialisation 9](#_Toc134042764)

[Disjoint 9](#_Toc134042765)

[Overlapping 10](#_Toc134042766)

[Union 10](#_Toc134042767)

[Conceptual ERD 11](#_Toc134042768)

[Rationale 11](#_Toc134042769)

[Conceptual Data Dictionary 12](#_Toc134042770)

[Purpose 12](#_Toc134042771)

[Entities 12](#_Toc134042772)

[Relationships 12](#_Toc134042773)

[Attributes 12](#_Toc134042774)

[Rationale 13](#_Toc134042775)

[Staff 13](#_Toc134042776)

[Contract 13](#_Toc134042777)

[Zone 13](#_Toc134042778)

[Subscriber 13](#_Toc134042779)

[Sensor 13](#_Toc134042780)

[Subscriptions 13](#_Toc134042781)

[Video Steam and Data 13](#_Toc134042782)

[Assumptions 14](#_Toc134042783)

[References 1](#_Toc134042784)

# Introduction

# Data Modelling Overview

## What is Data Modelling

IBM provides a good definition of Data modelling. “Data modelling is the process of creating a visual representation of either a whole information system or parts of it to communicate connections between data points and structures.” (*What Is Data Modeling?*, n.d.)

Data modelling is a process used in database development; it involves creating diagrams of varying abstraction levels that represent the data within the system.

## Entity Relationship Diagrams

Entity relationship diagrams are a type of diagram that captures the entities of a system, as well as their attributes and relationships with other entities.

### Conceptual Model

The conceptual model is the most abstracted diagram in the data modelling process. The conceptual model is used to identify and document the relationships between entities and their attributes. The conceptual model can be implemented into most database systems through logical and physical models.

### Logical Model

The logical model has less abstraction and provides more insight into how the relationships and attributes are defined. The logical model adds information about attribute types and lengths, as well as forming joining tables from associative entities and creating foreign and primary keys that are used in the relationships. The logical model is not specific to any database system in particular, but it can only be implemented as a relational database. To produce any other kind of database, the logical model would be different. There is little consensus on the logical model. Sources will often contradict one another and will creep into the realm of physical or conceptual, rendering those models nearly useless.

### Physical Model

The physical model reflects the exact design of the relational database; it is the least abstracted and has the highest amount of technical detail. This model is effectively a schema for how the database will be implemented. The physical model takes the logical model and provides even greater detail by documenting the exact types used in the database system, as well as using proper row and column names instead of the entity name.

# Chen’s Notation

## Entities

An entity can be an object, person, location, role, event, etc. Almost anything that is a noun can be made into an entity.

### Strong Entity

A strong entity is an entity that’s existence isn’t dependent on any other entities. A rectangle represents strong entities.

Text

Description automatically generated

### Weak Entity

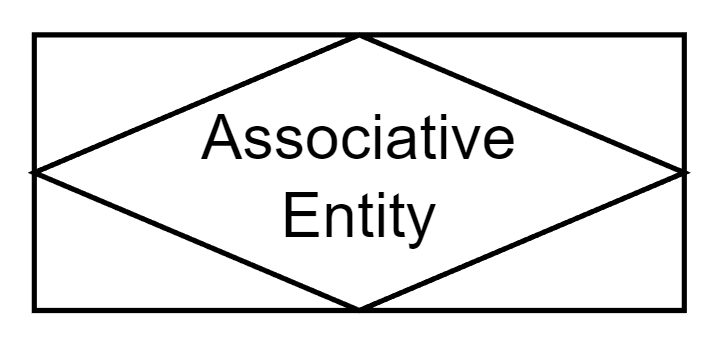
A weak entity is an entity that cannot be uniquely identified by itself, and it needs an accompanying entity to aid in the identification. Weak entities have a parent entity which is what will help identify it. A double rectangle represents them.

Rectangle

Description automatically generated with medium confidence

### Associative Entity

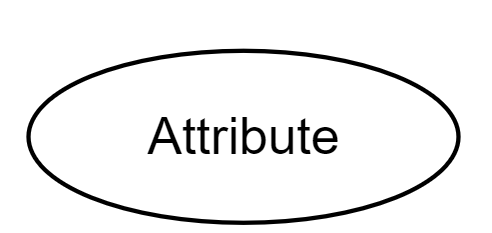
Associative entities represent many-to-many relationships, and they are verging on logical modelling as they effectively represent joining tables. A rectangle with a rhombus inside represents them.



## Attributes

Attribute

Attributes are the information about an entity. They show what will be stored within the database. They are represented by ovals.



### Candidate Key

A candidate key is any value (or multiple values) that can uniquely identify each record within the set. There can be many candidate keys, but only one will become the primary key. Candidate keys are represented by having underlined text.

A picture containing text, sport, athletic game

Description automatically generated

### Weak Key

Weak entities don’t have a candidate key to potentially identify them; a surrogate key usually wouldn’t help either. Instead, weak entities have a weak key which must be paired with the candidate key of the parent entity. They are represented by dashed underlined text.

Diagram

Description automatically generated

### Multivalued Attribute

Multivalued attributes have multiple values within themselves and tend to represent an array or list of items. They are represented by a double oval.

A black and white logo

Description automatically generated with low confidence

### Derived Attribute

A derived attribute is one whose value is calculated or created based on the values of other attributes. They are represented by a dashed border.

Shape, circle

Description automatically generated

### Composite Attribute

A composite attribute has multiple sub-attributes that altogether comprise the overall attribute. They are represented by being branched off from an existing attribute. They can also use a different colour.

Diagram

Description automatically generated

## Relationships

### Strong Relationship

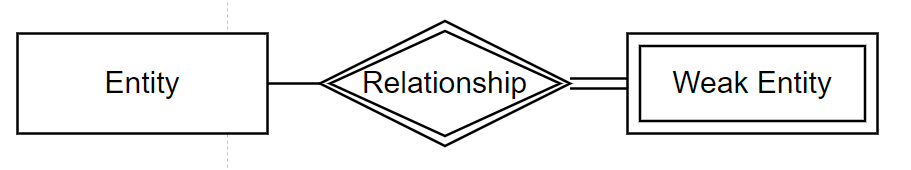
A strong relationship is one where the existence of both entities isn’t dependent on the other; in effect, neither is the parent or child of the other. They are represented by a single rhombus.

Diagram

Description automatically generated

### Weak Relationship

Weak relationships are used for parent-to-child relationships where the child entity is a weak entity. In a weak relationship, the child’s existence solely depends on the parents. They are represented by a double rhombus.

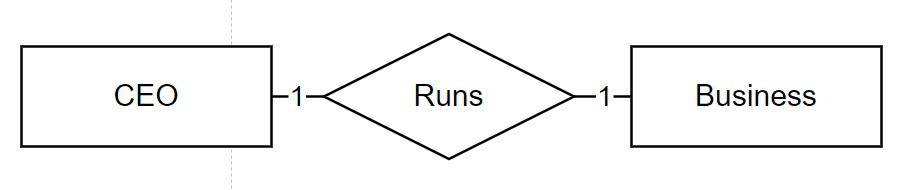


## Cardinality

Cardinality is how many records of one entity have a relationship with the records of another entity and to what ratio.

### One-to-One

A one-to-one relationship is where for every record of one entity, there is one in the other entity. Both entities are strong entities. However, one entity will take precedence over the other and be the primary entity of the relationship, and the other entity will receive the foreign key. These relationships are denoted by having a ‘1’ on both lines.



### One-to-Many

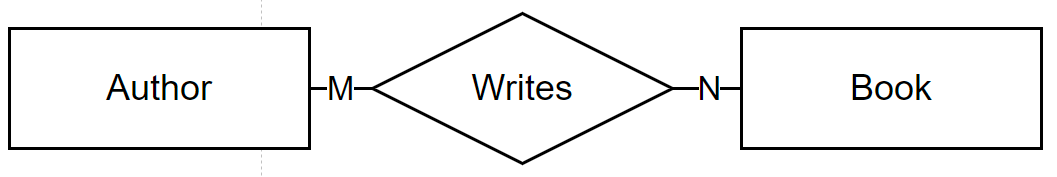
One-to-Many relationships are where one record from the primary entity (the 1) has a relationship with many records of the secondary entity (the ‘M’). In the example, one book has many chapters. In the example, the chapter is a weak entity because It is not identifiable without the book it belongs to.

A picture containing text, triangle

Description automatically generated

### Many-to-Many

Many-to-many relationships are when many records from one entity have a relationship with many records of another entity. In the above example, an author can write many books, and a book can have many authors; therefore, it is many-to-many. It is denoted by the primary entity having an ‘M’ and the second having an ‘N’.



## Participation

Participation is how many records of an entity participate in the relationship.

### Partial

In partial participation, not all records in the set have to participate in the relationship. Partial participation is denoted by a single line between the relationship and the entity.



### Total

In total participation, every record within a set must participate in the relationship. Total participation is denoted by a double line between the relationship and the entity.

Shape, rectangle

Description automatically generated

A good example to see the difference is the author and book. An author doesn’t have to have written a book to exist in the set, whereas a book can only exist if it is written by an author.

## Optionality

Optionality is used in relationships where there are more than two entities. It is used when one or more of the entities are not required for the relationship to take place.

### Mandatory

Mandatory is denoted by a solid line and means that the entity must be included in the relationship

### Optional

Optional is denoted by a dashed line and means that the entity is not required in the relationship.

Diagram

Description automatically generated

In this example, a customer is purchasing many items, and a salesperson can be included in the sale process or not; therefore, they are not required for the relationship to be valid and take place. If the line were solid, that would mean that for every purchase made, there would be a salesperson involved, which is only sometimes desired.

## Extended Notation

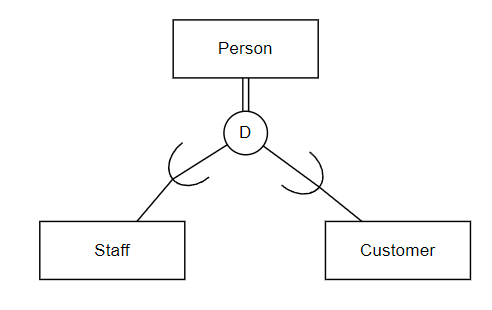
### Generalisation

Generalisation is when similar entities have common attributes removed and put into a superclass which they can inherit from.

### Specialisation

Specialisation is the reverse of generalisation, where an entity is found to have multiple sub-classes that have additional attributes that are not common among the other subclasses.

Both specialisation and generalisation use the same notation and represent the same thing from a different perspective.



### Disjoint

Disjoint means that an entity can only be one specialisation. Disjoint is show as a circle with a ‘D’.

Icon

Description automatically generated

### Overlapping

Overlapping means that an entity can be multiple specialisations at the same time, inheriting all of the attributes from each specialisation. Overlapping is shown as a circle with an ‘O’.

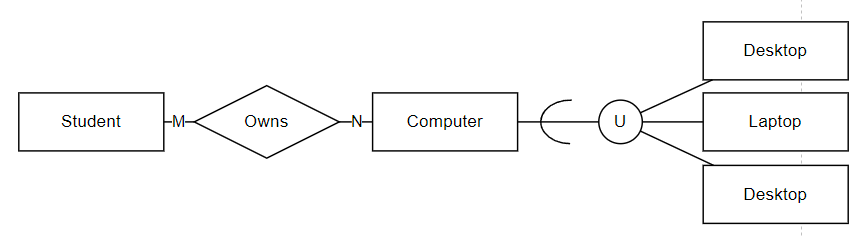
A picture containing diagram

Description automatically generated

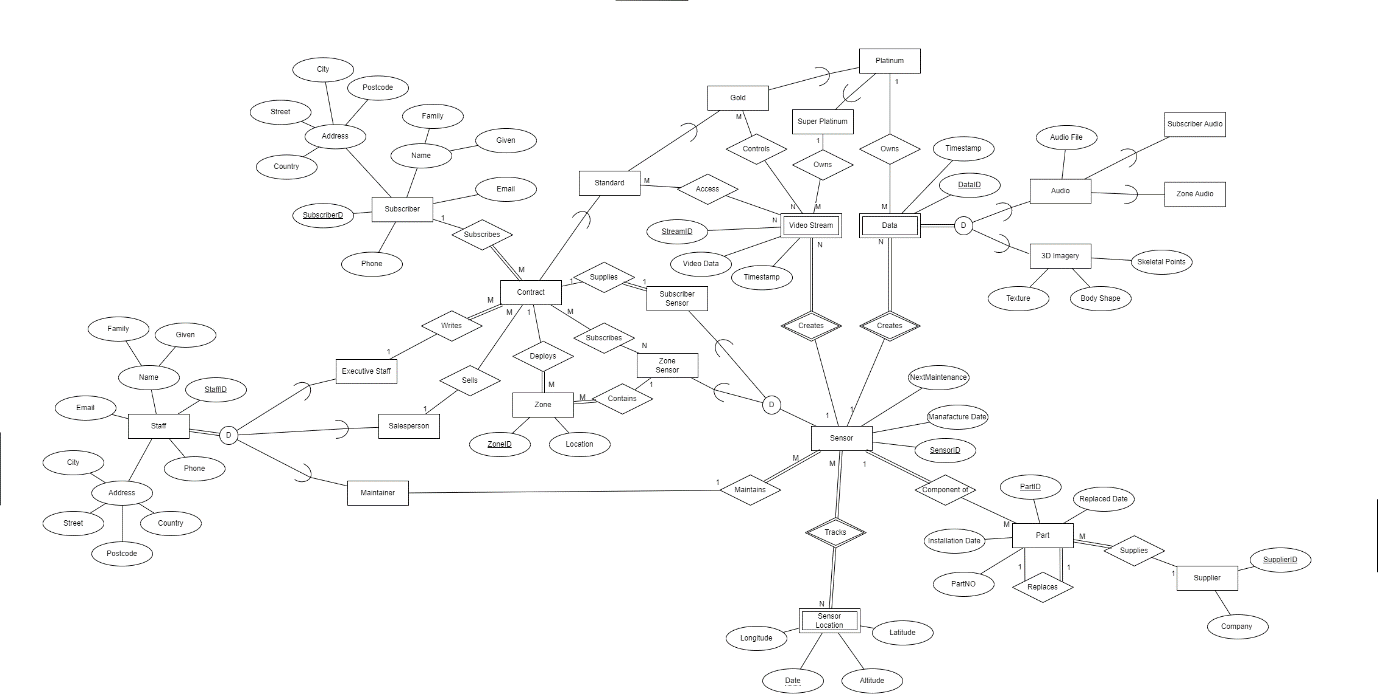
### Union

The union is the less commonly used extended component, but it has its place. It is used when an entity can be multple things or one of many things, but its relationships are not dependent on its forms.

Union is represented using a circle with a ‘U’, the superclasses are then ordered in reverse to the specialisation/generalisation.



# Conceptual ERD



- Double-click to open in the browser.

Although its an SVG the text doesn’t zoom in well.

## Rationale

I have combined my rationale for the ERD and Data Dictionary nearer to the bottom of the document.

# Conceptual Data Dictionary

## Purpose

Below are embedded Excel files that contain my data dictionary tables.

## Entities



## Relationships



## Attributes



# Rationale

## Staff

There are three kinds of staff. They share the same attributes but don’t share the same relationships, which is why they are specialisations of staff.

Not every staff member must participate in these relationships.

The specialisations are disjoint to separate the roles of responsibility; this would normally be a client decision.

## Contract

The contract has many relationships; in order to cover the standard and super platinum at the same time, I opted for having optional participation in most of the relationships. The reason for this is that while a standard subscription may subscribe to a 3D sensor, it is unlikey that a super platinum subscription would have a need for that. There are other cases where the standard differs from the two platinums. This could have been approached in a different manner. Next time I undertake a similar project, I will be more open to other approaches.

## Zone

I have modelled the zone how it is described in the project brief. However, in hindsight, I would have added more attributes as to whether it is private or what kind of access is provided to other levels of subscription.

## Subscriber

The subscriber is a standard person-type entity, so I gave it the basic attribute as I felt there wasn’t anything distinct about them. With this in mind, I could have had Subscriber and Staff inherit from a person generalisation.

## Sensor

The sensor entity itself is a superclass. Zone and subscriber sensors aren’t physically different and have the same attributes, but their relationships are different. Sensor location is something that would be updated whenever it is physically moved, so a timestamp will suffice.

## Subscriptions

Each of the subscriptions doesn’t have any attributes. They merely provide different relationships to the contract.

## Video Steam and Data

The Video Stream and audio will likely be captured as files. Data is a bit more complex. All three specialisations will be captured simultaneously but stored as separate records.

# Assumptions

The sensors used in zones are the same as the ones that are used by subscribers.

A contract is a subscription.

Maintainers are employed by Spaces.

Data created by sensors can be recorded simultaneously but stored separately.

Sensors can be contained within multiple zones.

While the sensor location can be tracked as a continuous stream, and my model does support that, it is extremely dumb to constantly track the location of stationary objects. Instead, Spaces should track its location only when it moves and stop when it comes to rest.

The sensors keeping onboard storage have nothing to do with the database, so no worries there.

Zone sensors provide the 3D video stream as well as zone audio which is akin to environmental audio or ambience.

Subscriber sensors provide both forms of audio as well as 3D imagery but do not upload a video stream of its user.

Subscriber audio is the subscriber’s voice extracted from the overall audio recorded by the sensor.

# References

*What is Data Modeling? | IBM*. (n.d.). Retrieved 3 May 2023, from https://www.ibm.com/topics/data-modeling