DAT601

Assessment 1

Josh Moss

Table of Contents

[Introduction to data modelling 2](#_Toc165501183)

[Conceptual modelling: 2](#_Toc165501184)

[Logical data modelling 2](#_Toc165501185)

[Physical implementation 2](#_Toc165501186)

[Description of conceptual modelling components (Chen’s ERD notation) 3](#_Toc165501187)

[Entity 3](#_Toc165501188)

[3](#_Toc165501189)

[Weak Entity: 3](#_Toc165501190)

[Associative Entity: 3](#_Toc165501191)

[Attributes: 4](#_Toc165501192)

[Key Attribute: 4](#_Toc165501193)

[Partial Key Attribute: 4](#_Toc165501194)

[4](#_Toc165501195)

[Derived Attribute: 5](#_Toc165501196)

[5](#_Toc165501197)

[Relationship: 6](#_Toc165501198)

[ Unary relationship (Recursive) 6](#_Toc165501199)

[ Binary relationship 6](#_Toc165501200)

[ Ternary relationship 6](#_Toc165501201)

[Weak (identifying) Relationship: 6](#_Toc165501202)

[6](#_Toc165501203)

[Optionality: 7](#_Toc165501204)

[7](#_Toc165501205)

[Cardinality: 7](#_Toc165501206)

[Conceptual Data model 8](#_Toc165501207)

[Assumptions made: 8](#_Toc165501208)

[Conceptual Model Documentation 9](#_Toc165501209)

[Table 1: Document Entities 9](#_Toc165501210)

[Table 2: Document Relationships 11](#_Toc165501211)

[Table 3: Attributes 12](#_Toc165501212)

[Assumptions 17](#_Toc165501213)

# Introduction to data modelling

In database design, data modelling is a critical aspect. It involves how data is structured within a database system. It encompasses the process of how data is organized and the relationships between this data. Data modelling is important to ensure data is managed accurately and efficiently. According to the “Principal components of relational model” (Vidhya, V, 2016, p, 2.5), there are three basic components the model must follow:

1. Data structure
2. Data integrity
3. Data manipulation

## Conceptual modelling:

In my understanding, conceptual database modelling should encapsulate all information that pertains to a specific enterprise without the consideration of physical implementation. In the conceptual stage, the focus should be on understanding the data structure and relationships between entities within the organization. The conceptual stage can be even more in-depth than the logical model as it encapsulates all ideas and possibilities that could be potentially present in the final database system. Even if some of those ideas are not present in the final system. Chen's ERD notation allows for the creation of diagrams that represent real-world objects, which can be translated into entities with relationships and attributes. This provides stakeholders and designers with a visual representation of the project's scope.

## Logical data modelling

The logical data model takes the elements from the conceptual model and adds more detail on how the data is organized within the entities, attributes, and relationships. It is essentially a low-level outline of how the database is structured. Designers can use logical models to translate requirements from the conceptual model into a structure suitable for the database system's physical implementation.

## Physical implementation

The physical implementation phase would involve translating the data model into a functional database. Tables will be created containing specific datatypes and constraints that correspond to the logical data model. Constraints, such as primary and foreign keys, act as rules to enforce data integrity. This helps ensure the data remains consistent and adheres to the defined structure, contributing to overall data accuracy.

# Description of conceptual modelling components (Chen’s ERD notation)

*Note: All figures and depictions were created using* [*Draw.io*](https://app.diagrams.net/)

# Entity

# **A close-up of a sign Description automatically generated**

Entities represent real-world objects or concepts that you want to capture specific data for. Entities are independently identified from other entities. According to Vidhya “An entity may be concrete, such as person or a book, or it may be abstract, such as a loan, or a holiday” (Vidhya, V., 2016, p, 2.8).

* An example of an entity could be a “customer”.
* Entities are defined by their attributes.

## Weak Entity:

**A black rectangle with black text

Description automatically generated**

An entity is considered weak when it depends on another entity to exist. This means it cannot be uniquely identified by its attributes. This entity exists dependently on another entity’s instance.

## Associative Entity:

**A white rectangle with black text

Description automatically generated**

The associative entity is used when a many to many relationships exists and you want to capture additional specific information within this entity (represents additional table)

# Attributes:

A black text on a white background

Description automatically generated

Attributes help define the characteristics of an entity. Attributes enable you to store information about a particular entity, for example, a customer’s details. A customer entity could have attributes such as “First Name”, “Last Name”, or “Email”. Relationships between entities can share attribute values, either through an identifying attribute or key attribute. Defining relevant attributes is important, as it aids in creating a well-structured database that effectively captures the information about the entities.

* **Simple attributes:** The attribute IS NOT divisible. E.g. Age.
* **Composite attributes:** the attribute IS divisible. E.g. Address: street, city.

**A diagram of a structure

Description automatically generated**

* **Single valued attributes:** Contains only a single value.

## Key Attribute:

A black text on a white background

Description automatically generated

A key attribute is a unique identifier that represents a specific entity instance. There can only be one “identifier” per entity. furthermore, no two instantiated entities of the same type can contain the same key value. A primary key is chosen (or composite key) to represent the retrieval of instantiated entities.

## Partial Key Attribute:

### A close-up of a label Description automatically generated

A partial key attribute provides an identifier for a weak entity by combining with the key attribute of the owner entity.

## Derived Attribute:

### A black text on a white background Description automatically generated

Derived attributes values are not stored, instead their values are calculated from other attributes in the database system. One benefit for using derived attributes is that you avoid storing potentially redundant data.

Multi valued Attribute:

**A black text in a oval shape

Description automatically generated**

“An attribute that can have many values” (Dybka, P. ,2019.)

Multivalued attributes allow an entity to contain multiple values within the same attribute, for example, a “PhoneNumber” attribute could hold a mobile number and home phone number within one attribute.

# A black hexagon with black text Description automatically generatedRelationship:

Relationships exist when an attribute of an entity refers to another entity type. There is a degree off relationship depending on the number of entities that participate in the relationship.

## Unary relationship (Recursive)

If a relationship contains an association within a single entity, then it’s a binary relationship.

A diagram of employee's work flow

Description automatically generated

## Binary relationship

If two entity types are participating in a relationship, then it’s a binary relationship.

A diamond shaped sign with black text

Description automatically generated

## Ternary relationship

If three entities are participating in a relationship, then it’s a ternary relationship. A diagram of relationship

Description automatically generated

### Weak (identifying) Relationship:

### A black hexagon with black text Description automatically generated

This relationship exists when a weak entity relies on a primary key from an owner entity to be uniquely identified.

## Optionality:

* Mandatory relationship

This relationship is shown as a double line. This means the entity must participate in the relationship.

* Optional relationship

Depicted as a single line. This means that the entity does not necessarily need to participate in the relationship.

### 

## Cardinality:

Cardinality represents the number of instantiated entities that can associate with one another. specific characters 1, N or M are used to determine this.

* One to one

One entity instance can relate with no more than one other entity instance.

A black and white hexagon with black text

Description automatically generated

* One to many (many to one)

One entity instance can relate with many instances of another entity.

A black and white hexagon with black text

Description automatically generated

* Many to many

Many instances of an entity instance can relate with many instances of another entity.

A black and white hexagon with black text

Description automatically generated

# A diagram of a diagram Description automatically generatedConceptual Data model

## Assumptions made:

* In this model I have assumed that contracts are bound only to Super Platinum subscribers. However, if each subscription required a contract, I could move the relationship between “Contract” and “Subscription”.
* I am assuming that maintenance contractors are employed by Flight Stream, although this may not be the case, as maintenance teams could be employed by external entities.
* I have assumed that gold subscribers have access to control hardware (Video Cameras) recording the live stream service. Therefore, the relationship will be between the gold subscription and the Data Scoop. However, If the ability was only to control the instance of the Livestream itself. Then perhaps the relationship is between the gold subscription and the live stream.
* Note “Address” is a composite attribute. The depiction may be confusing. Additionally, Name is a multi-valued attribute but can be changed to a composite attribute with “First Name” “Last Name” In logical model if needed.

*Note: Additional assumptions are made at the end of the document.*

# Conceptual Model Documentation

## Table 1: Document Entities

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity Name** | **Description** | **Aliases** | **Occurrence** |
| Client | Customer of Flight Stream that utilises their services | - | A client wants to access Flight Stream services |
| Account | A client made account unique to a specific client | - | A client creates an account for Flight Streams services |
| Subscription | A subscription to give clients access to specific Flight Stream services | - | A Client purchases a subscription |
| Contract | A special contract added to a specific subscription granting access to additional data | - | A Client purchases a subscription with a special contract to access data |
| DataScoop | Drone used by Flight Stream to provide services for clients | ds | DataScoop provides live video streams for its viewers |
| Employee | Personal that works for Flight Stream | - | - |
| LiveStream | Live video feed from the DataScoop drone | - | Clients interact with live streaming services provided by a DataScoop |
| Data | Information recorded by the DataScoop for specific clients to access | - | Client accesses recorded data from the DataScoop |
| Maintenance | A record of maintenance carried out by contractors | - | Maintenance contractor records maintenance carried out for a specific DataScoop |
| ScoopMaintenanceContractor | Contractor that carries out maintenance on DataScoops | - | If a DataScoop needs repair/maintenance, then a contractor is sent |
| DroneComponents | Components used to repair/maintain DataScoops | - | DataScoop requires upgrading its components |
| Supplier | Supplier of done components for the repair/maintenance of DataScoops | - | Contractors require parts to repair/maintain DataScoops |
| SalesRepresentative | Employee of Flight Stream who sell subscriptions to clients | - | Sales Representative selling subscriptions to clients |
| Zone | A specific area designated to a DataScoop | - | A DataScoop is transferred to a new zone |
| Administrator | Employee of Flight Stream who oversees contracts | - | An Administrator edits a contract |
| Gold | An upgraded version of a standard subscription that grants additional access | g | Gold subscribers gain the ability to control Data Scoops video streams |
| Platinum | An upgraded version of a gold subscription that grants additional access | p | Platinum subscribers gain access to data collected by the DataScoop |
| Super Platinum | An upgraded version of a platinum subscription that grants additional access | sp | Super Platinum subscribers have an exclusive right to specific DataScoop Zones |

## Table 2: Document Relationships

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Entity Name** | **Cardinality** | **Participation** | **Relationship** | **Participation** | **Cardinality** | **Entity Name** |
| Client | 1 | Mandatory | Creates | Optional | N | Account |
| Account | N | Mandatory | Acquires | Optional | 1 | Subscription |
| Subscription | N | Mandatory | Sells | Optional | 1 | SalesRepresentative |
| Administrator | 1 | Optional | Edits | Mandatory | N | Contract |
| Supplier | 1 | Optional | Supplies | Mandatory | N | DroneComponent |
| ScoopMaintenanceContractor | 1 | Optional | Utilizes | Optional | N | DroneComponent |
| Maintenance | N | Mandatory | Records | Optional | 1 | ScoopMaintenanceContractor |
| ScoopMaintenanceContractor | 1 | Mandatory | Maintained | Optional | N | DataScoop |
| DataScoop | N | Optional | Records | Mandatory | 1 | Data |
| DataScoop | N | Optional | Broadcasts | Mandatory | 1 | LiveStream |
| DataScoop | N | Optional | Resides | Optional | M | Zone |
| SuperPlatinum | 1 | Optional | Owns | Optional | N | Zone |
| Platinum | N | Optional | Views | Mandatory | M | Data |
| Gold | N | Optional | Controls | Optional | 1 | DataScoop |
| Subscription | 1 | Optional | Views | Optional | N | LiveStream |
| Administrator |  |  | Is a |  |  | Employee |
| SalesRepresentative |  |  | Is a |  |  | Employee |
| ScoopMaintenanceContractor |  |  | Is a |  |  | Employee |
| Gold |  |  | Inherits |  |  | Subscription |
| Platinum |  |  | Inherits |  |  | Gold |
| Super Platinum |  |  | Inherits |  |  | Platinum |

| **Entity Name** | **Attributes** | **Description** | **Domain** | **Aliases** | **Composite** | **Derived** | **Nulls** | **Key?** | **Default Value** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Client** | Name | Frist Name, Last name of the client | VARCHAR | - | NO | NO | NO | NO | - |
| Address | Location of the client | VARCHAR/INTEGER | - | Country, Zip Code, City, Street Name, Street Number, Suburb | NO | NO | NO | - |
| Phone | Cell phone, home phone numbers belonging to the client | VARCHAR | - | NO | NO | NO | NO | - |
| Email | Email address belonging to the client | VARCHAR | - | NO | NO | NO | NO | - |
| ClientNumber | Unique Identifier for the client within the system | INTERGER | - | NO | NO | NO | PK | - |
| Account | AccountNumber | Unique number issued for a clients account when creating an account | INTEGER | - | NO | NO | NO | PK | - |
| Display Name | Display Name chosen by the client must be unique | VARCHAR | - | NO | NO | NO | CK | - |

## Table 3: Attributes

| **Entity Name** | **Attributes** | **Description** | **Domain** | **Aliases** | **Composite** | **Derived** | **Nulls** | **Key?** | **Default Value** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Subscription | SubscriptionID | Identifies an instance of unique subscription | VARCHAR | - | NO | NO | NO | PK | - |
| Discount | A discount applied to a specific subscription instance | DECIAML | - | NO | NO | YES | NO | - |
| Employee | EmployeeID | Unique identifier for an employee | INTEGER | - | NO | NO | NO | PK | - |
| Email | Email address belonging to the employee | VARCHAR | - | NO | NO | NO | NO | - |
| Phone | Cell phone, home phone numbers belonging to the employee | VARCHAR | - | NO | NO | NO | NO | - |
| Name | Frist Name, Last name of the employee | VARCHAR | - | NO | NO | NO | NO | - |
| Address | Location of the employee | VARCHAR/INTEGER | - | Country, Zip Code, City, Street Name, Street Number, Suburb | NO | NO | NO | - |
| SalesRepresentative | SalesRepresentativeID | Uniquely identifies specific sales representatives | INTEGER | - | NO | NO | NO | PK | - |

| **Entity Name** | **Attributes** | **Description** | **Domain** | **Aliases** | **Composite** | **Derived** | **Nulls** | **Key?** | **Default Value** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Administrator | AdminID | Uniquely identifies specific administrator | INTEGER | - | NO | NO | NO | PK | - |
| ScoopMaintenanceContractor | MaintenanceContractorID | Uniquely identifies specific MaintenanceContractor | INTEGER | - | NO | NO | NO | PK | - |
| Supplier | SupplierID | Uniquely identifies specific supplier | VARCHAR | - | NO | NO | NO | PK | - |
| Name | Name of the supplier/suppliers off drone components | VARCHAR | - | NO | NO | NO | NO | - |
| DroneComponent | ComponentID | Uniquely identifies specific drone component | VARCHAR | - | NO | NO | NO | PK | - |
| Price | Price of the drone component | DECIMAL | - | NO | NO | NO | NO | - |
| ComponentName | Name of the component | VARCHAR | - | NO | NO | NO | NO | - |
| Quantity | The amount of components received | INTEGER | - | NO | NO | NO | NO | - |
| Maintenance | MaintenanceRecord | Uniquely identifies a specific instance of maintenance | VARCHAR | - | NO | NO | NO | PK | - |
| MaintenanceDiscription | Optional description of completed maintenance | VARCHAR | - | NO | NO | YES | NO | - |
| Date | Date that specific maintenance is done | DATE/TIME | - | NO | NO | NO | NO | - |
| DataScoop | DataScoopID | Uniquely identifies a specific DataScoop | VARCHAR | ds | NO | NO | NO | PK | - |
| Data | RecordID | Uniquely identifies a specific data record from the DataScoop | VARCHEAR | - | NO | NO | NO | PK | - |
| Time | Identifies time of data capture | DATE/TIME | - | NO | NO | NO | NO | - |
| Temperature | Recorded temperature data | VARCHAR | - | NO | NO | NO | NO | - |
| OrganicData | Recorded Organic Data | VARCHAR | - | NO | NO | NO | NO | - |
| AmbientLightStrengtth | Recorded Light strength | VARCHAR | - | NO | NO | NO | NO | - |
| Humidity | Recorded Humidity | VARCHAR | - | NO | NO | NO | NO | - |
| Zone | Climate | Climate of the zone snow, arid etc | VARCHAR | - | NO | NO | NO | CK | - |
| Location | Location of the world LAT/LONG | VARCHAR | - | NO | NO | NO | CK | - |
| LiveStream | StreamID | Uniquely identifies a specific instance of a livestream | VARCHAR | - | NO | NO | NO | PK | - |

| **Entity Name** | **Attributes** | **Description** | **Domain** | **Aliases** | **Composite** | **Derived** | **Nulls** | **Key?** | **Default Value** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Contract | ContractID | Uniquely identifies a specific contract | VARCHAR | - | NO | NO | NO | PK | - |
| OrganizationName | The name of the organization who has ownership of the contract | VARCHAR | - | NO | NO | NO | NO | - |
| Gold | Price | Price of the gold subscription | DECIMAL | g | NO | NO | NO | NO | - |
| Platinum | Price | Price of the platinum subscription | DECIMAL | p | NO | NO | NO | NO | - |
| Super Platinum | Price | Price of the super platinum subscription | DECIMAL | sp | NO | NO | NO | NO | - |

*Note : See assumptions below for reflection and rationale of data dictionary*

# Assumptions

* I assumed implementing an account to act as a middle ground between clients and subscriptions to be a logical choice. I believe this as a client can then create multiple accounts for separate requirements (business, personal). Additionally, clients can create a desired display name to interact with the live stream. This adds a layer of security by not displaying real names on the live stream. As a counterargument, removing the account entity and having a subscriber directly relating to a specific subscription could result in less complexity. However, when moving into a logical model, I feel normalization becomes skewed as you have too many attributes that do not directly correspond to a specific “subscriber”.  I understand, however, that this thought process is potentially outside of the scope of a conceptual model.
* The subscription entity (standard subscription) acts as a parent entity for each subscription instance and is inherited by all subsequent subscription entities. Furthermore, each subsequent subscription tier inherits from the previous. (gold, platinum, super platinum).
* The use of a data dictionary helps define all data elements that pertain to the system in a clear and concise manner. It aids in giving a user a more in-depth understanding of the data system before the transition into a logical model. Without the data dictionary, the transition into the next phases would be vague, lacking in established datatypes and potential validation rules.
* For my data dictionary entities table, I aimed to give a clear but brief overview of the core components providing a short description of each entity, hopefully assisting any external actors in understanding my data model. As for my relationships, this data is crucial for understanding how my data interacts within my system. It defines cardinality and participation between my entities. Lastly, my attributes table. This details each attribute that belongs to an entity within my system. The table contains descriptions, domains, keys, and null values. Detailed views of attributes such as this help ensure data is consistent and accurate throughout the data modelling process.

References

Dybka, P. (2019, February 13). Chen notation. Vertabelo Data Modeler. <https://vertabelo.com/blog/chen-erd-notation/>

Vidhya, V., et al. *Database Management Systems*, Alpha Science International, 2016.*ProQuest Ebook Central*, <https://ebookcentral.proquest.com/lib/lnmit/detail.action?docID=5248352.>

Learn Computer Science. (2020, December 11). What Is Entity Relationship Diagram ( ERD ) | ER Model Explained In DBMS With Examples. YouTube. <https://www.youtube.com/watch?v=BaN7Ng7a82Y&ab_channel=LearnComputerScience>

<https://www.youtube.com/watch?v=VGxGELh5n2U&ab_channel=BrianFinnegan>