DAT601 ASSESSMENT

Database Design and Administration

Caleb Eason

Contents

[1. Introduction to data modelling in Information Systems 3](#_Toc168004184)

[1.1. Overview 3](#_Toc168004185)

[1.2. Principles of data management 3](#_Toc168004186)

[1.2.1. Data quality management 3](#_Toc168004187)

[1.2.2. Data Governance 4](#_Toc168004188)

[1.2.3. Privacy 4](#_Toc168004189)

[1.2.4. Security 4](#_Toc168004190)

[1.2.5. Lifecycle management 4](#_Toc168004191)

[1.2.6. Metadata management 4](#_Toc168004192)

[2. Conceptual Modelling with Enhanced Chen ERD 5](#_Toc168004193)

[2.1. Elements of the conceptual data model. 6](#_Toc168004194)

[2.1.1. Entity 6](#_Toc168004195)

[2.1.2. Attribute 6](#_Toc168004196)

[2.1.3. Relationship 7](#_Toc168004197)

[2.1.4. Weak entities 8](#_Toc168004198)

[2.1.5. Specialisations 9](#_Toc168004199)

[3. Conceptual ER Model 10](#_Toc168004200)

[3.1. Rationale 11](#_Toc168004201)

[3.2. Application of Data Management Principles 12](#_Toc168004202)

[4. Data Dictionary 13](#_Toc168004203)

[4.1. Entities 13](#_Toc168004204)

[4.2. Relationships 16](#_Toc168004205)

[4.3. Attributes 17](#_Toc168004206)

[4.4. Rationale 21](#_Toc168004207)

[4.5. Review of Data Management Procedures 21](#_Toc168004208)

[5. Assumptions on Business Rules 22](#_Toc168004209)

[6. Conceptual to Relational Logical Model 23](#_Toc168004210)

[6.1. Methodology 23](#_Toc168004211)

[6.2. Description of Normalisation Forms 24](#_Toc168004212)

[6.3. Logical Model 25](#_Toc168004213)

[6.3.1. Revised Conceptual Model 26](#_Toc168004214)

[6.3.2. Rationale 27](#_Toc168004215)

[7. Data Dictionary 30](#_Toc168004216)

[7.1 Relations 30](#_Toc168004217)

[7.2. Attributes 32](#_Toc168004218)

[7.3. Derived Attributes 41](#_Toc168004219)

[8. NaLER Analysis 42](#_Toc168004220)

[8.1. Purpose and Use in Information Systems 42](#_Toc168004221)

[8.2. Diagram Conventions 43](#_Toc168004222)

[8.3. Syntax Check 43](#_Toc168004223)

[8.4. Entity Analysis 44](#_Toc168004224)

[8.4.1. Attributes 44](#_Toc168004225)

[8.4.2. Binary Relationships 46](#_Toc168004226)

[8.5. Examples with Sample Data 49](#_Toc168004227)

[8.5.1. Attributes 49](#_Toc168004228)

[8.5.2. Binary Relationships 50](#_Toc168004229)

[8.6. Ternary or Higher Relationships 52](#_Toc168004230)

[8.7. Ternary Relationship Examples 52](#_Toc168004231)

[9. References 54](#_Toc168004232)

Milestone 1: Conceptual Modelling

# Introduction to data modelling in Information Systems

## Overview

In information systems, data modelling is the process of creating a visual representation of data flow and storage within a system. Data models show how data points and structures are connected to each other through entity relationships, how data is stored and formatted, and how data is organised within a system (IBM, retrieved 2024).

There are three distinct phases of designing a database, conceptual modelling, logical modelling, and physical implementation.

The conceptual model is used for discovering the data requirements of system and defining how data entities relate to one another. In a conceptual model, data entities have attached attributes that define their properties and are linked together with relationship objects. For each entity, at least one candidate key is chosen as a possible identifier.

The logical model restructures the conceptual model to align with a relational database structure. In the logical model, entities are converted to relations which contain their attributes within themselves. Datatypes and sizes are defined for each attribute and a primary key is chosen from the available candidate keys. Relationship objects are removed and replaced with foreign keys in linked relations.

For the physical implementation, adjustments are made to the logical model to ensure it matches the requirements of the chosen database system. The physical implementation will vary depending on what database system is used. For example, one database system might use strings to store text, while another may use varchars. Physical implementation is sometimes accompanies by a physical data model, a redrawing of the logical model that matches the characteristics of the database system.

## Principles of data management

There are several fundamental principles that should be followed to ensure effective data management within an organisation.

### Data quality management

Data quality refers to the accuracy, completeness, consistency, validity, and timeliness of data. High quality data should correctly serve the outcome it is intended for (Sagacity Solutions, 2024). I.e. User account data should reflect a user’s details, ensuring they are accurately represented, and can properly utilise a system. An absence of data quality management will result in erroneous data entering a system. This can cause issues for users and businesses, who rely on data to make decisions and perform tasks. Low quality data can impair the functionality of a system which may require data in specific formats. To ensure data entering a system is of high quality, data management practices must be implemented. All incoming data should be profiled and controlled to prevent intake of erroneous data (Jotform, 2020).

### Data Governance

Data governance is the process of managing data to comply with internal and external policies and regulations, and to ensure effective use of data. Effective data governance aims to maintain data availability, integrity, security, and usability, to ensure data is accurate, consistent, and legal. In large organizations this is generally handled by a data governance team. This team will define and enforce polices and strategies for data governance within an organization (Stedman, 2022).

### Privacy

Data privacy refers to how an organisation handles sensitive personal and/or business information. To ensure compliance with legal requirements and to maintain a positive reputation, companies must utilise proper strategies for collection, transfer, storage, sharing, and use of sensitive data. Sensitive data includes any personal identifiable information from customers or employees such as contact details, health records or financial data, and any business information such as research or financial information (Gattis, 2024). Failure to implement proper data privacy policies could cause harm to a company and its customers and employees, which may result in legal repercussions. Companies must familiarize themselves with the legal requirements of the countries they are operating in and providing services to, to ensure they are in compliance.

### Security

Data security is the process of controlling and restricting access to information to protect it from unauthorised access, disclosure, tampering, or destruction. Data security can be enforced by implementing authentication and authorisation procedures to limit who has access to certain pieces of data. Authentication prevents any internet user who managers to find a database from accessing its contents. Authorisation ensures any individual who is allowed to access a database can only view the data they have been specifically authorised to access. For example, an employee may be able to view a customer’s name and email but may be restricted from viewing their payment information. Further security can be provided with encryption, masking, and anonymisation of data. Masking and anonymization scrubs data of any personal identifiable information, enabling data analysts to view data without compromising the privacy of the individuals it pertains to. Encryption of data ensures that even if a data breach occurs, the data cannot be interpreted without a key to decrypt the data (Fortinet, 2024).

### Lifecycle management

Data lifecycle management refers to how an organisation manages data in an information system throughout its lifecycle. The data lifecycle encompasses, creation, initial storage, use, long term storage, and eventual deletion of data. Companies will need to enact data lifecycle management policies to ensure data storage is inline with the key principles of data management throughout the data lifecycle (Sheldon, 2022).

### Metadata management

Metadata is data about data. Data Metadata provides information about data schemas, structures, content, context, access, ownership, data types, dependencies, models, and relationships. Effective management of metadata can significantly improve the useability of a data for an organisation. Good metadata enables better understanding of data, improving its consistency and availability, and supporting data governance and analysis. To ensure metadata is effective, business need to implement policies for capturing and storing metadata. Meta data must be actively maintained to ensure it is up to date and must be disseminated in a way that makes it easily accessible to those with access (Ranjan, 2024).

# Conceptual Modelling with Enhanced Chen ERD

A conceptual data model is a structured diagram that depicts the required data and data interactions to support business processes. The model is not focused on physical characteristics of data or processing flow, rather it focuses solely on identifying data requirements, and defining the overall structure of data. (Sherman, 2015). Conceptual models are built using specific components that represent different elements of a data structure. These elements are linked with relationships which define how they interact. The extended Chen notation expands on the conceptual model by enabling data entries (see below) to have specialisations (Ranjan, 2024).

The structured nature of the conceptual model helps enforce data quality. Defining exactly what attributes an entity has ensures data is complete and consistent. Carefully defining exactly what an entity is, how it relates with other entities, and what data it needs, helps to ensure data can be used to correctly serve its intended purpose. Creating a conceptual model requires a designer to consider strategies for data governance. The database architect will need to consider exactly how the entities and their interactions will align with internal and external polices when designing the structure of a conceptual model. The conceptual model will also highlight what types of information a system needs to store. This will bring attention to any sensitive information that will be stored in the database, requiring the designer to consider privacy policies for the database. While data security is not directly addressed in the conceptual model, the model provides a helpful view for visualising and determining what data access procedures need to be implemented. Relationships between entities can reveal some information on required access roles by showing which entities need to communicate with each other, and which don’t need anything to do with one another. Specialization of entities from the enhanced Chen ERD notation can also show what different access roles can be implemented into a database. Specialization of an entity that have different relationships will show the differing access roles that can be applied to an entity. The conceptual model promotes lifecycle management by directly addressing how data is stored, and how data can be used. The model also shows how the deletion of data from one entity will affect data in other entities though relationships. The conceptual model also reveals what metadata will be required for system structures and relationships. When designing a conceptual model, the database architect will need to consider candidate keys for identifying entities. Cardinality, optionality, and participation in relationships. And the content and context of data within entities. Specialization and generalisations from the enhanced Chen ERD notation can also provide additional information on data structure metadata.

## 2.1. Elements of the conceptual data model.

### Entity

A blue rectangle with white text

Description automatically generated

An entity represents a specific object within a system. An entity can be anything within a system that has data of some type. For FlightStream, some example entities include customers, DataScoops, and contracts. An entity is depicted as a rectangle with its name in the centre. An entity can have several attributes that describe its properties and can be connected to other entities through relationships.

### Attribute

A blue and white diagram

Description automatically generated

Attributes are attached to entities. They are the properties that describe an entity and can represent any piece of data that is attached to an entity. For example, a customer entity might have the attributes name, email, and phone number. An attribute is depicted as an oval attached to an entity with a straight line.

An entity can have a number of special types of attributes.

A blue oval sign with white text

Description automatically generated

A primary key attribute represents a unique identifier for an entity. Each instance of an entity must have a unique primary key attribute. A primary key attribute is indicated by underlining its name.

A multivalued attribute comprises multiple pieces of data as a single attribute. For example, latitude, longitude, and altitude could be stored as a single multivalued attribute called location. A multivalued attribute is represented with an inset outline.

A derived attribute is an attribute that is calculated from existing data. For example, a customer might have an attribute, age, which is calculated from another attribute, date of birth. There is no special notation for a derived attribute.

### Relationship

A blue diamond with white text

Description automatically generated

A relationship represents an connection between two entities, describing how they interact. For example, a customer entity might *view* a contract entity.

A blue diamond with white text

Description automatically generated

A relationship is described with a diamond and is attached to entities with a line.

Relationships between entities have different parameters that describe the relationship.

Participation dictates whether an entity is required to partake in a relationship. Participation can be either mandatory, or optional. Mandatory participation means an entity instance must be part of the relationship. For example, an employee of a company must be associated with a specific company branch. Optional participation means an entity is not required to partake in a relationship. For example, a company branch can exist without any employees. Mandatory participation is denoted with a double line, optional participation is denoted with a single line.

A blue diamond with white text

Description automatically generated

Cardinality dictates how many instances of an entity can partake in a relationship. Cardinality can be either one or many. For example, a branch of a company can have many employees, but an employee can only work for one branch. A cardinality of one is represented by a number 1 placed above the connecting line for the appropriate entity. A cardinality of many is represented by a letter M placed above the connecting line for the appropriate entity. In a relationship where both entities have a cardinality of many, one of the M symbols is replaced by an N to distinguish each side of the relationship.

A blue diamond with white text

Description automatically generated

A blue diamond with white text

Description automatically generated

### Weak entities

A blue sign with white text

Description automatically generated

A weak entity is an entity is an entity that is identified through a relationship with another entity. Weak entities can not exist on their own and do not have a primary key. For example, an employee may have dependents that can not exist without the employee. A weak entity is depicted with a inset outline. The relationship that defines the weak entity is called an identifying relationship and is depicted with an inset outline.

A blue diamond with white text

Description automatically generated

### Specialisations

A diagram of a company

Description automatically generated

Enhanced Chen notation introduces the idea of specialisations. A specialisation is a type of entity. That builds of a base entity called a generalisation. Specialisation entities inherit all the attributes and relationships of the generalisation and can augments it with additional attributes and relationships. Specialisations allow an entity to have different types without needing to define an entirely new entity. For example, a company may have several different types of employees, such as accountants, customer service representatives, and engineers. Each of these employee types will share some common attributes such as name and email, while also having some unique attributes. For this case, an employee entity can be created with the shared attributes, then specialisations can be created to contain the unique attributes for each type of employee.

If an instance of an entity can have multiple specialisations, the specialisations are described as overlapping. If an entity instance can only have a single specialisation, the specialisations are described as disjoint. Overlapping specialisations are depicted with a letter O placed in the connecting circle. Disjoint specialisations are depicted with a letter D placed in the connecting circle.

A blue circle with white text

Description automatically generated A blue circle with white letters

Description automatically generated

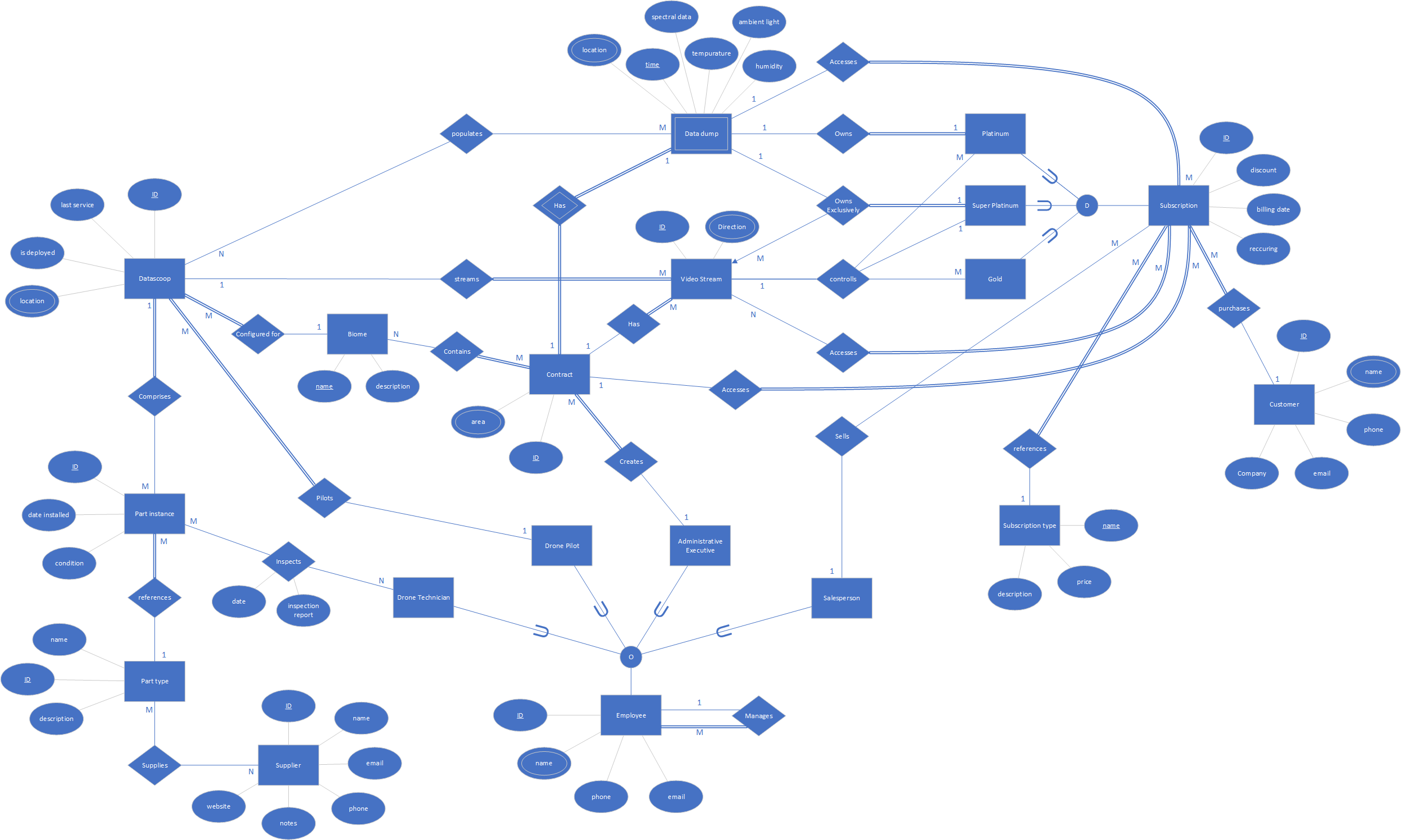
Specialisations can be mandatory or optional. This is depicted with the same notation as relationship participation. If an entity must have a specialsation, a double line is used for the connection between the generalisation and the specialsation circle. If an entity is not required to have a specialisation, a single line is used.

A blue sign with white text

Description automatically generated A blue sign with white text

Description automatically generated

# Conceptual ER Model



## Rationale

A diagram of a diagram

Description automatically generatedThe section of the diagram that displays the relationships between subscription specialisation, and the data dumps and video streams has some relationships that include more than two entities. It is important to note that this does NOT represent a ternary or quaternary relationship. This is a simplification of the actual relationship structure intended to improve the readability of the diagram. The actual relationships are as follows:

* Platinum Owns Data Dump
* Platinum Controls Video Stream
* Super Platinum Owns Exclusively Data Dump
* Super Platinum Owns Exclusively Video Stream
* Super Platinum Controls Video Stream
* Gold Controls Video Stream

This simplification of the diagram was made to greatly improve the readability of the diagram. Individually depicting each relationship in the diagram creates structure that is difficult to follow visually, as a result this compromise was made.

The customer entity is not directly connected to the contract, data damp, and video stream entities. This may appear as a chasm trap, but this is not the case. At first glance, it may seem that the customer should have a relationship with these entities. However further examination will reveal this is not the case. This is because accessing this information will be handled by a system external to the database, such as an online website for watching video streams. This system will determine access, by seeing if a customer has a relationship with a subscription that allows them access to the information from the contract, video stream and data dump entities. Within the database, there is no information in these entities that is dependent on a customer entity, so the relationship is not required.

The subscription type table is a reference table that holds information about a type of subscription. This table is referenced by the subscription table which represents a specific instantiation of a subscription. This means an administrative executive can easily update information about a subscription type without needing to change attributes in every single instance of a subscription.

The data dump is a weak entity identified by its relationship to the contract entity. This is because each contract entity has unique data dump linked to it. A data dump can not exist without being attached to a contract, so it is a weak entity. In contrast, the video stream is not a weak entity. A video stream is produced by a DataScoop, which can have up to 100 individual video streams. Gold subscribers have the ability to control one video stream, so it is necessary to have a method of identifying individual video streams. As a result, they can not be a weak entity.

The employee entity has a unary relationship with itself. This is to enable employees to be managed by other employees. The employee also has several specialisations which have different relationships with other entities. These specialisations are overlapping and are not mandatory. This makes the database flexible by allowing an employee to exist without having a specialised role. It also allows employees to take on multiple roles, so a drone pilot could also act as a drone technician as an example.

The biome entity acts as a lookup table. This ensures biomes remain consistent within DataScoops and contracts.

The part type table is a reference table that holds information about a type of drone part. This table is referenced by the part instance table which represents a specific instantiation of a part. This means information on a part can be stored in a single place and does not need to be duplicated for every instance of a part, ensuring data integrity. Part instances can partake in a relationship with the DataScoops entity. This provides an easy way to show what parts are within a DataScoop instance. This relationship is not mandatory for the part instance entity, allow parts to exist without being part of a DataScoop.

## Application of Data Management Principles

This model defines a specific structure for a data model. Every entity has all the required attributes defined; each required relationship is shown. This will support data quality, ensuing data is complete and consistent and can be used to fulfil its purpose. This model is designed with a data governance approach in mind to facilitate the company policies of FlightStream as defined in the project brief. This model implements the aspects of FlightStream’s muti-tiered subscription model into the database design through relationships and specialisations of with/of the subscription entity. This model also shows the sensitive information stored in the database, revealing data privacy needs. The database will store personal identifiable information of both FlightStream customers and employees, along with vast amounts of data collected from DataScoops that can belong to a customer in some cases (Super Platinum contracts own the rights to data collected for a contract). This conceptual model also shows which entities will need to communicate with each other -administrative executives need to access contract entities- and which entities shouldn’t have anything to do with one another -salespersons don’t need to know anything about drone parts-. This conceptual model provides a basis to develop lifecycle management strategies by showing how data will be stored, and how data from different entities can interact, informing how it can be used. The relationships in this model show how deleting data from entities will affect other entities, informing on requirements of lifecycle management. This conceptual model also reveals the metadata for system structures and relationships, and defines candidate keys. This will provide the information needed to develop comprehensive metadata management strategies in the future.

# Data Dictionary

## Entities

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity Name** | **Description** | **Aliases** | **Occurrence** |
| DataScoop | A DataScoop is a drone that is used to collect data from geographic areas and stream video. DataScoops are controlled remotely by a FlightStream employee. DataScoops can configured different types of geographical regions (biomes). | Drone | A DataScoop is purchased by FlightStream or assembled from parts provided by external suppliers. |
| Part instance | A mechanical component of a DataScoop. DataScoop drones are comprised of mechanical parts that can be replaced or modified. A part instance represents a physical part that is either installed in a DataScoop or possessed by FlightStream. | Drone part, part, drone component. | Drone parts are purchased from external suppliers. |
| Part type | A lookup table for all the different types are parts used by DataScoops. Does not represent a specific part instance. | Part list, parts lookup, component list. | A part type is created for each different type of drone component used in FlightStream’s DataScoops |
| Supplier | A third party that manufactures or supplies drone parts used by FlightStream. | Part supplier | Any suppliers that currently or have previously supplied drone parts are recorded by FlightStream. |
| Employee | A person employed by FlightStream. Employees can have specialised roles. |  | FlightStream employs individuals to work for the company. |
| Drone Technician | A specialisation of Employee. Drone technicians are responsible for inspecting and replacing drone parts, as well as changing drone biome configurations. | DataScoop Technician | FlightStream employs individuals to maintain and configure DataScoops. |
| Drone Pilot | A specialisation of employee. Drone pilots control DataScoops, piloting them through a mobile application. One pilot can control multiple drones. | Pilot, DataScoop Pilot. | FlightStream employs individuals to fly their DataScoops |
| Salesperson | A specialisation of employee. Salespersons attempt to sell contracts to customers or attract new customers. Salespersons can apply a 3% discount to subscriptions. |  | FlightStream employs individuals to sell subscriptions. |
| Administrative Executive | A specialisation of employee. Administrative executives can change the prices of subscriptions and enter contract details. |  | FlightStream employs individuals to manage subscriptions and contracts. |
| Customer | An individual registered with FlightStream as a consumer or potential consumer of FlightStream’s services. | Subscriber | Individuals who wish to use FlightStream’s services can register for a customer account online or through a FlightStream employee. |
| Subscription | Customers can subscribe to FlightStream contracts. These agreements are itemised as subscriptions. Subscriptions allows employees to access data and video streams from contracts. There are several types of subscriptions that provide additional benefits, modelled as specialisations. |  | Subscriptions are purchased by customers. |
| Gold | A specialisation of subscription. Gold subscriptions allow customers to control one of the streams from the DataScoop. | Gold Subscriptions | Gold subscriptions can be purchased by a customer. |
| Platinum | A specialisation of subscription. Platinum subscriptions allow customers to control a video stream from the DataScoop and grant exclusive ownership of data collected by the DataScoop. | Platinum Subscriptions | Platinum subscriptions can be purchased by a customer. |
| Super Platinum | A specialisation of subscription. Super Platinum subscriptions allow customers to control a video stream and grant exclusive rights and access to video streams and data. | Super Platinum Subscriptions | Super Platinum subscriptions are negotiated with FlightStream employees. |
| Subscription type | A lookup table that contains some information on subscriptions such as price. | Subscription lookup, Subscriptions list | Subscriptions models are designed by FlightStream. |
| Contract | A geographical area designating for DataScoop surveying. Contracts can be surveyed by one or many DataScoops. | Survey area, Contract area. | Contracts are designed by FlightStream alone or in conjunction with Customers. |
| Biome | A list of biomes DataScoops can be configured for. A biome is a type of geographical region that has specific climate conditions that effect the use of DataScoops. DataScoops need to be configured in different ways for different biomes. | Regional zone | Fight Stream has a static list of biome types that require specific DataScoop configuration. |
| Video Stream | A 3D stereographic video streamed live from a DataScoop in operation. Each data scoop has 100 individual video streams | Live video, video. | When in operation a DataScoop will create live video streams which can be viewed on the FlightStream website. |
| Data dump | A weak entity attached to a contract. The data dump contains all the data (not including video) transmitted from DataScoops in a contract area. |  | When a contract is made, a data dump is created to store data from DataScoops |

## Relationships

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Entity Name** | **Cardinality** | **Participation** | **Relationship** | **Participation** | **Cardinality** | **Entity Name** |
| Supplier | N | Optional | Supplies | Optional | M | Part type |
| Part instance | N | Mandatory | References | Optional | 1 | Part type |
| DataScoop | 1 | Mandatory | Comprises | Optional | M | Part instance |
| Drone Technician | M | Optional | Inspects (part instance)  Relationship attributes:   * “date”, the date of the inspection, date, not null, * “Inspection report”, the results of the inspection, string, not null. | Optional | N | Part instance |
| DataScoop | M | Mandatory | Is configured for | Optional | 1 | Biome |
| DataScoop | N | Optional | Populates | Optional | M | Data dump |
| DataScoop | 1 | Optional | Streams | Mandatory | M | Video stream |
| Done Pilot | 1 | Optional | Pilots | Mandatory | M | DataScoop |
| Administrative Executive | 1 | Optional | Creates | Mandatory | M | Contract |
| Salesperson | 1 | Optional | Sell | Optional | M | Subscription |
| Employee | 1 | Optional | Manages | Mandatory | M | Employee |
| Contract | M | Mandatory | Contains | Optional | N | Biomes |
| Contract | 1 | Mandatory | Has (identifying relationship) | Mandatory | 1 | Data dump |
| Contract | 1 | Optional | Has | Mandatory | M | Video stream |
| Subscription | M | Mandatory | References | Optional | 1 | Subscription type |
| Subscription | M | Mandatory | Accesses | Optional | 1 | Contract |
| Subscription | M | Mandatory | Accesses | Optional | N | Video stream |
| Subscription | M | Mandatory | Accesses | Optional | 1 | Data dump |
| Platinum | 1 | Mandatory | Owns | Optional | 1 | Data dump |
| Platinum | M | Optional | Controls | Optional | 1 | Video stream |
| Super platinum | 1 | Mandatory | Owns exclusively | Optional | 1 | Data dump |
| Super platinum | 1 | Mandatory | Owns exclusively | Optional | M | Video stream |
| Super Platinum | 1 | Optional | Controls | Optional | 1 | Video stream |
| Gold | M | Optional | Controls | Optional | 1 | Video stream |
| Customer | 1 | Optional | Purchases | Mandatory | M | subscription |

## Attributes

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity Name** | **Attribute Name** | **Description** | **Domain** | **Aliases** | **Composite** | **Derived** | **Nullable?** | **Key?** | **Default Value** |
| DataScoop | ID | A unique identifier | integer |  |  |  | NO | YES | Autoincrementing |
| Last service | When the DataScoop was last serviced | date | Last inspected, serviced |  |  | YES | NO | null |
| Is deployed | States if the DataScoop is currently in use | boolean | In use, active |  |  | NO | NO | False |
| Location | The DataScoops current GPS coordinates and altitude. If the DataScoop is not in use, the value will be null. | [decimal, decimal, decimal] | Position | Composed of latitude, longitude, and altitude. |  | YES | NO | null |
| Part instance | ID | A unique identifier | integer |  |  |  | NO | YES | Autoincrementing |
| Date installed | The date when the part was installed in the DataScoop | date |  |  |  | NO | NO |  |
| Condition | The physical condition of the part (worn out, working, etc..) | string | State |  |  | NO | NO |  |
| Part type | ID | A unique identifier | string (e.g. CY255K) |  |  |  | NO | YES |  |
| Name | The name of the part | string |  |  |  | NO | NO |  |
| Description | A description of the part. | string |  |  |  | NO | NO |  |
| Supplier | ID | A unique identifier | integer |  |  |  | NO | YES | Autoincrementing |
| Name | The name of the company/individual | string | Manufacturer, company |  |  | NO | NO |  |
| Email | The supplier’s contact email address | string with regex mask | Contact email |  |  | YES | NO | null |
| Phone | The suppliers contact phone number | string with regex mask | Contact phone |  |  | YES | NO | null |
| Website | The supplier’s website | string | Webstore |  |  | YES | NO | null |
| Notes | Any notes on the suppler from FlightStream | string |  |  |  | YES | NO | null |
| Employee | ID | A unique identifier | integer |  |  |  | NO | YES | Autoincrementing |
| Name | The employee’s full name | string, string |  | Comprised of fist name and last name. |  | NO | NO |  |
| Phone | The employee’s work phone number | string with regex mask | Work phone |  |  | NO | NO |  |
| Email | The employee’s work email address | string with regex mask | Work email |  |  | NO | NO |  |
| Biome | Name | The name of the biome | string | Type |  |  | NO | YES |  |
| Description | Description of the requirements for operating in the biome. | String | Operational requirements |  |  | NO | NO |  |
| Contract | ID | A unique identifier | integer |  |  |  | NO | YES | Autoincrementing |
| Area | A list of GPS coordinates that defines the geographical area encompassed by the contract | [  [decimal, decimal],  [decimal, decimal],  . . .  ] |  | Comprised of a list of coordinate pairs of indeterminate length. |  | NO | NO |  |
| Data dump | Time | The exact time the data was recorded by the DataScoop | datetime |  |  |  | NO | NO | Current time |
| Location | The GPS location and altitude of the DataScoop when the data was recorded | [decimal, decimal, decimal] | Position | Comprised of latitude, longitude, and altitude |  | NO | NO |  |
| Spectral data | Organic spectral data from one litre of air | Exact datatype requirements are unclear, can store as a blob until datatype is determined | Organic spectral data | Unknown |  | YES | NO | null |
| Temperature | The ambient air temperature | decimal |  |  |  | YES | NO | null |
| Ambient light | The ambient light detected by the DataScoop | decimal | Light strength |  |  | YES | NO | null |
| Humidity | The ambient humidity | decimal |  |  |  | YES | NO | null |
| Video stream | ID | A unique identifier | integer |  |  |  | NO | YES | Autoincrementing |
| Direction | The pitch, roll, and yaw of the video feed | [decimal,  decimal,  decimal] | Angle | Comprised of pitch, roll and yaw. |  | NO | NO |  |
| Subscription | ID | A unique identifier | integer |  |  |  | NO | YES | Autoincrementing |
| Discount | A percentage off discount on the subscription price | decimal |  |  |  | NO | NO | 0.0 |
| Billing date | The date when the subscription is billed | date |  |  |  | NO | NO |  |
| Recurring | Whether the subscription will automatically renew. | boolean |  |  |  | NO | NO | False |
| Subscription type | Name | The name of the subscription model | string | Type |  |  | NO | YES |  |
| Price | The monthly cost of the subscription | decimal | Cost |  |  | NO | NO |  |
| Description | A description of the subscription’s benefits | string | Benefits |  |  | NO | NO |  |
| Customer | ID | A unique identifier | integer |  |  |  | NO | YES | Autoincrementing |
| Name | The customer’s full name | string, string |  | Comprised of fist name and last name. |  | NO | NO |  |
| Phone | The customer’s contact phone number | string with regex mask | Contact phone |  |  | YES | NO |  |
| Email | The customer’s contact email address | string with regex mask | Contact email |  |  | NO | NO |  |
| Company | The customer’s company, if applicable. | string |  |  |  | YES | NO | null |

## Rationale

For rational, refer to section 3.1. The same rationale applies for this section.

## Review of Data Management Procedures

Data dictionaries apply a high degree of data quality management by defining the datatypes used for each attribute. The rigid definition of attributes from data dictionaries eliminates the possibility of entering erroneous data into a database, enforcing data quality. Data dictionaries also provide data governance. A description of the purpose of an attribute and details on the attribute’s composition or derivation helps to ensure data complies with company policy. Descriptions of relationships and entities ensure data is easily useable and governable. Data dictionaries do not directly contribute to privacy and security. Like conceptual models though, they can be used to analyse what data is being stored in a system in order to highlight any privacy concerns and security needs. Data dictionaries help with data lifecycle management by defining exactly how data is stored and how it interacts with other data. The data dictionary also shows how data is created by defining occurrences for entities. A description of attributes also reveals how data can be used, assisting with lifecycle management. Data dictionaries define the metadata for database attributes, relationships and entities. Providing exact definitions for metadata makes data dictionaries an invaluable tool for metadata management.

# Assumptions on Business Rules

For this system, it is assumed that providing data from contracts to customer is handled by an external system, this results in not having a relationship between customer and contract directly. This is a reasonable assumption to make as there is no direct requirements to the customer entity to modify data in the contract, video stream, or data dump entities.

It is assumed that a contract relates to a specific geographic location. It is assumed that multiple subscriptions can be related to a contract unless the contract is negotiated as a super platinum contract. This is implied by the ability of multiple standard user to access a contract.

It is assumed that a super platinum contract can only be accessed by a single subscription. This is based off the fact that a super platinum subscriber owns the exclusive access rights to the data and video streams of a given contract.

It is assumed that FlightStream will have employee roles for piloting and maintaining DataScoop drones. This enables the creation of employee specialisations allowing control over which employees can participate in certain relationships.

It is assumed that “extreme cold – Ice and Snow” is meant to refer to a single biome, and not two distinct biomes due to use of a hyphen instead of a comma.

It is assumed that organic spectral data from one litre of air can be stored in a single attribute. What exactly this data entails is not defined, so this is assumed to be a single blob of data.

Platinum subscribers have exclusive rights to the video streams from a contract, however, it is later stated that only super platinum subscribers have exclusive *access* to video streams. This implies that other subscribers can view the video streams from a contract negotiated by a platinum subscriber, but do not own the video.

It is assumed that video streams are not stored, and it is up to the subscriber to record video streams themselves if they want.

Milestone 2: Logical Modelling

# Conceptual to Relational Logical Model

## Methodology

Converting the conceptual model to a logical model

* Each entity in the conceptual model becomes a relation in the logical model.
* Each attribute becomes an entity in the new relation.
  + Composite attributes are split into their constituent parts; a new attribute for each part.
  + Multivalued attributes are extracted into a new relation which will have a many to one relationship with the original.
  + Candidate keys may become a primary key if they fit the criteria of the relational model. A primary key must be minimal, meaningful, and stable.
* One to one relationships:
  + A foreign key is placed in one relation that references the primary key of the other.
* One to many relationships:
  + A foreign key is placed in the relation with the ‘many’ cardinality, referencing the primary key form the relation with the ‘one’ cardinality.
* Many to many relationships:
  + A many to many relationship cannot be model in a relational diagram. Instead, a joining relation must be used.
  + The joining relation will be placed between the two relations in the many to many relationship and will have a one to many relationship with reach.
  + The joining table will contain foreign keys from each relation that reference the primary keys of the relations.
  + The foreign keys will form a composite primary key for the joining relation.
* Relationships with attributes:
  + A joining table can be used to model relationships with attributes.
  + The same procedure can be used as modelling many to many relationships with the addition of the relationship attribute in the joining table.
* Superclass and subclasseses:
  + One relation will be created for the superclass that contains all common attributes.
  + Each subclass will be converted into a unique relation that contains all attributes unique to itself.
  + Subclass relations will be connected to the superclass in one to one relationships.
  + Each subclass will contain a foreign key that references the superclass. This foreign key will also function as the primary key for the subclass.
* Weak entities
  + Weak entities will not have a unique primary key. Instead they will have a foreign key the references their identifying relation.

## Description of Normalisation Forms

Normalisation is the process of transforming a relational database model into a normalised form. Normalising a database helps reduce redundancy, which can cause issues in a database. There are several forms of normalisation. Each form builds off if the previous form with additional rules for data storage.

First normal form (1NF) states that each cell in a table/relation should contain only one piece of data, and each column must have a unique name. 1NF form help simplify queries by eliminating data duplication. To normalise a relation to 1NF, any records that contain cells with more than one value should be split into separate records for each value.

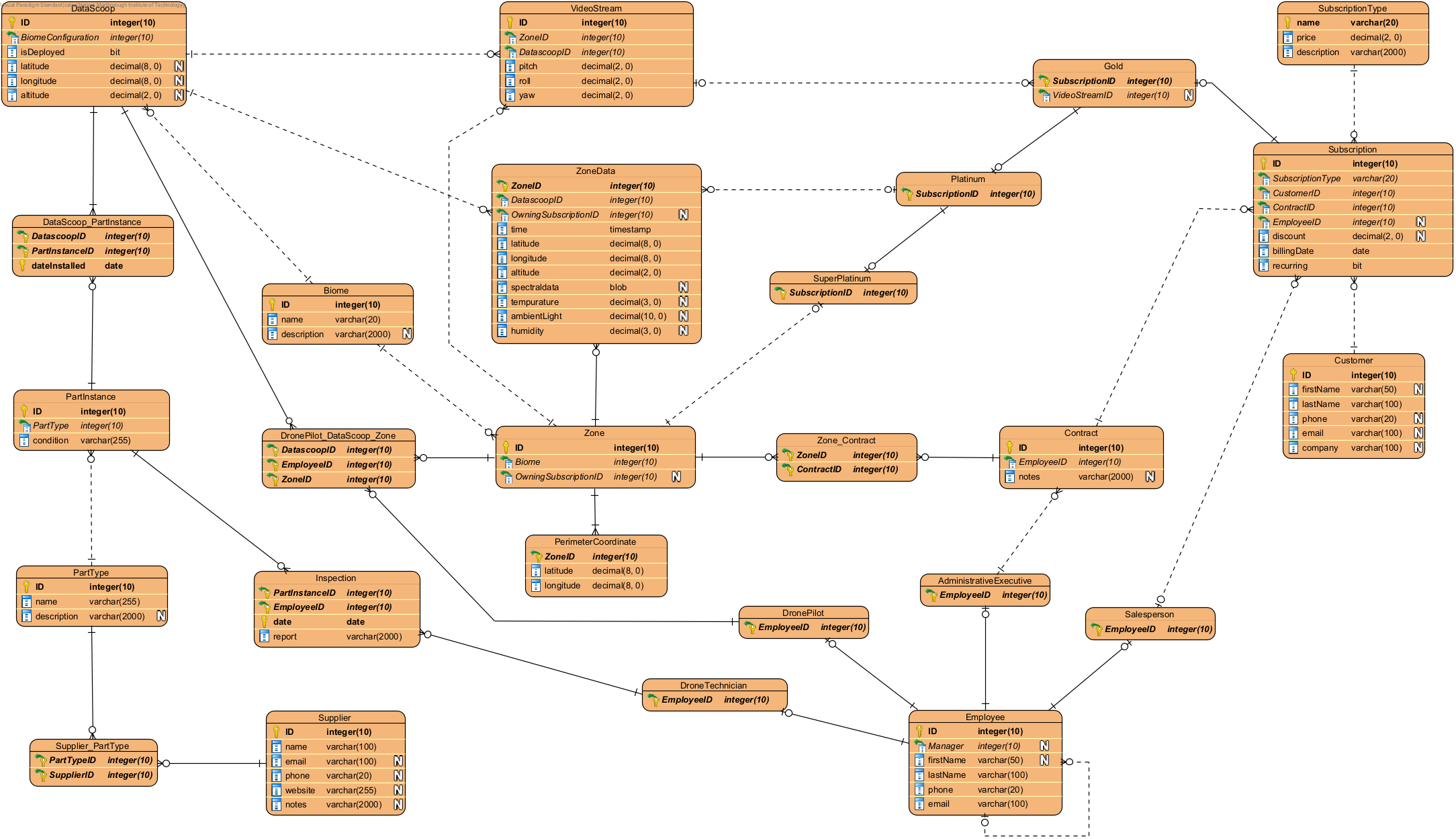
Second normal form (2NF) requires every attribute in a relation to be dependent on the primary key. In 2NF the value of each attribute can be determined by the primary key. 2NF removes redundant data from a relation. To normalise a relation to 2NF, any attributes that are dependent on an attribute other than the primary key should be moved to a new relation. The new relation can then be linked to the original relation with a foreign key.

Third normal form (3NF) addresses transitive dependencies. A transitive dependency is where an attribute is not directly dependent on the primary key, but instead is dependent on an intermediary attribute. As an example, imagine attribute X is dependent on the primary key, and attribute Y is dependent on attribute X. In this example, while attribute Y is linked to the primary key through attribute X, it is not directly related to the primary key, and so exists as a transitive dependency. 3NF requires transitive dependencies to be taken out of a relation. After a transitively dependent attribute is extracted into a new relation, it can be linked to the original relation with a foreign key (GeeksforGeeks, 2023).

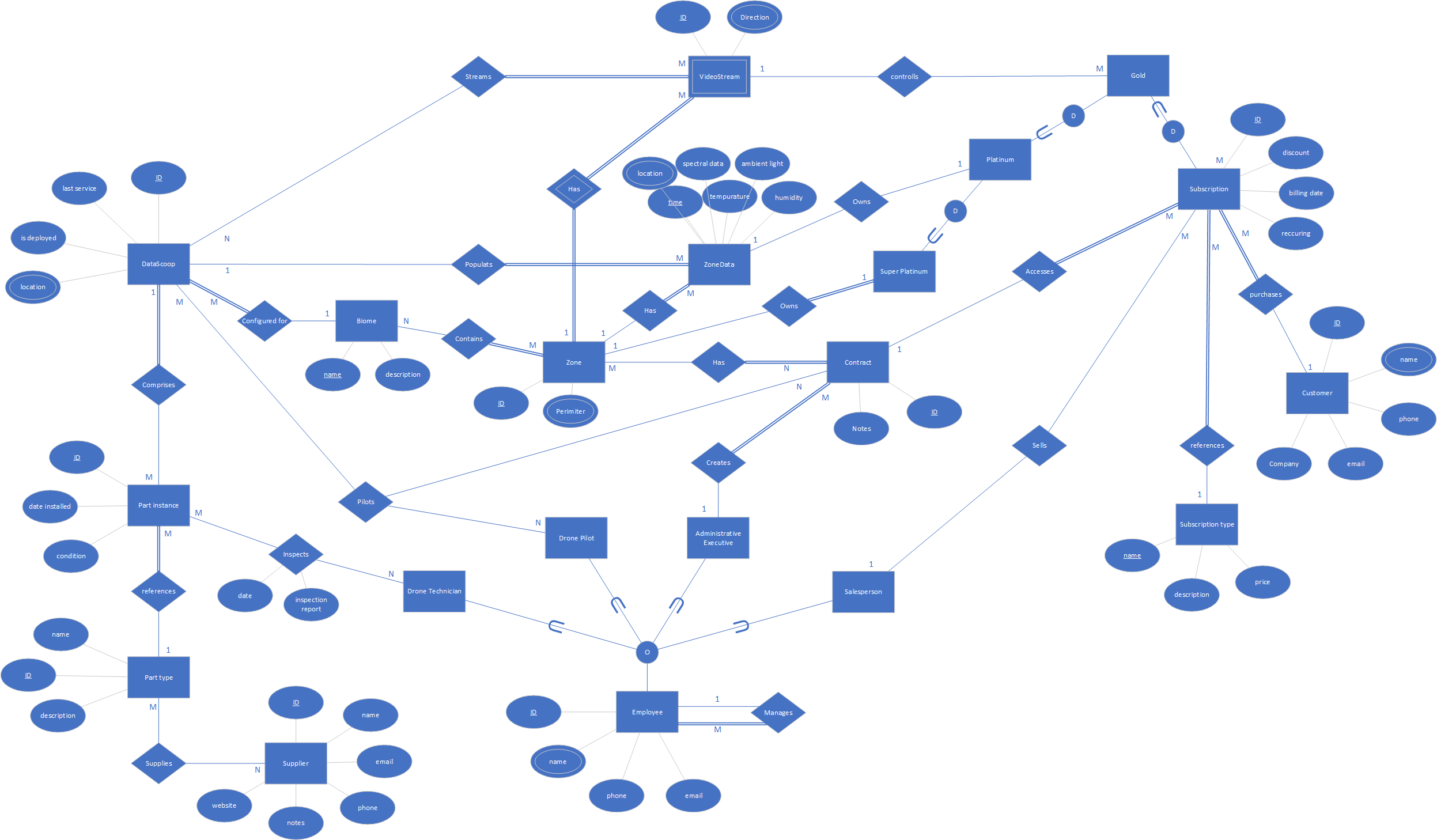
Boyce-Codd normal form (BCNF) can be applied to 3NF in some cases where 3NF fails to completely eliminate data redundancy. BCNF requires attributes in a functional dependency to be candidate keys. In some cases, a 3NF relation contains functional dependencies where A is dependant on B, but A is not the candidate key of the relation. BCNF removes these dependencies by separating the non-candidate dependents into a new relation. The new relation can be linked with the original relation with a foreign key if required (Jain, 2022).

Fourth normal form (4NF) requires a relation to have no non-trivial multivalued dependencies (excluding the primary key). To have a multivalued dependency a relation must have at least three attributes where multiple values of attributes B and C are dependent on a single value of attribute A, but B and C are not related. In this situation, a single value of A can appear multiple times for each dependency from a different value of B. The same is true for each value of C, which can cause issues as it forces a link between the values of B and C which are not related. To fix this, 4NF requires the relation to be split into two relations. One that contains attributes A and B, and another with attributes A and C. This allows the dependencies between A and B, and A and C to exist independent of one another without forcing a link between B and C (JavaTPoint, retrieved 2024).

## Logical Model



## Revised Conceptual Model



### Rationale

DataScoop

The location attribute was separated into latitude, longitude, and altitude. This is to comply with the requirements of first normal form which requires only one value per cell. The date of last service attribute was removed. This eliminates data redundancy as the date of last service is already listed for each DataScoop part in the Inspection relation. A foreign key referencing the Biome relation creates the many to one relationship with Biome.

DataScoop\_PartInstance

This joining relation was created to model the many to many relationship between DataScoop and PartInstance. It contains the composite primary attribute dateInstalled which allows a PartInstance to be installed in a DataScoop on different dates.

PartInstance

The dateInstalled attribute has been moved to the DataScoop\_PartInstance joining relation where it is part of a composite key. The dateInstalled attribute is only partially dependent on Part Instance’s primary key, so violates second normal form if it stays in the PartInstance relation. A foreign key has been added to model the relationship with PartType.

PartType

No changes needed.

Supplier\_PartType

This joining relation was added to model the many to many relationship between Supplier and PartType

Supplier

No changes needed.

Biome

No Changes necessary.

DronePilot\_DataScoop\_Zone

This joining relation was added to model the many to many relationship between DronePilot and DataScoop. Zone has been added to the relationship to give context on where a DataScoop is being flown.

Inspection

This joining relation was added to model the many to many relationship between PartInstance and DroneTechnician. This relation contains the relationship attributes date, and report from the conceptual model. The date attribute is part of the primary key, allowing the same DroneTechnician to inspect the same PartInstance multiple times.

VideoStream

The direction attribute has been separated into pitch, roll, and yaw to comply with first normal form. The foreign keys ZoneID, and DataScoopID have been added to model the relationships with Zone and DataScoop respectively.

ZoneData

This relation was previously known as DataDump. The original setup for this entity was incorrect, the original model envisioned a new instance of DataDump for each record in Contract. This does not comply with any form of normalisation and was due to a misunderstanding of weak entities on the part of the designer. This has been remedied with a new name, ZoneData, which better represents its new form in a normalised relational model. This entity has been properly remodelled as a weak entity with a primary key that is also a foreign key referencing Zone. This allows each record to be attached to a specific zone and prevents data from existing independent of a zone, eliminating data redundancy. The attribute, location, has been separated into its constituent parts to comply with first normal form. The foreign keys, DataScoopID and OwningSubscriptionID have been added to model the relationships with DataScoop and Platinum respectively.

Zone

This entity was missing from the conceptual design. It is required to meet the project brief which states one contract can have many zones each with different biomes. In the original conceptual model, a contract can only have one biome. The area attribute of contract which is a multivalued attribute of composite values was moved to Zone, separated into its latitude and longitude to comply with first normal form, and then extracted into a new relation to comply with third normal form. Zone contains the foreign key Biome to make a many to one relationship with the Biome relation. The foreign key OwningSubscriptionID creates the relationship with SuperPlatinum.

PerimiterCoordinate

This relation contains latitude and longitude coordinates that define a point along the perimeter of a zone. This is a weak entity that is identified by Zone. Together, multiple records can be used to define a perimeter of a Zone. This relation was created to comply with first and third normal forms which state a cell can only contain one value, and attributes must not be transitively dependent on a primary key.

Zone\_Contract  
This joining relation was created to model the many to many relationship between contract and the new Zone relation.

Contract

The area attribute has been transferred into a new relation attached to the new Zone relation. An EmployeeID foreign key has been added for the relationship with AdministrativeExecutive.

Employee

This relation models the employee superclass from the conceptual model. It contains all the common attributes for each subclass to prevent data duplication. The self-referencing foreign key, manager, models the recursive relationship Employee has with itself.

DroneTechnician, DronePilot, AdministrativeExecutive, and Salesperson

These relations model the subclasses of Employee. Each would contain the unique attributes for each subclass if there were any. These could be added to each subclass if requested by FlightStream. These subclasses allow different employee types to have specialised relationships with other relations. This allows relationships to be not nullable, which would not be possible with subclassing. Each subclass has a foreign primary key that references their parent record in Employee.

Customer

No changes required.

SubscriptionType

No changes required.

Subscription

This relation models the subscription superclass from the conceptual model. It contains the common attributes for each subclass to prevent data duplication. The subclass structure has been remapped into a less complicated structure. Each subclass now inherits in a linear hierarchy, removing the need for the complex relationship web in the conceptual mode. The foreign keys SubscriptionType, CustomerID, ContractID, and EmployeeID have been added to model relationships with the respective entities.

Gold

A foreign key, VideoStreamID, has been added for the relationship with VideoStream

Platinum

This subclass now inherits for Gold instead of from subscription. This means the relationship for controlling the VideoStream does not need to be duplicated. Using this method should significantly improve data quality be removing redundant relationships.

SuperPlatinum

This subclass now inherits from Platinum for the reasons listed above. The relationship to DataDump has been replaced with a relationship with Zone now that DataDump has been restructured.

# Data Dictionary

## 7.1 Relations

|  |  |  |  |
| --- | --- | --- | --- |
| **Relation Name** | **Start Volume** | **Yearly Growth** | **Comments** |
| DataScoop | 50,000 | 1% | Assuming FlighStream continues to operate in existing zones, no new DataScoops will be needed except in the case of replacing old or broken DataScoops. |
| PartInstance | 500,000 | 10% | “Each DataScoop has a large number of parts” Approximating 10 parts per DataScoop will give 500,000 parts. Parts will need to be replaced much more often than a full DataScoop, so will grow at a faster rate. |
| DataScoop\_PartInstance | 500,000 | 10% | Joining table between DataScoop and Part\_partinstance. Will require one record for each part instance that has been installed in a DataScoop at a given time. Not all Partinstaces will be installed in a DataScoop, some may be installed in multiple DataScoops across their lifetime. To simplify, the number of records will be approximately equal to Partinstance. |
| PartType | 70 | No Growth | Look up table for types of parts. Assuming a DataScoop has 10 parts with 7 variations for different biomes. In actuality there will be parts that can be used across multiple biomes so this will likely be smaller. |
| Supplier | 5 | No Growth | There is no data provided to estimate how many suppliers FlightStream uses. They will likely not need more than 5 to supply their needed parts. |
| Supplier\_PartType | 200 | No Growth | Assuming each supplier will supply multiple parts type, but not all types. |
| Biome | 7 | No Growth | FlightStream has specified a fixed selection of available biomes. |
| Employee | 6,110 | 50% | Assuming a pilot can handle 10 DataScoops at once, 50,000 deployed DataScoops will require 5,000 pilots. To upkeep 50,000 DataScoops at least 500 technicians will be required. To manage 500,000 subscribers at least 500 salespeople will be required. Creating 2,000, contracts will need at least 10 administrative executives. An additional 100 employees added to include employees not in a specialised role.  Assuming a 50% yearly turnover rate of staff. |
| DronePilot | 5000 | Proportional to emplyees | See above |
| DroneTechnicion | 500 | Proportional to emplyees | See above |
| Salseperson | 500 | Proportional to emplyees | See above |
| AdministrativeExecutive | 10 | Proportional to emplyees | See above |
| Inspection | 500,000 | 20% | Every PartInstance must be inspected every 5 years. |
| Contract | 2000 | 0.5% | Assuming 100 new contracts each year. |
| DronePilot\_DataScoop\_ Zone | 50,000 | 5% | 50,000 deployed data scoop will require a minimum of 50,000 records. |
| Zone | 100 | 1% | Assuming one new zone each year. |
| PerimeterCoordinates | 2000 | 1% | Assuming at takes 20 coordinates to map a zone perimeter. |
| VideoStream | 5,000,000 | 1% | 100 video streams per DataScoop. |
| ZoneData | 7 billion | Proportional to zones | 50,000 DataScoops over 100 zones is 500 per zone. Each data scoop sensing data every second for 100 zones would give > 1 billion records in 8 hours. There is no data on how long FlightStream spends collecting data in a zone. Assuming each zone takes 1 work week to survey (40 hours) there would 7 billion records of data from all 100 zones. |
| Customer | 501,000 | 0.2% | Assuming 1000 new customers per year with 1,000 customers not subscribed to a contract. |
| Subscription | 500,000 | 0.2% | Assuming 1000 new subscription per year. |
| SubscriptionType | 4 | No growth | FlightStream has four subscription types. |
| Gold | 5,000 | Proportional to Subscription | “1% of subscribers are gold subscribers”. |
| Platinum | 2,500 | Proportional to Subscription | “0.5% are platinum subscribers”. |
| SuperPlatinum | 100 | Proportional to Subscription | “100 contracts are exclusive ‘Super Platinum’”. |

## 7.2. Attributes

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Relation Name** | **Attribute** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Null** | **Key** | **References Entity** | **Integrity Constraints** |
| DataScoop | ID | Unique identifier | integer | 10 | >0 | Unique | Autonumber | NO | Primary |  |  |
| Biome  Configuration | The biome the DataScoop is configured for | integer | 10 | 1 to 7 |  |  | NO | Foreign | Biome | Must reference valid Biome record |
| isDepolyed | Whether the drone is currently deployed. | bit |  | 0 or 1 |  | 0 | NO |  |  |  |
| latitude | The drones geographical latitude | decimal | 8,0 | -90 to 90 | Must be valid coordinate | 0 | YES |  |  |  |
| longitude | The drones geographical longitude | decimal | 8,0 | 0 to 360 | Must be valid coordinate | 0 | YES |  |  |  |
| altitude | The drones height above sea level. | decimal | 2,0 |  |  | 0 | YES |  |  |  |
| DataScoop\_  Part  Instance | DataScoopID | The DataScoop the part instance is installed in | integer | 10 | >0 |  |  | NO | Composite Primary  and Foreign | DataScoop | Must reference valid DataScoop record |
| PartInstanceID | The Partinstance installed | integer | 10 | >0 |  |  | NO | Composite Primary  and Foreign | PartInstance | Must reference valid PartInstance record |
| dateInstalled | The date the part instance was installed in the DataScoop | date |  |  | Must be valid date |  | NO | Composite Primary |  |  |
| Part  Instance | ID | Unique identifier | integer | 10 | >0 | Unique | Autonumber | NO | Primary |  |  |
| PartType | The type of part the part instance is | integer | 10 | >0 |  |  | NO |  |  | Must reference valid partType record |
| condition | The condition the part instance is in. | varchar | 255 |  |  |  | NO |  |  |  |
| PartType | ID | Unique identifier | integer | 10 | >0 | Unique | Autonumber | NO | Primary |  |  |
| name | The name of the part | varchar | 255 |  |  |  | NO |  |  |  |
| description | A description of the part | varchar | 2000 |  |  |  | NO |  |  |  |
| Supplier\_  PartType | PartTypeID | The part type supplied by the specified supplier. | integer | 10 | >0 |  |  | NO | Composite Primary and Foreign | PartType | Must reference valid partType record |
| SupplierID | The supplier that supplies the specified part type. | integer | 10 | >0 |  |  | NO | Composite Primary and Foreign | Supplier | Must reference valid supplier record |
| Supplier | ID | Unique identifier | integer | 10 | >0 | Unique | Autonumber | NO | Primary |  |  |
| name | The company name of the supplier | varchar | 100 |  |  |  | NO |  |  |  |
| email | The supplier’s email. | varchar | 100 |  | Must be a valid email |  | YES |  |  |  |
| phone | the supplier’s phone number. | varchar | 20 |  | Must be a valid phone number |  | YES |  |  |  |
| website | The supplier’s website. | varchar | 255 |  | Must be a valid website |  | YES |  |  |  |
| notes | Any notes on the supplier | varchar | 2000 |  |  |  | YES |  |  |  |
| Biome | ID | Unique identifier | integer | 10 | >0 | Unique | Autonumber | NO | Primary |  |  |
| name | The name of the biome | varchar | 20 |  |  |  | NO |  |  |  |
| description | The required drone configurations for the biome | varchar | 2000 |  |  |  | NO |  |  |  |
| DronePilot-  DataScoop\_  Zone | DataScoopID | The piloted DataScoop for the contract. | integer | 10 | >0 |  |  | NO | Composite Primary and Foreign | DataScoop | Must reference a valid DataScoop record. |
| EmployeeID | The pilot for the DataScoop for the contract. | integer | 10 | >0 |  |  | NO | Composite Primary and Foreign | DronePilot | Must reference a valid DronePilot record. |
| ZoneID | The zone of the piloted DataScoop | integer | 10 | >0 |  |  | NO | Composite Primary and Foreign | Zone | Must reference a valid Zone record. |
| Inspection | PartInstanceID | The inspected part. | integer | 10 | >0 |  |  | NO | Composite Primary and Foreign | PartInstance | Must reference a valid PartInstance record. |
| EmployeeID | The employee carrying out the inspection. | integer | 10 | >0 |  |  | NO | Composite Primary and Foreign | DroneTechnician | Must reference a valid DroneTechnician record. |
| date | The date the inspection happened | date |  |  | Must be valid date |  | NO | Composite Primary |  |  |
| report | the results of the inspection | varchar | 2000 |  |  |  | NO |  |  |  |
| Contract | ID | Unique identifier | integer | 10 | >0 | Unique | Autonumber | NO | Primary |  |  |
| EmployeeID | The employee who created the contract. | integer | 10 | >0 |  |  | NO | Foreign | Administrative  Executive | Must reference a valid Administrative  Executive record. |
| notes | Any notes on the contract. | varchar | 2000 |  |  |  | YES |  |  |  |
| Zone  Data | ZoneID | Unique identifier and reference to associated Zone. | integer | 10 | >0 |  |  | NO | Primary and Foreign | Zone | Must reference a valid Zone record. |
| DataScoopID | The DataScoop that provided the data. | integer | 10 | >0 |  |  | NO | Foreign | DataScoop | Must reference a valid DataScoop record. |
| Owning  SubscriptionID | The subscription that owns the Zonedata. | integer | 10 | >0 |  |  | YES | Foreign | Platinum | Must reference a valid Platinum record. |
| time | The time the record was recorded by the DataScoop. | timestamp |  |  | Must be valid timestamp |  | NO |  |  |  |
| latitude | The DataScoop’s latitude when the data was recorded. | decimal | 8,0 | -90 to 90 | Must be valid coordinate | 0 | NO |  |  |  |
| longitude | The DataScoop’s longitude when the data was recorded. | decimal | 8,0 | 0 to 360 | Must be valid coordinate | 0 | NO |  |  |  |
| altitude | The DataScoop’s altitude when the data was recorded. | decimal | 2,0 |  |  | 0 | NO |  |  |  |
| spectralData | Organic spectral data from one litre of air. | blob |  |  |  |  | YES |  |  |  |
| temperature | The ambient air temperature. | decimal | 3,0 |  |  |  | YES |  |  |  |
| ambientLight | The ambient light. | decimal | 10,0 |  |  |  | YES |  |  |  |
| humidity | The ambient air humidity | decimal | 3,0 |  |  |  | YES |  |  |  |
| Video  Stream | ID | Unique identifier | integer | 10 | >0 | Unique | Autonumber | NO | Primary |  |  |
| ZoneID | The zone the video stream is for | integer | 10 | >0 |  |  | NO | Primary and Foreign | Zone | Must reference a valid Contract zone. |
| DataScoopID | The DataScoop that provides the video stream. | integer | 10 | >0 |  |  | NO | Foreign | DataScoop | Must reference a valid DataScoop record. |
| Pitch | The video stream’s pitch (vertical tilt). | decimal | 2,0 | -90 to 90 |  | 0 | NO |  |  |  |
| Roll | The video stream’s roll (rotation). | decimal | 2,0 | -180 to 180 |  | 0 | NO |  |  |  |
| Pitch | The video stream’s yaw (horizontal tilt). | decimal | 2,0 | -90 to 90 |  | 0 | NO |  |  |  |
| Zone\_  Contract | ZoneID | The Zone the contract is for. | integer | 10 | >0 |  |  | NO | Composite  Primary and Foreign | Zone | Must reference a valid Zone record. |
| ContractID | The contract the zone is for. | integer | 10 | >0 |  |  | NO | Composite Primary and Foreign | Contract | Must reference a valid Contract record. |
| Zone | ID | Unique identifier. | integer | 10 | >0 | Unique | Autonumber | NO | Primary |  |  |
| Biome | The Zone’s biome. | integer | 10 | 1 to 7 |  |  | NO | Foreign | Biome | Must reference a valid Biome record. |
|  | Owning  SubscriptionID | The subscription that owns the Zone. | integer | 10 | >0 |  |  | YES | Foreign | SuperPlatinum | Must reference a valid SuperPlatinum record. |
| Perimiter  Coordinate | ZoneID | The Zone the coordinate is for. | integer | 10 | >0 |  |  | NO | Primary and Foreign | Zone | Must reference a valid Zone record. |
| latitude | Geographical latitude. | decimal | 8,0 | -90 to 90 | Must be valid coordinate | 0 | NO |  |  |  |
| longitude | Geographical latitude. | decimal | 8,0 | 0 to 360 | Must be valid coordinate | 0 | NO |  |  |  |
| Employee | ID | Unique identifier. | integer | 10 | >0 | Unique | Autonumber | NO | Primary |  |  |
| Manager | The employee’s manager. | integer | 10 | >0 |  |  | YES | Foreign (recursive) | Employee | Must reference a valid employee record. |
| firstName | The employee’s first name. | varchar | 50 |  |  |  | YES |  |  |  |
| lastName | The employee’s last name. | varchar | 100 |  |  |  | NO |  |  |  |
| phone | The employee’s work phone number. | varchar | 20 |  | Must be a valid phone number |  | NO |  |  |  |
| email | The employee’s work email. | varchar | 100 |  | Must be a valid email. |  | NO |  |  |  |
| Drone  Technician | EmployeeID | The parent record of the specialisation. | integer | 10 | >0 | Unique |  | NO | Primary and foreign |  | Must reference a valid Employee record. |
| Drone  Pilot | EmployeeID | The parent record of the specialisation. | integer | 10 | >0 | Unique |  | NO | Primary and foreign |  | Must reference a valid Employee record. |
| Adminis  trative  Executive | EmployeeID | The parent record of the specialisation. | integer | 10 | >0 | Unique |  | NO | Primary and foreign |  | Must reference a valid Employee record. |
| Sales  person | EmployeeID | The parent record of the specialisation. | integer | 10 | >0 | Unique |  | NO | Primary and foreign |  | Must reference a valid Employee record. |
| Customer | ID | Unique identifier. | integer | 10 | >0 | Unique | Autonumber | NO | Primary |  |  |
| firstName | The customer’s first name. | varchar | 50 |  |  |  | YES |  |  |  |
| lastName | The customer’s last name. | varchar | 100 |  |  |  | NO |  |  |  |
| phone | The customer’s contact phone number. | varchar | 20 |  | Must be a valid phone number |  | YES |  |  | Can not be null if email is also null. |
| email | The customer’s contact email. | varchar | 100 |  | Must be a valid email. |  | YES |  |  | Can not be null if phone is also null. |
| company | the customer’s company | varchar | 100 |  |  |  | YES |  |  |  |
| Subscription | ID | Unique identifier. | integer | 10 | >0 | Unique | Autonumber | NO | Primary |  |  |
| Subscription  Type | The type of subscription. | integer | 10 | >0 |  |  | NO | Foriegn | Subscription  Type | Must reference a valid Subscription  Type record. |
| EmployeeID | The employee who sold the subscription. | integer | 10 | >0 |  |  | YES | Foreign | Salseperson | Must reference a valid Salesperson record. |
| ContractID | The contract the subscription is for. | integer | 10 | >0 |  |  | NO | Foreign | Contract | Must reference a valid Contract record. |
| CustomerID | The customer who purchased the subscription | integer | 10 | >0 |  |  | NO | Foreign | Customer | Must reference a valid Customer record. |
| discount | The discount to the subscription price | decimal | 2,0 | 0 to 0.03 |  | 0 | YES |  |  |  |
| billingDate | The date the subscription will be charged. | date |  |  | Must be a valid date. |  | NO |  |  |  |
| recurring | Whether the subscription will automatically renew. | bit |  | 0 or 1 |  |  | NO |  |  |  |
| Subscription  Type | Name | Unique identifier. | integer | 10 | >0 | Unique | Autonumber | NO | Primary |  |  |
| price | the price of the subscription. | decimal | 2,0 | >0 |  |  | NO |  |  |  |
| description | A description of the subscription | varchar | 2000 |  |  |  | NO |  |  |  |
| Gold | SubscriptionID | The parent record of the specialisation. | integer | 10 | >0 | Unique |  | NO | Primary and Foreign | Subscription | Must reference a valid subscription record. |
| VideoStreamID | The video stream the subscription can control. | integer | 10 | >0 |  |  | YES | Foreign | VideoStream | Must reference a valid VideoStream record. |
| Platinum | SubscriptionID | The parent record of the specialisation. | integer | 10 | >0 | Unique |  | NO | Primary and Foreign | Gold | Must reference a valid Gold record. |
| Super  Platinum | SubscriptionID | The parent record of the specialisation. | integer | 10 | >0 | Unique |  | NO | Primary and Foreign | Platinum | Must reference a valid Platinum record. |

## 7.3. Derived Attributes

There are no derived attributes in this design. An example of a derived attribute could be full name for customer or employee.

|  |  |  |
| --- | --- | --- |
| **Relation Name** | **Attribute** | **Derived from** |
| Customer | fullName | firstName + “ “ + lastName |

# NaLER Analysis

## Purpose and Use in Information Systems

NaLER or Natural Language method for interpreting Entity Relationship (ER) models is a technique for analysing and/or documenting ER models. In a NaLER analysis, an analyst will construct sentences that describe each attribute and relationship depicted in a model. These sentences conform to strict syntactic rules to accurately describe attributes and relationships. A NalER analysis of a model can provide an alternate way to interpret a relational design and reveal flaws in a model.

A NaLER analysis is an excellent tool for data quality management. It requires the designer to thoroughly consider every attribute and relationship in a model, revealing any flaws that could lead to data quality issues. A NaLER analysis is a useful tool for defining data governance strategies. It provides strict data storage rules in a natural language format that can be easily interpreted and used for defining governance policies. Although it does not directly contribute to security and privacy, it can help to analyse privacy and security requirements by showing which relations need to communicate with each other, and defining how potentially sensitive data will be stored. A NaLER analysis defines the second section of data lifecycle management (storage) in a natural language form. The NaLER analysis takes the data storage information from a diagram and converts into a linear readable format. Sentences populated with example data can show exactly how attributes and relationships are stored, providing useful insight when defining lifecycle management practices. The NaLER analysis does not provide as much information on metadata other modelling techniques, but it still provides useful information how attributes and relationships fit into the database structure.

## Diagram Conventions

The database diagram is a logical entity diagram depicting a normalised relational design.

The diagram depicts entities/relations using rounded rectangles. Entities have the relation name above a list of attributes displayed in rows. Each row contains a symbol indicating the type of attribute, the attributes name, datatype and length, and a letter N to represent nullability.

A close-up of a computer screen

Description automatically generated

|  |  |
| --- | --- |
| Symbol | Meaning |
|  | Primary Key |
|  | Foreign key |
|  | Primary and Foreign key |
|  | Non-Key attribute |
|  | Attribute is nullable |

A cut line with a white background

Description automatically generated with medium confidenceRelationships between entities are represented using crows foot notation. Crows foot notation links entities together with lines. Symbols at each end of a line depict cardinality for the adjoining entity. A solid line represents an identifying relationship, a dotted line represents a non-identifying relationship.

|  |  |
| --- | --- |
| Symbol | Meaning |
|  | Exactly one |
|  | Zero or one |
|  | One or more |
|  | Zero of more |

## Syntax Check

The diagram uses consistent syntax throughout, no adjustments needed.

## Entity Analysis

### Attributes

1. Each DataScoop is uniquely identified by one ID
2. One DataScoop (ID) must have one isDeployed
3. One DataScoop (ID) may have one latitude
4. One DataScoop (ID) may have one longitude
5. One DataScoop (ID) may have one altitude
6. Each DataScoop\_PartInstance is uniquely identified by DataScoopID, PartInstanceID, dateInstalled
7. Each PartInstance is uniquely identified by one ID
8. One PartInstance (ID) must have one condition
9. Each PartType is uniquely identified by one ID
10. One PartType (ID) must have one name
11. One PartType (ID) may have one description
12. Each Supplier\_PartType is uniquely identified by PartTypeID, SupplierID
13. Each Supplier is uniquely identified by one ID
14. One Supplier (ID) must have one name
15. One Supplier (ID) may have one email
16. One Supplier (ID) may have one phone
17. One Supplier (ID) may have one website
18. One Supplier (ID) may have one notes
19. Each Biome is uniquely identified by one ID
20. One Biome (ID) must have one name
21. One Biome (ID) must have one description
22. Each DronePilot\_DataScoop\_Zone is uniquely identified by DataScoopID, EmployeeID, ZoneID
23. Each Inspection is uniquely identified by PartInstanceID, EmployeeID, date
24. One Inspection (PartInstanceID, EmployeeID, date), must have one report.
25. Each VideoStream is uniquely identified by one ID
26. One VideoStream (ID) must have one pitch
27. One VideoStream (ID) must have one roll
28. One VideoStream (ID) must have one yaw
29. Each ZoneData is weakly identified by one ZoneID
30. Each ZoneData (ZoneID) must have one time
31. Each ZoneData (ZoneID) must have one latitude
32. Each ZoneData (ZoneID) must have one longitude
33. Each ZoneData (ZoneID) must have one altitude
34. Each ZoneData (ZoneID) may have one spectralData
35. Each ZoneData (ZoneID) may have one temperature
36. Each ZoneData (ZoneID) may have one ambientLight
37. Each ZoneData (ZoneID) may have one humidity
38. Each Zone is uniquely identified by one ID
39. Each PerimiterCoordinate is weakly identified by one ZoneID
40. One PerimiterCoordinate (ZoneID) must have one latitude
41. One PerimiterCoordinate (ZoneID) must have one longitude
42. Each Zone\_Contract is uniquely identified by ZoneID, ContractID
43. Each Contract is uniquely identified by one ID
44. One Contract (ID) may have one notes
45. Each Employee is uniquely identified by one ID
46. One Employee (ID) may have one firstName
47. One Employee (ID) must have one lastName
48. One Employee (ID) must have one phone
49. One Employee (ID) must have one email
50. Each DroneTechnician is uniquely identified by one EmployeeID
51. Each DronePilot is uniquely identified by one EmployeeID
52. Each AdministrativeExecutive is uniquely identified by one EmployeeID
53. Each Salesperson is uniquely identified by one EmployeeID
54. Each Customer is uniquely identified by one ID
55. One Customer (ID) may have one firstName
56. One Customer (ID) must have one lastName
57. One Customer (ID) may have one phone
58. One Customer (ID) may have one email
59. One Customer (ID) may have one company
60. Each Subscription is uniquely identified by one ID
61. One Subscription (ID) may have one discount
62. One Subscription (ID) must have one billingDate
63. One Subscription (ID) must have one recurring
64. Each SubscriptionType is uniquely identified by one Name
65. One SubscriptionType (Name) must have one price
66. One SubscriptionType (Name) must have one description
67. Each Gold is uniquely identified by one SubscriptionID
68. Each Platinum is uniquely identified by one SubscriptionID
69. Each SuperPlatinum is uniquely identified by one SubscriptionID

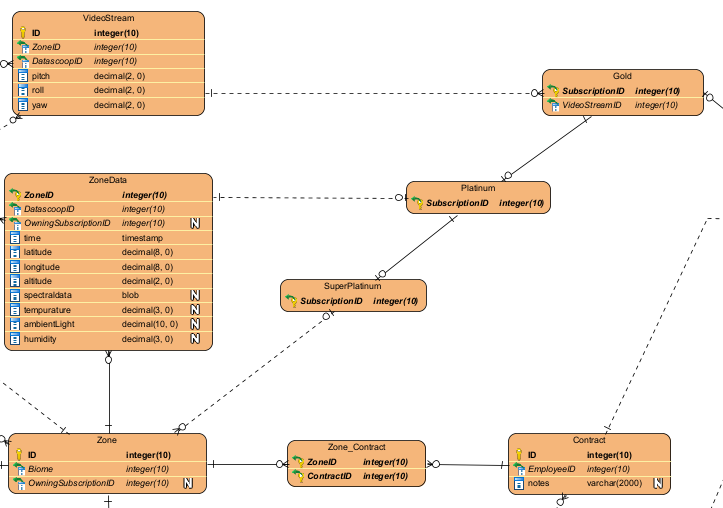
Analysis of attributes did not reveal any flaws in the model.

### Binary Relationships

1. One DataScoop (ID) may stream one or more VideoStream (ID)
2. One VideoStream (ID) must relate to one DataScoop (ID)
3. One DataScoop (ID) may provide one or more ZoneData (ID)
4. One ZoneData (ID) must relate to one DataScoop (ID)
5. One PartType (ID) may have one or more PartInstance (ID)
6. One PartInstance (ID) must reference one PartType (ID)
7. One Supplier (ID) may supply one or more PartType (ID)
8. One PartType may be supplied by one or more Supplier (ID)
9. One Biome (ID) may relate to one or more DataScoop (ID)
10. One DataScoop (ID) must be configured for one biome (ID)
11. One Biome (ID) may relate to one or more Zone (ID)
12. One Zone (ID) must reference one Biome (ID)
13. One Zone (ID) must have one or more PerimiterCoordinate (ZoneID)
14. One PerimiterCoordinate (ZoneID) must be for one Zone (ID)
15. One Zone (ID) may have one or more ZoneData (ZoneID)
16. One ZoneData (ZoneID) must be for one Zone (ID)
17. One Zone (ID) may have one or more VideoStream (ID)
18. One VideoStream (ID) must be for one Zone (VideoStream)
19. One Zone (ID) may have one or more Contract (ID)
20. One Contract may have one or more Zone (ID)
21. One Employee (ID) may manage one or more Employee (ID)
22. One Employee (ID) may be managed by one Employee (ID)
23. One Employee (ID) may be one DroneTechnician (EmployeeID)
24. One DroneTechnician (EmployeeID) must be one Employee (ID)
25. One Employee (ID) may be one DronePilot (EmployeeID)
26. One DronePilot (EmployeeID) must be one Employee (ID)
27. One Employee (ID) may be one AdministrativeExecutive (EmployeeID)
28. One AdministrativeExecutive (EmployeeID) must be one Employee (ID)
29. One Employee (ID) may be one Salesperson (EmployeeID)
30. One Salesperson (EmployeeID) must be one Employee (ID)
31. One AdministrativeExecutive (EmployeeID) may write one or more Contract (ID)
32. One Contract (ID) must be written by one AdministrativeExecutive (EmployeeID)
33. One Salesperson (EmployeeID) may sell one or more Subscription (ID)
34. One Subscription (ID) may be sold by one Salesperson (EmployeeID)
35. One Contract (ID) may have one or more Subscription (ID)
36. One Subscription (ID) must be for one Contract (ID)
37. One Customer (ID) may have one or more Subscription (ID)
38. One Subscription (ID) must relate to one customer (ID)
39. One SubscriptionType (name) may have one or more Subscription (ID)
40. One Subscription (ID) must reference one SubscriptionType (name)
41. One Subscription (ID) may be one Gold (SubscriptionID)
42. One Gold (SubscriptionID) must be one Subscription (ID)
43. One Gold (SubscriptionID) may be one Platinum (SubscriptionID)
44. One Platinum (SubscriptionID) must be one Gold (SubscriptionID)
45. One Platinum (SubscriptionID) may be one SuperPlatinum (SubscriptionID)
46. One SuperPlatinum (SubscriptionID) must be one Platium (SubscriptionID)
47. One Gold (SubscriptionID) may control one VideoStream (ID)
48. One VideoStream may have one Gold (SubscriptionID)
49. One Platinum (SubsriptionID) must have own one or more ZoneData (ZoneID)
50. One ZoneData (ZoneID) may have one Platinum (SubsciptionID)
51. One SuperPlatinum (SubscriptionID) must own one Zone (ID)
52. One Zone (ID) may have one SuperPlatinum (SubscriptionID)

Analysis of binary relationships revealed three flaws in the model. The cardinality of VideoStream to Gold should be zero or one, not Exactly one, and VideoStreamID on the Gold Entity should be nullable. The cardinality of ZoneData to Platinum should be zero or many, not exactly one. The cardinality of Zone to SuperPlatinum should be exactly one, not zero or more. These changes have been made to the logical diagram.

Flawed Diagram section:



## Examples with Sample Data

### Attributes

1. DataScoop 001 has isDeployed as True (1)
2. DataScoop 001 has latitude -74.44425
3. DataScoop 001 has latitude -153.233678
4. DataScoop 001 has altitude 134.21
5. PartInstance 245 has condition “Needs repair”
6. PartType 23 has name “Motor”
7. PartType 23 has description NULL
8. Supler 006 has name “Done Parts inc.”
9. Supplier 006 has email “droneparts@email.server”
10. Supplier 006 has phone “123 456 7890”
11. Supplier 006 has website NULL
12. Supplier 006 has notes “Provides high quality parts”
13. Biome 1 has name “Jungle”
14. Biome 1 has description “Needs water resistant components”
15. Inspection 23, 145, 11/09/24 has report “The motor had a broken gear replaced”
16. VideoStream 2143 has pitch -23.32
17. VideoStream 2143 has roll 0.00
18. VideoStream 2143 has yaw 45.00
19. ZoneData 86 has time 2024-05-30T17-48-26-211Z
20. ZoneData 86 has latitude -86.09598
21. ZoneData 86 has longitude -02.22679
22. ZoneData 86 has altitude 235.95
23. ZoneData 86 has spectralData NULL
24. ZoneData 86 has the temperature 15.41
25. ZoneData 86 has the ambientLight 235.21747632
26. ZoneData 86 has the humidity 77
27. PerimiterCoordinate 56 has latitude 37.342556
28. PerimiterCoordinate 56 has longitude 359.99999
29. Contract 146 has notes “Access to area by boat only”
30. Employee 2156 has firstName “Zvrambus”
31. Employee 2156 has lastName “Thurnalpth”
32. Employee 2156 has phone “011 011 01111”
33. Employee 2156 has email “ZThurnalpth@FlightStream.nz”
34. Customer 007 has firstName “Froomba”
35. Customer 007 has lastName “Ongong-gabolck”
36. Customer 007 has phone NULL
37. Customer 007 has email “dumname@gmail.com”
38. Customer 007 has company “Department of Conservation”
39. Subscription 426 has discount 0.02
40. Subscription 426 has billingDate 30/06/2024
41. Subscription 426 has recurring False (0)
42. SubscriptionType Gold has price 149.99
43. SubscriptionType Gold has description “Allows control of one video stream”

### Binary Relationships

1. DataScoop 4029 streams VideoStream 26
2. VideoStream 26 is streamed by DataScoop 4029
3. DataScoop 4029 provides ZoneData 23
4. ZoneData 1203434345 relates to DataScoop 4029
5. PartType 46 has PartInstance 405
6. PartInstance 405 references PartType 46
7. Supplier 400 supplies PartType 46
8. PartType 46 is supplied by Supplier 400
9. Biome 7 relates to DataScoop 005
10. DataScoop 005 is configured for Biome 7
11. Biome 7 relates no Zones
12. Zone 46 references Biome 4
13. Zone 46 has PerimiterCoordinate 46
14. PerimiterCoordinate 46 is for Zone 46
15. Zone 46 has ZoneData 46
16. ZoneData 46 is for Zone 46
17. Zone 46 has VideoStream 2442
18. VideoStream 2442 is for Zone 46
19. Zone 46 is in Contract 55
20. Contract 55 has Zone 46
21. Employee 38 manages Employee 29
22. Employee 29 is managed by Employee 38
23. Employee 8888 is DroneTechnician 8888
24. DroneTechnician 8888 is Employee 8888
25. Employee 8888 is DronePilot 8888
26. DronePilo 8888 is Employee 8888
27. Employee 3453 is AdministrativeExecutive 3453
28. AdministrativeExecutive 3453 is Employee 3453
29. Employee 002 is Salesperson 002
30. Salesperson 002 is Employee 002
31. AdministrativeExecutive 123 writes Contract 54
32. Contract 54 is written by one AdministrativeExecutive 123
33. One Salesperson 456 sells Subscription 999
34. Subscription 999 is sold by one Salesperson 456
35. Contract 8976 has Subscription 1234
36. Subscription 1234 is for Contract 8976
37. Customer 007 buys Subscription 1234
38. Subscription 1234 relates to Customer 007
39. SubscriptionType Gold has Subscription 453
40. Subscription 453 references SubscriptionType Gold
41. Subscription 453 is Gold 453
42. Gold 453 is Subscription 453
43. Gold 453 is Platinum 453
44. Platinum 453 is Gold 453
45. Platinum 453 is SuperPlatinum 453
46. SuperPlatinum 453 is Platinum 453
47. Gold 005 controls VideoStream 1098
48. VideoStream 1098 has Gold 005
49. Platinum 011 owns ZoneData 46
50. ZoneData 46 has Platinum 011
51. SuperPlatinum 011 owns Zone 46
52. Zone 46 has SuperPlatinum 011

## Ternary or Higher Relationships

1. One DronePilot (EmployeeID) may pilot one or more DataScoop (ID) in one or more Zone (ID)
2. One DataScoop (ID) may be piloted by one or more DronePilot (EmployeeID) in one or more Zone (ID)
3. One Zone (ID) may have one or more DronePilot (EmployeeID) piloting one or more DataScoop (ID)
4. One DataScoop (ID) must comprise one or more PartInstance (ID) on one or more dateInstalled (composite key attribute)
5. One PartInstance (ID) may be installed in one DataScoop (ID) on one or more dateInstalled (composite key attribute)
6. One PartInstance (ID) may be inspected by one DroneTechnician (EmployeeID) on one or more date (composite key attribute)
7. One DroneTechnician (EmployeeID) may inspect one PartInstance (ID) in one or more date (composite key attribute)

Analysis of ternary relationships did not reveal any flaws in the model.

## Ternary Relationship Examples

1. DronePilot 42 pilots DataScoop 4002 in Zone 24
2. DataScoop 4002 is piloted DronePilot 42 in Zone 24
3. Zone 24 has DronePilot 42 piloting DataScoop 4002
4. DataScoop 4002 comprises PartInstance 86 on 24/05/2024
5. PartInstance 86 is installed in DataScoop 4002 on 24/05/2024
6. PartInstance 86 is inspected by one DroneTechnician 24 on 13/01/2025
7. DroneTechnician 24 inspects PartInstance 86 on 13/01/2025

Milestone 3: Physical Implementation

# References

Craig Stedman. (2022, May 31). What is data governance and why does it matter? Data Management. <https://www.techtarget.com/searchdatamanagement/definition/data-governance>

Fortinet. (2024). Data security: Definition, importance, and types. <https://www.fortinet.com/resources/cyberglossary/data-security>

Gattis, T. (2024, February 14). Security, data privacy & data management | Best practices guide. Annex Cloud. <https://www.annexcloud.com/resources/guides/data-security-and-privacy-best-practices-guide/>

GeeksforGeeks. (2023, September 19). Normal forms in DBMS. <https://www.geeksforgeeks.org/normal-forms-in-dbms/>

IBM. (n.d.). What is data modeling? Retrieved February 25, 2024, from <https://www.ibm.com/topics/data-modeling>

Jain, M. (2022, July 22). BCNF in DBMS. Scaler Topics. <https://www.scaler.com/topics/bcnf-in-dbms/>

JavaTPoint. (n.d.). DBMS 4NF. Retrieved May 26, 2024, from <https://www.javatpoint.com/dbms-forth-normal-form>

Jotform. (2020, August 4). 6 key data management principles. <https://www.jotform.com/blog/data-management-principles/>

Ranjan, S. (2024, January 1). Metadata management in data-driven organisations. Meer. <https://www.meer.com/en/75824-metadata-management-in-data-driven-organisations>

Sagacity solutions. (2024). What is data quality? Why it's important. <https://www.sagacitysolutions.co.uk/about/news-and-blog/what-is-data-quality/>

Sheldon, R. (2022, June 28). What is data lifecycle management (DLM)? TechTarget. <https://www.techtarget.com/searchstorage/definition/data-life-cycle-management>

Sherman, R. (2015). Foundational Data Modelling. ScienceDirect. <https://www.sciencedirect.com/topics/computer-science/conceptual-data-model>