

DAT601 assignments

Lee Vartha



-/-/-

Contents

[Assignment 1 Part 1: 2](#_Toc198811122)

[Assignment 1 Part 2 4](#_Toc198811123)

[1. Description of Conceptual Modelling: 4](#_Toc198811124)

[2. Components of an ERD: 5](#_Toc198811125)

[Reasoning and Purpose of all Parts of the Model: 7](#_Toc198811126)

[Data Dictionary: 12](#_Toc198811127)

[Rationale for Data Dictionary: 5](#_Toc198811128)

[Assumed Business Rules (with reasoning): 6](#_Toc198811129)

[Assignment 2 Part 1 – Logical Design: 8](#_Toc198811130)

[1. Conceptual to Relational Logical Model: 8](#_Toc198811131)

[2. NaLER Analysis: 27](#_Toc198811132)

[3. Report Details 41](#_Toc198811133)

[Assignment 2 Part 2 – Physical Design: 41](#_Toc198811134)

# Assignment 1 Part 1:

|  |  |
| --- | --- |
| QUERIES | |
| **Query 1:** | **Query 2:** |
|  |  |
| **Query 3:** | **Query 4:** |
|  | A screenshot of a computer  AI-generated content may be incorrect. |
| **Query 5:** | **Query 6:** |
| A screenshot of a computer  AI-generated content may be incorrect. | A screenshot of a computer  AI-generated content may be incorrect. |
| **Query 7:** | |
| A screenshot of a computer  AI-generated content may be incorrect. | |
| JOINS | |
| **Join 1:** | **Join 2:** |
| A screenshot of a computer  AI-generated content may be incorrect. | A screenshot of a computer  AI-generated content may be incorrect. |
| SUBQUERIES: | |
| **Subquery 1:** | **Subquery 2:** |
|  |  |
| Combined Query: | |
|  | |
| VIEWS: | |
| **View 1:** | **View 2:** |
|  |  |

# Assignment 1 Part 2

# Description of Conceptual Modelling:

Conceptual Modelling is the implementation of creating a simplified representation of a complicated system, which allows it to be more understandable and manageable. This represents the first layer of data modelling and ensures that the system reflects the requirements of the business accurately.

The Extended Chen Entity-Relationship Diagram (ERD) is a technique that goes beyond the original Chen ER model. The extended version utilizes more features include subtypes (ISA relationships), weak entities, multi-valued attributes, associative entities. Etc.

# Components of an ERD:

1. Entities

Entities are definable things that can have data stored within it. Examples could be a customer, product etc. Entities are shown as a rectangle and may have different views depending on the type:

* **Strong entity**: the standard entity type that can act as its own independent state
* **Weak entity**: this is an entity that cannot independently exist and relies on the owner entity.
* **Associative entity:** this is an entity used for many-to-many relationships

1. Attributes

These define the traits of an entity. The attributes are shown as ovals in diagrams and are connected to their parented entities using a connected line.

Some attributes include:

* **Simple** **attributes**: these are very concise traits (e.g., Name)
* **Key Attributes**: these represent an attribute that uniquely identifies an entity (e.g., ‘CustomerID’
* **Composite** **attributes**: these are traits that can be divided into multiple simpler attributes (e.g., “Address” can be broken down into ‘street’, ‘city’, ‘zip code’ etc.)
* **Derived Attributes**: These are attributes which values need to be calculated from other attributes to get a result (e.g., Age is derived from dateOfBirth)
* **Multi**-**valued attributes**: these can have more than one value/results for an entity (e.g., Hobby (people usually have more than one)

1. Keys

Keys are a way to uniquely identify an entity from a set. Key attributes are represented by an oval (same as attributes), but the text of the attribute is underlined to differentiate it.

Some keys include:

* **Candidate key**: This is a simple/composite key that is unique as no two rows in the table would have the same value
* **Composite key**: this composes two or more attributes
* **Primary key**: This is a key that is a non-null value to represent identification for a whole entity
* **Foreign key**: These are keys that an entity can reference a primary key from another entity

1. Relationships

Relationships show the connections between entities – these are represented by diamonds (between the lines connected for two entities) and can have attributes of their own.  
Relationships include cardinality and participation, which defines how the entities are linked and whether participation is necessary.

* **Cardinality**: Regards the number of times an instance in one entity can relate to instances of another entity (One customer can purchase many products – many products can be purchased by one customer)
* **Participation**: Explains the level of dependency another entity has with a relation with another entity (A bill cannot be created without a purchase being made)

1. Relationship Degrees

Relationship degrees mean the number of entities participated in the relationship set

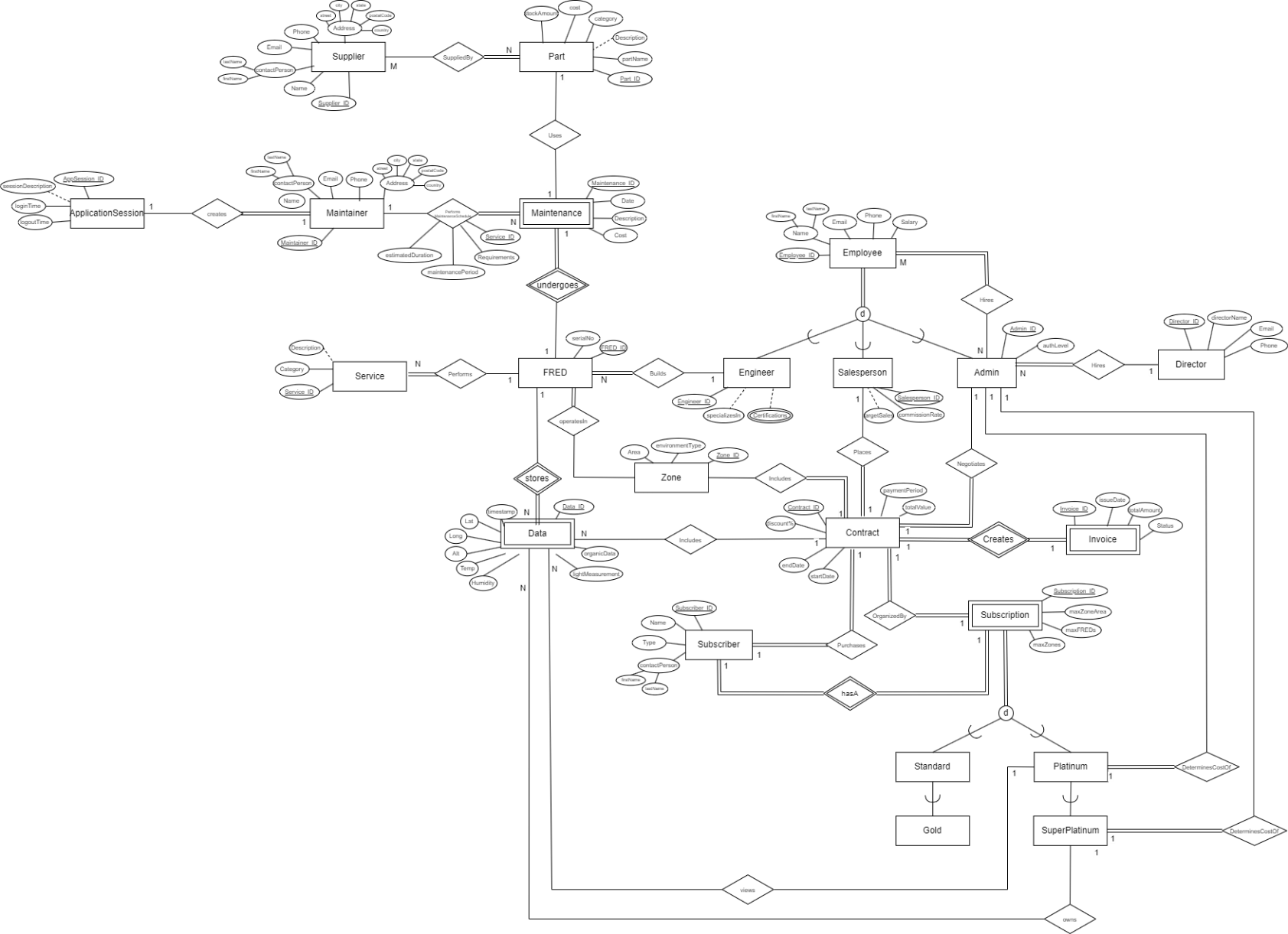
* **Unary Relationship**: Only one entity is participating as a relationship (also known as a recursive relationship
* **Binary Relationship**: Two entities are involved in the set of relationships
* **Ternary Relationship**: Three entities are participating in the relationship created
* **N-ary Relationship**: This is a form of relationship when the **n** number of total entities are participating – this could have up to 5+ entities relating.

Bhagwhat, S. (2022, June 23). *Degree of relationship in DBMS.* Scaler Topics.  
<https://www.scaler.com/topics/degree-of-relationship-in-dbms/>

Rodina, D. (n.d.). *Composite key (entity-relationship diagram).* Software Ideas Modeler – diagramming case tool.  
<https://www.softwareideas.net/composite-key>

Lucidchart (n.d). *What is an entity relationship diagram (ERD)?.*  
<https://www.lucidchart.com/pages/er-diagrams>

**Chen Conceptual Entity Relationship Model**



**[file for this diagram is in GITHUB]**

# Reasoning and Purpose of all Parts of the Model:

This ERD model was created using the Chen notation – utilizing this style made it easier to visualize the system involving drone services, maintenance, staff roles etc.  
Each component of the model – entities, attributes, relationships, keys, ISA hierarchies etc. – have been designed to reflect the required business logic.

Entities:

1. **FRED**

This represents the drone devices that are used to carry out services and data collection. The purpose of this is being the operational unit of the business.

1. **Data (Weak Entity)**

This is the results that have been collected from the FREDs – this includes (but not limited to) images, analytics etc. The purpose of this is used for zone records, reporting and analysis.

1. **Contract**

This is the mutual agreement between the ParkWorks company and the subscriber whose purchasing services. The purpose of the contract is to solidify the use of a FRED for a subscribing company.

1. **Subscription**

Service tier classifications that are implemented using ISA (e.g., Standard ISA SubscriptionType). The purpose of this is to define the levels of access that the contract withholds (in terms of service)

1. **Zone**

These are areas used to categorize specific locations that FREDs can perform within to do particular services. The purpose of this is to optimize service routing.

1. **Service**

This defines the activities that are implemented by the FRED drones, such as tree trimming, track clearing, grass mowing, planting etc. The purpose of this is to show the link between what was agreed in the contracts and what the required operations are.

1. **Maintenance (Weak Entity)**

This is used to log specific maintenance actions that is done for a FRED, whether its cleaning, software updates or fixing/replacing a part. The purpose of this is to service the FREDs to maintain its longevity to perform tasks – through this, the maintainer would log a record to track drone upkeep history.

1. **Maintainer/Engineer/Admin/Director/Salesperson**

These each represent staff/third party roles, each equipped with different responsibilities. The purpose of this is to show employee organization and task delegation. For example, engineer and salesperson are visualized in the diagram using ISA hierarchy (e.g., Engineer ISA Employee)   
Directors are used to hire admin, and admin hire employees. Maintainers are used to maintain/service the FREDs.  
Engineers design and build the FREDs and salespersons use marketing specialties to advertise the FREDs and initialize contracts, where admins finalize them.

1. **MaintenanceSchedule (PerformsMaintenanceSchedule) – Relationship with Entities**

This defines the maintenance operations and what need to be done. The purpose of this is to track future tasks and enable the organization of services that need to be carried out.

1. **ApplicationSession**

This represents the active use of the maintenance system and shows how it was used by maintainers. The purpose of this is to monitor system access and usage.

1. **Subscriber**

This represents the organization that signs up for a service and agrees to a contract. The purpose of this is to identify the individuals who are entitled to levels of service.

1. **Part**

This defines the components used in the repair and maintenance of a drone. The purpose of this to track items for a FRED.

1. **Supplier**

This is a third party organization that provides the parts used in the drone maintenance. The purpose of this is to represent the companies that can restock the parts used.

1. **Invoice**

This represents the bill document given to a subscriber, requesting payment for the contracted services. The purpose of this is to provide a financial audit for the services done by a FRED.

Relationships and Design:

1. **FRED performs Service**

This shows the drones are responsible for doing the services – a service MUST be performed by a drone. (One FRED can perform zero to many services)

1. **FRED OperatesIn Zone**

A FRED operates in particular zones that they have been delivered to based on contracts that have been finalized from subscribers. (One or more FREDS could operate in 1 or more zones)

1. **Zone LocatedIn Region**

This shows the hierarchical relationship between the two entities – regions can have one or more zones within them. Every zone MUST belong to a region (Zero to many zones may be located in a region)

1. **Director hires Admin / Admin Hires Employee**

This represents authority hierarchy. Admins can’t exist without directors hiring and employees won’t be there without an admin hiring them. (Many directors can hire many administrators – One or more administrators can hire one or more employees)

1. **Contract Includes Zone**

This indicates where the contracted service will be delivered to (One contract includes one or more zones)

1. **Contract overlooks Data**

A contract tells the types of data that needs to be collected or managed depending on what the agreement is. (One contract overlooks one or more sets of data from a FRED)

1. **FRED undergoes Maintenance**

Each FRED is subject to scheduled maintenance to checkup on its performance and whether it needs to be cleaned, fixed etc. (One or more FREDs undergo one or more maintenance sessions in their lifetime)

1. **Maintainer PerformsMaintenanceSchedule (for) Maintenance**

A maintainer carries out the maintenance tasks that need to be done to FRED units. (Many maintenance sessions can be carried out by a maintainer). Each maintainer needs to follow the schedule which outlines when services need to be done to FREDs. (a maintainer follows a maintenance schedule)

1. **Maintainer creates ApplicationSession**

The maintainer logs application sessions and gets monitored when performing entries on the application. (A maintainer creates an application session)

1. **Maintenance uses Part**

Parts may be used during the maintenance sessions for FREDs (One maintenance session may use zero to many part components for FRED maintenance services)

1. **Part SuppliedBy Supplier**

Each part used in maintenance needs to be supplied by third-party suppliers. (One or more parts have been supplied by one or more suppliers)

1. **Engineer builds FRED**

An engineer designs and assembled FRED drones. (One engineer can build one or many FREDs)

1. **Engineer ISA Employee**

An engineer is a subtype of an employee within ParkWorks.

1. **Salesperson ISA Employee**

A salesperson is a subtype of an employee within ParkWorks.

1. **Admin negotiates Contract**

The administrators handle the negotiation of the service contracts with subscribers, finalizing the details that have been initialized by the salesperson. (One admin negotiates many contracts – one contract is negotiated by one admin)

1. **Admin determinesCost of Platinum/SuperPlatinum**

Admins have the ability to determine details and cost of the platinum and super platinum subscriptions, given a subscriber has chosen one of those types. (An admin determinesCost of a platinum/superplatinum subscription)

1. **Platinum Views Data**

Subscribers who purchased a contract with a platinum subscription can view data that’s been made by FRED drones. (A platinum subscription views many sets of data)

1. **SuperPlatinum Owns Data**

Subscribers who have purchased a contract with a super platinum subscription has the ability and right to own the data that’s been created by the FRED drones (A super platinum subscription owns many sets of data)

1. **Salesperson places Contract**

The salesperson processes the placement of contracts between the company and the subscriber/s involved – the starting point before its passed onto the admin to finalize. (One salesperson can place multiple contracts – but one contract is placed by one salesperson)

1. **Contract OrganizedBy SubscriptionType**

Each contract is under a certain subscription type – this determines the pricing and feature capabilities of the contract requirements (One contract is organized by one subscription type)

1. **Subscriber purchase Contract**

Subscribers may agree to purchase a contract to be able to utilize drone services. (A subscriber purchases one contract)

1. **Subscriber hasA Subscription**

This shows that the subscriber would also be linked to a specific subscription type, in relation to the contract – including this reduces ambiguity and confirms they would be associated with a subscription. (A subscriber has a (singular) subscription)

1. **Contract creates Invoice**

When a contract is initialized, it creates an invoice that bills the subscriber for the services performed. (One contract creates one invoice)

How I used ‘EXTENDED’ notation for the ERD:

1. **ISA**

I used ISA for the salesperson and engineer entities as they are a subtype of ‘employee’. This was used as it supports inheritance.

1. **Weak entities**

I used these as it aligns with data normalization – these entities cannot exist independently and relies on the parent entities (e.g., Data cannot exist without FREDs)

1. **Associative entities**

The contract was modelled as associative as I had linked subscriber, SubscriptionType, zone, salesperson and admin.

Principles of Data Management:

1. **Entity Integrity:**

Each entity has a defined primary key, which encourages the use of unique identification.

1. **Data accuracy**:

The attributes used for each entity showcases completeness.

1. **Normalization**:

This model attempts to avoid duplication through entities and weak entities.

1. **Scalability**:

The use of ISA hierarchies supports system scalability without having to do major changes in the display.

# Data Dictionary:

Table 1: Document Entities

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity Name** | **Description** | **Aliases** | **Occurrence** |
| FRED | A ‘Free Roaming Environmental Drone’ that is used for maintenance and collects sensor data | Drone, Robot | One record for each FRED drone that are deployed by ParkWorks |
| Contract | An arrangement between ParkWorks and a client for the FRED services | Agreement, Document, Subscription | One record for every subscription service account with a client |
| Employee | A person that has been employed by ParkWorks | Staff, Worker | One record for every ParkWorks worker |
| Salesperson | An employee that sells subscriptions | Sale representative | One record for each employee who sells subscription |
| Engineer | An employee that designs and builds FREDs | Designer, Builder | One record for each employee that are capable of technical engineering |
| Executive Admin | An employee that has administrative capabilities | Administrator | One record for each employee that can manage contracts and change prices |
| Director | A high level member that is responsible for hiring admins | Executive | One record for each director that hires admins A system may have one or more directors |
| ApplicationSession | This represent the usage instance of the application that is designed for the maintainer. | App session, app usage | One record for each session that is entered by a maintainer |
| Region | Represents a geological area that contains zones for FREDs to roam in. | Area | One record for each region that has zones and has FREDs occupying it. A region may exist with or without any assigned zones. |
| Subscriber | Someone who contracts ParkWorks for FRED services | Customer, Client, End User | One record for every client that has subscribed to ParkWorks services |
| Subscription | A category of subscription with different tiers (standard, gold, platinum, super platinum) | Plan | One record for each subscription plan (standard, gold, platinum, super platinum) |
| Standard | The first subscription type with the lowest features, including a maximum of ONE FRED, ONE Zone with maximum area being 10 hectares | Tier 1/Subscription Plan 1 | One record for the subscriber who chose the standard plan for the contract |
| Gold | The second subscription type with the generic features, including a maximum of THREE FREDS that can do THREE services in THREE zones. The maximum area being no more than 100 hectares | Tier 2/Subscription Plan 2 | One record for the subscriber who chose the gold plan for the contract |
| Platinum | The third subscription type with higher features, including being able to use as many FREDs with as many zones and services. They are able to receive data for their zones. | Tier 3/Subscription Plan 3 | One record for the subscriber who chose the platinum plan for the contract |
| Super Platinum | The forth subscription type with the highest features, including exclusive control of any FREDs, having as many zones and services desirable, alongside having the ability to view the location of FREDs using an application | Tier 4/Subscription Plan 4 | One record for the subscriber who chose the super platinum plan for the contract |
| Zone | An area where FREDs operate and gather data | Area, Section | One record for each of the boundaries where FREDs are released |
| Part | Component used to design and put together a FRED | Element, segment | One record for each part type used in a FRED |
| Supplier | Organization that provides the parts necessary for FREDs | Distributor, Seller, Provider | One record for each organization that supplies parts for a FRED |
| Service | A task that FREDs can do | Resource, Solution, Utility | One record for each type of service FREDs can do |
| Maintenance | A record of service that has been done for a FRED | Conservation, Care, Service | One record for each maintenance activity that a FRED has performed |
| Maintainer | A third-party company that performs maintenance for FREDs | Service Provider | One record for each company that maintains FREDs |
| Data | Environment data gathered by FREDs | Input, Details, Statistics | One record for a set of environmental data collected by a FRED |
| Invoice | A bill for subscription services | Bill, Account, Charge | One record for each bill period for a contract |

**Table 2: Document Relationships/Specializations- Generalizations**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Entity Name** | **Cardinality** | **Participation** | **Relationship** | **Participation** | **Cardinality** | **Entity Name** |
| **FRED** | N | Total | OperatesIn | Partial | M | **Zone** |
| **FRED** | 1 | Total | Stores | Total | N | **Data** |
| **FRED** | 1 | Total | Performs | Total | N | **Service** |
| **FRED** | 1 | Total | Undergoes | Total | 1 | **Maintenance** |
| **Part** | N | Total | SuppliedBy | Partial | M | **Supplier** |
| **Maintenance** | 1 | Partial | Uses | Partial | 1 | **Part** |
| **Contract** | 1 | Total | OrganizedBy | Partial | 1 | **Subscription** |
| **Contract** | 1 | Partial | Overlooks | Partial | N | **Data** |
| **Contract** | 1 | Total | Creates | Total | 1 | **Invoice** |
| **Contract** | 1 | Total | Includes | Partial | N | **Zone** |
| **Subscriber** | 1 | Partial | Purchases | Total | 1 | **Contract** |
| **Subscriber** | 1 | Total | HasA | Total | 1 | **Subscription** |
| **Engineer** | 1 | Partial | ISA (Specialization) | Total | 1 | **Employee** |
| **Salesperson** | 1 | Partial | ISA (Specialization) | Total | 1 | **Employee** |
| **Admin** | 1 | Partial | ISA (Specialization) | Total | 1 | **Employee** |
| **Admin** | N | Partial | Hires | Total | M | **Employee** |
| **Director** | 1 | Partial | Hires | Total | N | **Admin** |
| **Engineer** | 1 | Partial | Builds | Total | N | **FRED** |
| **Admin** | 1 | Partial | Negotiates (for platinum and super platinum) | Total | 1 | **Contract** |
| **Maintenance** | M | Total | PerformsMaintenanceSchedule | Partial | 1 | **Maintainer** |
| **Zone** | N | Total | LocatedIn | Partial | 1 | **Region** |
| **Maintainer** | 1 | Partial | Creates | Total | 1 | **ApplicationSession** |
| **Standard** | 1 | Partial | ISA | Total | 1 | **Subscription** |
| **Gold** | 1 | Partial | ISA | Total | 1 | **Subscription** |
| **Platinum** | 1 | Partial | ISA | Total | 1 | **Subscription** |
| **SuperPlatinum** | 1 | Partial | ISA | Total | 1 | **Subscription** |
| **Platinum** | 1 | Partial | Views | Partial | N | **Data** |
| **Super Platinum** | 1 | Partial | Owns | Partial | N | **Data** |

**Table 3: Document Attributes**

| **Entity Name** | **Attributes** | **Description** | **Domain** | **Aliases** | **Composite** | **Derived** | **Nulls** | **Key?** | **Default Value** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **FRED** | FredID | The unique identifier for every FRED | Integer | RobotID, DroneID | No | No | No | **Primary** | Auto-increment |
|  | serialNumber | The manufacturers number | Char(20) | serialNo | No | No | No | No | None |
|  | dateCommissioned | The date the FRED started to be of service | DateTime | startDate | No | No | Yes | No | Current date |
|  | Status | The FREDs current status (active, inactive) | Char(20) | Condition | No | No | No | No | ‘Active’ |
|  | lastCommunication | The timestamp of when it last transmitted data | Integer | lastContact lastDataTime | No | No | Yes | No | Current date/time |
|  | batteryLevel | The battery percentage | Decimal(5,2) | currentBattery powerLevel | No | No | No | No | 100 |
| **Contract** | contractID | The unique identifier for every contract | Integer | subscriptionID | No | No | No | **Primary** | Auto-increment |
|  | startDate | The timestamp of the date the contract started | Date | BeginDate | No | No | No | No | Current date/time |
|  | endDate | The timestamp of the date the contract will/has ended | date | TerminatedDate | No | No | **Yes** | No | None |
|  | paymentPeriod | The period schedule of the payments | Char(10) | paymentCycle | No | No | No | No | ‘Monthly’ |
|  | discountPercent | The discount that’s used for the subscription price | Decimal(5,2) | Discount | No | No | No | No | 0.00 |
|  | totalValue | The total money value of the contract | Currency | billValue | No | No | No | No | 0.00 |
|  | isExclusive | Boolean that indicates if any FREDs are exclusive to the contract | Boolean | Exclusive | No | No | No | No | False |
| **Employee** | employeeID | The unique identifier of every employee | Integer | workerID | No | No | No | **Primary** | Auto-increment |
|  | Name | The name of the employee | Char(100) | OrganizationName, CompanyName | **Yes (firstName, lastName)** | No | No | No | None |
|  | Email | The email address of the employee | Char(255) | WorkerEmail, EmployeeEmail | No | No | No | No | None |
|  | Phone | The phone number of the employee | Integer | WorkerPhone, WorkerPhone | No | No | **Yes** | No | None |
|  | Position | The employees position | Char(100) | Title, Job, Role | No | No | No | No | None |
|  | Salary | The employees annual salary | Currency | Pay | No | No | **Yes** | No | None |
| **Salesperson** | salesPersonID | The unique identifier of every salesperson | Integer | SalesID, SalesRepID | No | No | No | **Primary** | Auto-increment |
|  | commissionRate | The percentage of sales that have been earned | Decimal(5,2) | Commission | No | No | No | No | 0.00 |
|  | maxDiscountPercent | The discount percent that the salesperson can provide for clients | Decimal(5,2) | MaxDiscount | No | No | No | No | 0.00 |
|  | targetSales | The target number amount of sales they aim for monthly | Currency | Target, SalesTarget | No | No | **Yes** | No | None |
| **Engineer** | engineerID | The unique identifier for each engineer | Integer | TechID | No | No | No | **Primary** | Auto-increment |
|  | specializesIn | The area that the engineer is an expert in | Char(100) | Expertize, Specialization, Specialty | No | No | **Yes** | No | None |
|  | Certification | The list of technical certifications | Char(255) | Qualification/s | No | No | **Yes** | No | None |
| **Executive Admin** | adminID | The unique identifier of every executive administrator | Integer | ExecutiveAdminID, AdministratorID | No | No | No | Primary | Auto-increment |
|  | authorizationLevel | Sees the level of authorization | Char(50) | AuthLevel | No | No | No | No | ‘Standard’ |
|  | canChangePrices | The Boolean to see if they have the ability to change prices (for subscriptions) | Boolean | PriceAuthority | No | No | No | No | False |
| **Director** | directorID | The unique identifier for each director | Integer | supervisorID | No | No | No | Primary | Auto-increment |
|  | directorName | The name of the director | Char(255) | SupervisorName | Yes (firstName, lastName) | No | No | No | None |
|  | Email | The email of the director | Char(255) | SupervisorEmail | No | No | No | No | None |
|  | Phone | The phone number for the director | Integer | SupervisorPhone | No | No | No | No | None |
|  | position | The position of the director | Char(255) | SupervisorPosition | No | No | No | No | None |
| **Subscriber** | subscriberID | The unique identifier for every subscriber | Integer | CustomerID, ClientID | No | No | No | **Primary** | Auto-increment |
|  | Name | The name of the subscriber (organization) | Char(100) | ClientName, CustomerName, OrganizationName | No | No | No | No | None |
|  | Type | The type of organization subscriber (govt. or private) | Char(50) | OrganizationType, ClientType | No | No | No | No | None |
|  | contactPerson | The name of the contact person for the subscriber | Char(100) | PrimaryContact | **Yes (firstName, lastName)** | No | No | No | None |
|  | Email | The email address of the subscriber | Char(255) | ContactEmail, ClientEmail | No | No | No | No | None |
|  | Phone | The phone number of the subscriber | Integer | ContactPhone, ClientPhone | No | No | No | No | None |
|  | Address | The physical address of the subscriber | Char(255) | ClientAddress, CustomerAddress, OrganizationAddress, Location | **Yes (street, city, state, postalCode, country)** | No | No | No | None |
|  | SubscriptionType | Defines what tier they have associated with the contract | Char(10) | TierType, TierPlan, PlanType | **No** | No | No | No | ‘Standard’ |
|  | accountMade | The timestamp date of when the subscriber registered into the system | Datetime | registerDate | No | No | No | No | Current date |
| **Subscription** | subscriptionID | The unique identifier for each subscription type | Integer | PlanID, TierID | No | No | No | **Primary** | Auto-increment |
|  | Name | The name of the subscription type | Char(50) | PlanName, TierName, SubscriptionName | No | No | No | No | None |
|  | Description | The description of the subscription type | Char(255) | PlanDescription, TierDescription, SubscriptionDescription, SubscriptionTypeDescription | No | No | **Yes** | No | None |
|  | maxZoneArea | Describes the max zone area in hectares for that type | Decimal(10,2) | MaxArea, ZoneArea | No | No | **Yes** | No | None |
|  | maxFREDs | Maximum amount of FREDs allowed for the type | Integer | MaxRobots, MaxDrones, MaxRobotAmount, MaxDroneAmount | No | No | **Yes** | No | None |
|  | maxZone | Describes the max amount of zones for that type | Integer | MaxRegion | No | No | **Yes** | No | None |
|  | basePrice | The price before discounts | Currency | UsualPrice, StandardPrice, StandardAmount, UsualAmount | No | No | No | No | 0.00 |
|  | discountPercent | The discount percentage for payments | Numeric(5,2) | DiscountAmount | No | No | No | No | 0.00 |
| **Standard** | StandardID | The unique identifier for every standard subscription | Integer | Tier1ID | No | No | No | **Primary** | Auto-increment |
|  | AmountOfFRED | The number of FREDs applied (LIMIT 1) | Integer | FREDNumber | No | No | No | **No** | 1 (LIMIT 1) |
|  | AmountOfZones | The number of zones applied in the contract (LIMIT 1) | Integer | ZoneNumber | No | No | No | **No** | 1 (LIMIT 1) |
| **Gold** | GoldID | The unique identifier for every gold subscription | Integer | Tier2ID | No | No | No | **Primary** | Auto-increment |
|  | AmountOfFRED | The number of FREDs applied (LIMIT 3) | Integer | FREDNumber | No | No | No | **No** | 1 (LIMIT 3) |
|  | AmountOfZones | The number of zones applied in the contract (LIMIT 3) | Integer | ZoneNumber | No | No | No | **No** | 1 (LIMIT 3) |
| **Platinum** | PlatinumID | The unique identifier for every platinum subscription | Integer | Tier3ID | No | No | No | **Primary** | Auto-increment |
|  | AmountOfFRED | The number of FREDs applied (UNLIMITED) | Integer | FREDNumber | No | No | No | **No** | 1 (UNLIMITED) |
|  | AmountOfZones | The number of zones applied in the contract (UNLIMITED) | Integer | ZoneNumber | No | No | No | **No** | 1 (UNLIMITED) |
| **Super Platinum** | SuperPlatinumID | The unique identifier for every region | Integer | Tier4ID | No | No | No | **Primary** | Auto-increment |
|  | AmountOfFRED | The number of FREDs applied (UNLIMITED) | Integer | FREDNumber | No | No | No | **No** | 1 (UNLIMITED) |
|  | AmountOfZones | The number of zones applied in the contract (UNLIMITED) | Integer | ZoneNumber | No | No | No | **No** | 1 (UNLIMITED) |
| **Zone** | zoneID | The unique identifier for every region | Integer | AreaID | No | No | No | Primary | Auto-increment |
|  | zoneName | The name of the zone | Char(20) | areaName | No | No | No | No | None |
|  | environmentType | The type of environment in the region | Char(50) | SceneryType | No | No | No | No | None |
|  | area | The size of the region in hectares | Decimal(10,2) | ZoneArea, SizeInHectares | No | No | No | No | 0.00 |
|  | boundary | The boundary coordinates | Char(100) | AreaBoundary, ZoneBoundary | No | No | No | No | None |
|  | country | The country where the zone is | Char(100) | Nation | No | No | No | No | None |
| **Part** | partID | The unique identifier for every part for FREDs | Integer | ComponentID | No | No | No | **Primary** | Auto-increment |
|  | partName | The name of the part | Char(100) | Name, ComponentName | No | No | No | No | None |
|  | Description | The description of the part | Char(255) | PartDescription, ComponentDescription, PartDetails, Component Details | No | No | **Yes** | No | None | |  |
|  | Category | What type of part it is | Char(50) | PartType, PartCategory | No | No | No | No | None |
|  | Cost | The amount it costs to purchase a part | Currency | PartCost, PartAmount, PartPrice | No | No | No | No | 0.00 |
|  | stockAmount | How many pieces of stock is available for a part | Integer | StockQuantity, stockCount, InventoryQuantity, InventoryCount | No | No | No | No | 0 |
| **Supplier** | supplierID | The unique identifier for every supplier associated | Integer | VendorID | No | No | No | **Primary** | Auto-increment |
|  | Name | The name of the supplier company | Char(100) | VendorName, SupplierName | No | No | No | No | None |
|  | contactPerson | The name of the primary contact for the company | Char(100) | VendorContact, SupplierContact | **Yes (firstName, lastName)** | No | No | No | None |
|  | email | The suppliers email address | Char(255) | VendorEmail, SupplierEmail | No | No | No | No | None |
|  | Phone | The suppliers phone number | Integer | VendorPhone, SupplierPhone | No | No | No | No | None |
|  | Address | The supplier company’s physical address | Char(255) | VendorAddress, VendorLocation, SupplierAddress, SupplierLocation | **Yes (street, city, state, postalCode, country)** | No | No | No | None |
| **Service** | serviceID | The unique identifier for every service/task done | Integer | JobID, TaskID | No | No | No | **Primary** | Auto-increment |
|  | Description | The description of the service (what task is done) | Char(255) | ServiceDescription, TaskDescription, ServiceDetails, TaskDetails | No | No | **Yes** | No | None |
|  | category | What type of service it is | Char(50) | ServiceType, ServiceCategory, TaskType, TaskCategory | No | No | No | No | None |
| **ApplicationEntry** | applicationEntryID | The unique identifier of every application entry session | Integer | programEntryID, applicationUsageID, applicationSessionID | no | No | No | Primary key | Autoincrement |
|  | usageDescription | The description of what was done during the session | Char(255) | sessionDescription, entryDetails, usageDetails | No | No | Yes | No | None |
|  | LoginTime | The time of login | Datetime | logTime | No | No | Yes | No | CurrentDate |
|  | logoutTime | The time of logout | datetime | exitTime | No | no | Yes | No | None |
| **MaintenanceSchedule RELATINOSHIP** | ScheduleID | The unique identifier of every maintenance schedule | Integer | timetableID | No | No | No | Primary Key | Auto-increment |
|  | Requirements | The description of the maintenance required | Char(255) | serviceDescription, ServiceDetails, maintenanceDetails | No | No | No | No | None |
|  | maintenancePeriod | The maintenance interval in days | Integer | ServicePeriod, ServiceInterval, MaintenanceInterval | No | No | No | No | 182.5 (6 months) |
|  | extimatedDuration | The estimated time to complete the maintenance in hours | Integer | maintenanceEstimation | No | No | No | No | 0.00 |
| **Maintenance (Weak)** | maintenanceID | The unique identifier for every maintenance service | Integer | RepairID, ServiceID | No | No | No | **Composite** | Auto-increment |
|  | Date | The date of the service | Datetime | ServiceDate | No | No | No | No | Current Date |
|  | Description | The description of the maintenance service (what was done) | Char(255) | ServiceDescription, ServiceDetails | No | No | No | No | None |
|  | partsUsed (multivalued) | The details on what parts were worked with | Char(255) | componentsUsed, componentsFixed | No | No | Yes | No | None |
|  | Cost | The cost of the maintenance service | Currency | ServiceCost, MaintenanceCost | No | No | No | No | 0.00 |
|  | nextDate | The date and time of when the next service/maintenance appointment will be | Datetime | NextService, NextMaintenanceDate | No | No | **Yes** | No | None |
| **Maintainer** | maintainerID | The unique identifier for every maintainer | Integer | ServiceProviderID | No | No | No | **Primary** | Auto-increment |
|  | Name | The name of the maintainer (organization) | Char(100) | OrganizationName | No | No | No | No | None |
|  | contactPerson | The name of the contact person for the maintainer | Char(100) | PrimaryContact | **Yes (firstName, lastName)** | No | No | No | None |
|  | Email | The email address of the maintainer | Char(255) | ContactEmail, ClientEmail | No | No | No | No | None |
|  | Phone | The phone number of the maintainer | Integer | ServicePhone, OrganizationPhone | No | No | No | No | None |
|  | Address | The physical address of the maintainer | Char(255) | MaintainerAddress, OrganizationAddress, Location | **Yes (street, city, state, postalCode, country)** | No | No | No | None |
|  | contractStartDate | The start of the maintenance contract | Datetime | serviceStartDate | No | No | No | No | Current Date |
|  | contractEndDate | The end of the maintenance contract | Datetime | serviceEndDate | No | No | Yes | No | None |
| **Data - weak** | dataID | The unique identifier for every dataset | Integer | DetailsID, ResultsID | No | No | No | **Composite** | Auto-increment |
|  | Timestamp | The date and time when the data was collected | Datetime | DataTime | No | No | No | No | Current date/time |
|  | Latitude | The latitude coordinate | Decimal(10,5) | Lat | No | No | No | No | None |
|  | Longitude | Longitude coordinate | Decimal(10,5) | Long | No | No | No | No | None |
|  | Altitude | Altitude written in meters | Decimal(10,5) | Alt | No | No | No | No | None |
|  | Temperature | The temperature of the environment (in Celsius) | Decimal (5,2) | Temp | No | No | No | No | None |
|  | Humidity | The percentage of humidity in the environment | Decimal (5,2) | HumidityLevel | No | No | No | No | None |
|  | lightMeasurement | The measurement of light | Decimal(10,2) | LightLevel | No | No | No | No | None |
|  | organicData | The description for the organic compounds (hydrogen, carbon, oxygen, nitrogen) | Char(255) | AirComposition | No | No | No | No | None |
| **Invoice - weak** | invoiceID | The unique identifier for every invoice bill | Integer | BillID | No | No | No | **Composite key** | Auto-increment |
|  | issueDate | The date the invoice was issued | Datetime | BillDate | No | No | No | No | Current date/time |
|  | dueDate | The date that the payment is due | Datetime | PaymentDue, PayDue | No | No | No | No | Current date + 30 days (month) |
|  | subtotalAmount | The payment amount before tax | Currency | BeforeTaxAmount, PreTaxAmount | No | No | No | No | 0.00 |
|  | taxAmount | The amount of tax | Currency | GST | No | **Yes (subtotalAmount \* 0.15)** | No | No | 0.00 |
|  | totalAmount | The total amount for the bill with tax | Currency | BillTotal, InvoiceTotal | No | **Yes (subtotalAmount + taxAmount)** | No | No | 0.00 |
|  | status | The status of the payment (if its been paid or if overdue) | Char(20) | PaymentStatus, PayCondition | No | No | No | No | ‘Unpaid’ |

# Rationale for Data Dictionary:

This dictionary includes definitions for every entity, attribute, relationship and key used in the Extended Entity Relationship Diagram that I had created.   
This dictionary focuses on how the data works, what it all means and how its all interconnected with each other.

Each entity was considered based on the business requirements and the processes of the system.

Entity:  
Each entity represented an object that is utilized within the system. Each entity is defined with the name, description, aliases (for variety of terminologies), occurrence details and key attributes. Identifying these helps with clarity throughout the system.

Weak Entities:  
Entities like ‘data’, ‘maintenance’ etc. are weak

entities as they depend on others (e.g., ‘data’ is dependent on a FRED drone). Utilizing this shows identification rules in the data model.

Attributes:  
The attributes describe the data traits for entities. Naming conventions (camelCase) and keys (primary key, foreign key, composite key) have been defined.  
The attributes have fields filled to determine if theyre composite, derived and could be nullified. Additional fields include the domains (INT, CHAR, DATETIME) and default values.

Relationships:  
All relationships between entities were defined through verb phrases (‘FRED *performs* SERVICE’), cardinality (M, M:N, 1:1 etc.) and participation (total/partial).   
Speciality/Generalization subtypes have been defined as well – for example, the superclass, ‘Employee’, has subtypes including ‘Salesperson’ and ‘Admin’ – these were showed utilizing a disjoint, which shows that an entity can belong to only one subtype (employee can be either a salesperson or admin, not both.) – as well as this, the U symbol indicates the subtype ‘ISA’ (is a) subset of the supertype.

This data dictionary utilizes the principles of effective data management through:

1. **Data Quality**

The names, aliases and descriptions show that the model supports consistent data entry.

1. **Scalability**

The dictionary allows for easy updates and changes – if a new entity is added in future development, following the structure will mean that its easy to visualize how it can be added into the system.

1. **Support for normalization**

The dictionary is useful as it identifies how the system could work to promote data integrity and overcomplication in the interface and database.

1. **Clarity**

By being able to document what each entity/attribute/relationship is and how it’s used, it ensures that designing the system is simple and that the stakeholders can understand how it’ll work. This reduces confusing when the database system is created.

# Assumed Business Rules (with reasoning):

1. **A region contains one or more zones**

Regions are bigger areas that can be subdivided into different zones. The reason for this is that it is easier to manage particular areas, which avoids redundant environment information and entries

1. **Directors hire admins, admins hire salespeople and engineers**

There is a set hierarchy for employing workers. The reason for this is to have business structure with set controls – this is beneficial for RBAC (role based access control)/

1. **An invoice is created for one contract**

A contract will eventually have a bill for the subscriber and the invoice is generated based on what had been included in the contract. It enables financial tracking and integration with the accounting systems.

1. **Maintainers perform maintenance based on a schedule**

Maintainers will be able to follow a schedule, which supports timely service and proper organization.

1. **FRED performs services and collects/generates data**

FREDs are responsible to perform services such as tree trimming, planting, track clearing etc.. Through doing the services, they capture data to create reports – this ensures automation in data generation.

1. **Contracts can operate within one or more zones (dependent on subscription)**

Services and data are generated based on the zones that are set – based on the subscription that the contract has set, one or more zones can be monitored and serviced by FREDs.

1. **Contracts are categorized by subscription type**

Every contract should have one subscription type assigned (standard, gold, platinum or super platinum). The categorization is necessary to showcase the additional features that might apply and to display any service level negotiations.

1. **A salesperson can place many contracts, but each contract is placed by one person**

One salesperson is responsible for closing each contract – this simplifies accountability. It helps prevent additional employees being set to the same contract, which isn’t necessary. One salesperson can place many contracts for different clients, however.

1. **An admin can negotiate many contract, but each contract is negotiated by one admin**

The admin finalizes contracts that had been placed by salespeople. Having the differentiation between what the admin does and what the salesperson does is necessary as it shows the hierarchy levels of the roles – the salesperson markets and initiates and the admin confirms. The contract cannot be finalized until the admin overlooks it.

1. **Subscription types define the contract pricing, service levels etc.**

Subscription tiers determine what the contract can have. This helps with service automation

1. **The maintenance application entry records maintainer usage and is used to store maintenance records**

The maintainers use an app to input the maintenance details and check the schedules – the application entry entity would record when the last session was for every maintainer and when they were active. This would facilitate monitoring the usage and accountability of a particular maintainer.

# Assignment 2 Part 1 – Logical Design:

# Conceptual to Relational Logical Model:

**How to transform a conceptual model (CHEN ERD) into a logical model:**

The difference between the conceptual model and logical model is that the conceptual model is a high-level representation of entities and their relationships. The conceptual model utilizes Chen notation, showcasing ovals for entities and diamonds for relationships, representing business concepts and requirements. The entities do showcase examples of some attributes, but it is limited as it’s not as necessary to showcase, in comparison to the logical model.

The logical model is more detailed as it considers how the entities will be implemented in the database. The diagrams uses ‘crows foot’ as one of the main visualizers for relationships, and presents entities using tables listed with primary keys, foreign keys and attributes.   
Unlike the conceptual model, this model resolves many-to-many relationships and generates a junction table to represent them.  
The logical model considers normalization principles as well, which will be mentioned further in the document.

To transform the conceptual model to logical model is as follows:

1. **Convert entities to tables**

Each entity in the conceptual model will then come a table in the logical model – for example, utilizing Visual Paradigm will allow you to create tables. By creating all the tables per entity at once, they can all be laid and mapped out to make it easier to visualize.   
It is then important to assign a primary key to each table depending on what it is called (for example, ‘Employee’ have a unique identifier corresponding to its name (EmployeeID)) and whether it’s a strong or weak entity.  
From this, we can then include all the attributes that are necessary for each table, including the data types (integer, varchar etc.), value number and constraints (NOT NULL, unique, check). By previously generating a data dictionary defining appropriate attributes, it simplifies the entire conversion process.

For handling attributes, it’s key to figure out that with multi-valued attributes, whether normalized structures would be more efficient or simply using separate tables.

1. **Handle relationship types**

One-to-One (1:1)

For a **one-to-one relationship**, it is described as **“one [table1] has one [table2]”**   
(i.e., one person has one passport).

To do this, decide which entity is more dependent between the two (however, you can choose either side, with consideration of the business’ context) and from there, place a foreign key in one table which references the primary key of the other. Additionally, you can add the ‘unique’ constraint of the foreign key column to insinuate that the ID can’t be duplicated.

One-to-Many (1:N)

For a **one-to-many relationship**, it is described as “**one [table1] has many [table2]”**   
(i.e., one company has many employees”.

For this, place a foreign key on the ‘many’ sides table, pointing in the direction of the primary key of the ‘one’ side. Just like the one-to-one relationship, there are no additional tables needed.

Many-to-Many (N:M)

For a **many-to-many relationship**, it is described as **“many [table1] has many [table2]”**   
(i.e., many products can be purchased by many customers).

This relationship utilizes a junction (associative) table, with two foreign keys referencing both tables involved. Additionally, it is optional to add any relationship attributes to the junction table.

1. **Figure out any weak entities**

A weak entity doesn’t have a primary key of its own, rather it “relies on a combination of its attributes and the primary key of the related strong entity” (Miro). If the conceptual model had any weak entities, consider adding the primary key of the strong entity as a foreign key and combine the foreign key and partial key as the composite primary key. Ensure referential integrity and total participation by including the ‘not null’ and ‘foreign key’ constraints.

1. **Implementing Inheritance/Specialization:**

Single Table:

If the conceptual model had one supertype with different subtypes, then when converting to a logical model, you create one big table, combining all the attributes from both supertype and subtypes.

Table per Concrete Entity:

If the conceptual model has one supertype with concrete subtypes, for the logical model, each concrete subtype gets its own table with all its corresponding attributes, including the shared ones from the supertype. No supertype table is created for this. This is only used when no shared behaviour must be stored centrally as these supertypes aren’t queried on its own.

Table per Entity:

If the conceptual had subtypes which are inherited from a shared supertype, then for the logical diagram, you create a table for the supertype and create separate tables for the different subtypes, storing attributes particular to the subtypes. The subtype tables would use the same primary key as the supertype and don’t have their own personal primary key identifier.

**Mapping rules:**

1. Entity Transformation

* Each **entity** in the conceptual model becomes a ‘class’ table in the logical model
* Each **attribute** from an entity becomes a ‘column’ in that specific table, represented by a ‘paper’ icon
* **Primary key** attributes are designated as the primary key of the table, with an icon of a ‘key’
* Entities that have need to acquire a primary key from another table need to reference it as a **foreign key** in the table, a column represented by a ‘paper’ icon with a green arrow

1. Relationship Transformation

* **One-to-One (1:1)** relationships are done by putting a foreign key in either of the tables, usually the table with the least amount of participations – however this might not always be necessary depending on the context.
* **One-to-Many (1:M)** relationships are done by placing the primary key of the ‘one’ side as a foreign key in the table on the ‘many’ side
* **Many-to-Many (M:N**) relationships are done by creating an intersection table which contains the primary keys from both related entities (mentioned as ‘primary/foreign’ keys) – e.g. supplier and part
* **Recursive** relationships require self-referencing foreign keys in the same table

1. Subtypes and Inheritance

* **Table Per Type**: Creating separate tables for supertype and the subtypes, with foreign key references from supertype table input into subtype
* **Table Per Hierarchy:** Creating one table with all attributes from supertype and all subtypes (subtype attributes are ‘nullable’, with the attribute corresponding to the chosen subtype typed and the rest are nulled)
* **Table Per Concrete**: Create separate tables for each subtype, each all including attributes from both supertype and specific subtype

1. Constraints

* **Primary key** integrity is enforced by primary key constraints
* **Referential** integrity is enforced by foreign key constraints
* **Nullable** constraints are set based on how many optional attributes are in the table
* **Domain** constraints are implemented as data types and check constraints

1. Other rules

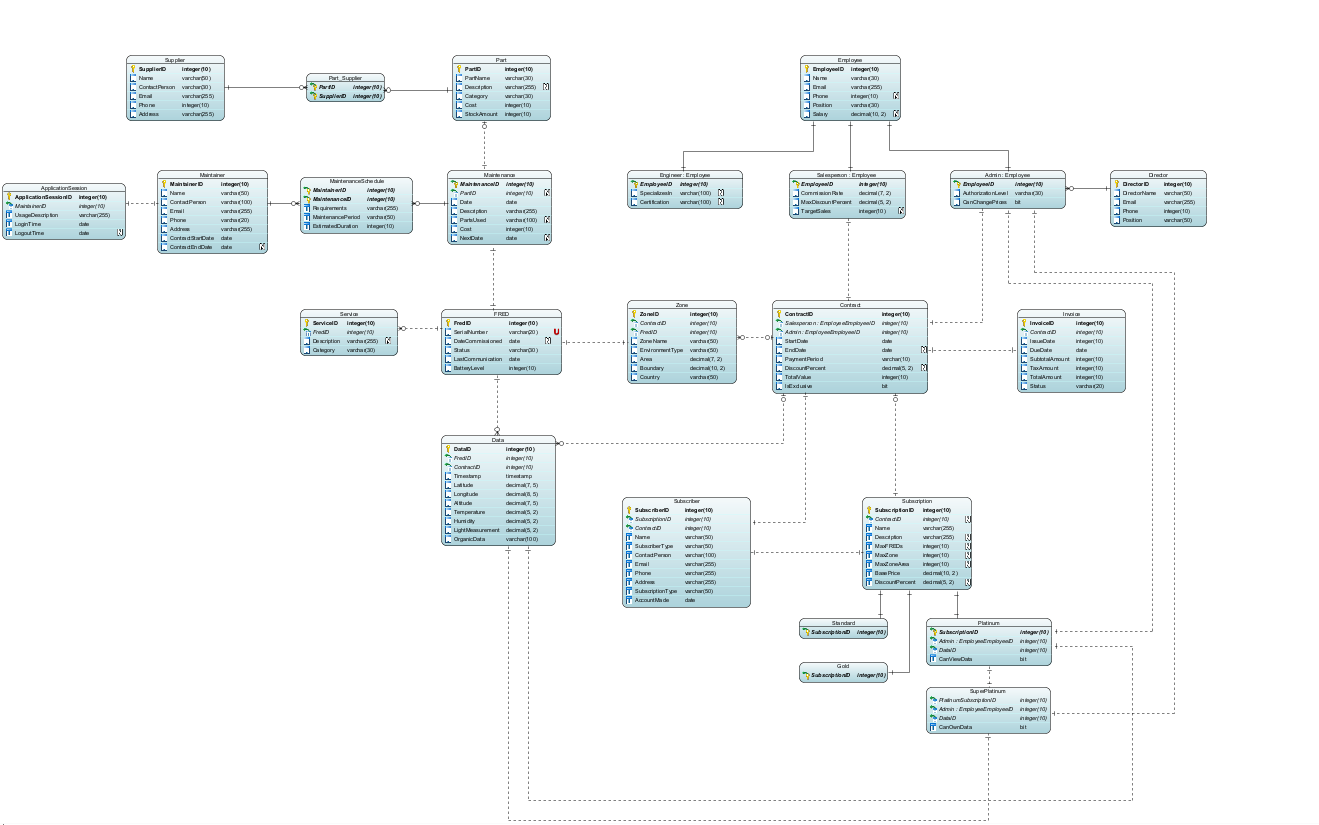
* **Multi-valued** attributes could require a separate table with a foreign key reference to the corresponding entitys table (could reference the attribute itself)
* **Composite** attributes are either put into multiple simple attributes or written as a single field

1. Normalization

After mapping, the tables should be looked at to check if normalization is up-to-date and proficient

* Utilize **1NF** by making sure all the attributes are atomic (meaning one that cannot be decomposed into meaningful components (Open Library, N.D)
* Utilize **2NF** by making sure theres no partial dependencies on the primary key
* Utilize **3NF** by making sure theres no transitive dependencies
* Utilize **4NF** for making sure theres no table with two or more independent or multivalued data of an entity

**ERD:**



[I will upload the VPP file into the existing Github link I have for this assignment – which will be linked into the dropbox]

**ERD Rationale:**

The database model represents the ParkWorks system that manages FRED (Field Remote Environmental Devices), their maintenance, collected data and other business operations.

This logical data model is a structured blueprint of the conceptual model, with applied normalization and data management principles to provide a scalable design. By converting the conceptual model into this logical form, it makes it easier to visualize the systems proper operational needs just before it is physically designed. Relationships are now expressed in a more formal way rather than descriptive words.

**Concepts of Data Modelling Include:**

1. Business Requirements:

This model follows the business rules and needs identified by the requirements for this system. Key requirements such as managing employees, maintaining FRED units, tracking contracts and subscriptions (including their subtypes) and recording maintenance are all captured in the presented normalized tables, with visualized relationships and related keys.

1. Clear Entity Relationships:

Each relationship between the entities is done using foreign keys, which proves I ensure referential integrity in the design. Following this, I ensured all tables have their respective primary key representing themselves. M:N relationships are resolved with junction tables (e.g., Part\_Supplier and MaintenanceSchedule).

1. Representation for subtypes

The model showcases inheritance through the ‘Table per Subtype’ (TPT) pattern. The inherited roles from ‘Employee’, such as ‘Salesperson, Engineer and Admin’, are all subtypes, which shows that the model has laid out a structure that ensures role-specific attributes are in their separate spots while maintaining relational integrity with the supertype, parent table (EmployeeID is used as FK and PK). This has also been applied for the ‘Subscription’ table, with the subtypes assigned to it (Standard, Gold, Platinum, SuperPlatinum)

1. Normalization

I have made sure my tables have been normalized up to 4NF, enabling the reduction of redundancy and making sure the model supports data integrity and maintainability of the schema.

|  |  |  |
| --- | --- | --- |
| Normal Form | Rule | IsApplied? |
| 1NF | All attributes should be atomic (nothing repeating) | Yes, all fields are atomic |
| 2NF | Every non-key attribute is fully dependent on the entire primary key | Yes, there are no composite primary keys and all non-key attribute depends on a single-column primary key. |
| 3NF | No non-key attribute depends on another non-key attribute | Yes, example attributes such as phone or email are tied to the entity and not to other non-primary key columns |
| Boyce-Codd | Every determinant (value) is a candidate key | Yes, theres no attribute that controls another attribute unless it’s a primary or candidate key |
| 4NF | No table contains two or more independent or multivalued data of an entity | Yes, attributes are handled using relational structure and junction tables for many-to-many relationships, which will prevent any violation. |

**Concepts of Data Management Principles Include:**

1. Reliability for the Schema

The layout of the model proves that any changes in the future can be modified without any complication, which makes it intuitive for developers, end-users etc.

1. Data Integrity

Each entity has their own defined primary key, alongside foreign keys added to maintain consistency across each table that relates with each other.   
Every column in each table are atomic, meaning theres no repetition anywhere in every table and all data types are appropriate for the attributes assigned to one.

**Staff**:

A screenshot of a computer

AI-generated content may be incorrect.

The ‘Employee’ class has 3 subtypes of itself, engineer, salesperson and admin.  
These subtypes each have a relationship with employee through a ‘one-to-one’ cardinality and inherit the employee’s primary key.   
Director is responsible for the admin and is not part of the employee scheme – “one director can hire many admin employees”.

**Subscription**:

A diagram of a program

AI-generated content may be incorrect.

Much like the employee table, subscription has their own subtypes, (utilizing the ‘Table Per Subtype”) which represents tiers that can be acquired for a subscription.  
Each subtype inherits the subscriptions primary key, yet only platinum and super platinum has their own attributes. The cardinality is represented through a ‘one-to-one’ relationship with the subscription superclass.

FRED:

A screenshot of a computer

AI-generated content may be incorrect.

This is a generalized view of the FRED, Service, Zone, Data, Contract and Invoice. We can see how each is interconnected with each other and how contract communicates with the data generated by FRED and can see which zones are utilized by FREDs.   
As well as this, we can see the contract is assigned to a subscriber, as well as linking itself to the subscription superclass.

Maintenance and Parts:

A diagram of a computer

AI-generated content may be incorrect.

Maintenance is conducted towards all FRED units. Maintenance is done by a maintainer company, who has access to an application and a schedule.   
Any parts used in the maintenance session can be ordered and stocked by a supplier. This is the section of the diagram where we see junction tables for M:N relationships, with one having their own relationship attributes. This shows I have utilized referential integrity.

**Description of normalization to 4th Normal form:**

Normalization is the process to improve the design of a database to meet rules for efficiency. The benefits of following a normalization guide to increase proficiency and productivity is that less data is considered: “With multiple smaller tables, theres less data for the database to look through and consider when returning results of a query, compared to tables with more rows and more columns” (POPSQL, 2023)

**1NF**

Requirements for 1NF:

1. The table must have a primary key
2. Each column should have a unique name
3. All attributes must contain atomic values (no repeating arrays/groups)
4. All columns should contain values of the same domain

Analysis of 1NF:

1. All the tables I have include a clearly defined primary key, which is listed as ID fields (e.g., DataID, FredID etc.)
2. All attributes store single values and aren’t repeated (e.g. Contact information are stored as separate attributes (email, phone, address)
3. Necessary data types are used
   1. Identifiers use ‘integer(10)’ appropriately
   2. Text fields use varchar with proper lengths
   3. Money values use decimal with proper lengths (10,2 or 5,2)
   4. Dates use the date data type

**2NF**

Requirements for 2NF:

1. Table must be in 1NF
2. No partial dependencies (where an attribute only depends on only one part of a composite key
3. No attributes depending solely on an entire primary key

Analysis of 2NF:

1. Most of my tables use a single-column primary key (FredID)
2. Many-to-many relationships are done with actual junction tables – for example, ‘Part\_Supplier’ table with composite key (PartID, SupplierID)

**3NF**

Requirements for 3NF:

1. Table must be in 2NF
2. Every non-key attribute must depend only on the primary key
3. Non-key attributes should not depend on other non-key attributes

Analysis of 3NF:

1. All my attributes depend on the table’s primary key (e.g. in the FRED table, attributes such as status and serial number depend only on FredID and in Employee table, Salary depends only on EmployeeID)
2. I don’t have any tables where one non-key attribute determines another (e.g. contact information depends directly on the entity IDs, not other attributes)

**BCNF:**

Requirement for BCNF:

1. Table must be in 3NF
2. Must remove functional dependencies

Analysis of BCNF:

1. All determinants in the model are candidate keys (In FRED table, only FredID determines the other attributes)
2. In my diagram theres no areas where a non-key attribute controls other attribute (in subscription tables, pricing and features are determined (in employee specializations, role-specific attributes depend only on one EmployeeID)

I have ensured that everything depends only on one the main thing that identifies the row

**4NF:**

Requirement for 4NF:

1. The tables must be in BCNF
2. No multi-valued dependencies not implied by candidate keys

Examples of analysis of 4NF:

1. Multi-valued dependencies are handled using proper table relationships (for example (subscription features by tier are modeled in separate tables (standard, gold etc.)
2. Independent multi-valued facts are separated into different relationships (e.g., Data measurements are organized by type in data entity – as well as this, the characteristics of one table is isolated from another (such as zone and contract having separate, isolated characteristics)
3. Different subscription tiers have independent multi-valued attributes segregated appropriately
4. Different role types in for employees have independent attributes properly segregated

**Denormalization in my diagram:**

1. Some of my calculated fields only exist for performance reasons – for example, TotalAmount in Invoice is calculated from the subtotal and tax attributes – this represents denormalization intentionally designed for efficiency

# Logical Data Dictionary:

**Document Relations:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Start Volume** | **Growth %** | **Comments** |
| **FRED** | **20** | **5-10 new FREDs per year** | **Small scale initially but more added slowly throughout the years with increase of production and demographic** |
| **Data** | **10000** | **50% per year** | **One FRED can generate more than one set of data rows a day** |
| **Maintenance** | **100** | **10% per year** | **Each FRED drone would need to have regular maintenance – growth would increase depending on how many new FREDs are built** |
| **Maintainer** | **5** | **No growth** | **The ParkWorks company would only need a few trusted maintenance providers** |
| **Supplier** | **5** | **>5% per year** | **The company would only need a few trusted suppliers.  Only growth would be if new suppliers are chosen in case parts become discontinued in the near future** |
| **Parts** | **200** | **5% per year** | **New parts might be added, and more stock would be ordered** |
| **Contract** | **15** | **20% per year** | **More contracts would be generated as the company expands** |
| **Subscription** | **15** | **10% per year** | **Clients only need one subscription so growth would only happen depending on how many new clients are found** |
| **Standard** | **5** | **Stable – no growth** | **Only a few would choose standard as it has the least number of features – not much growth is expected in comparison to ‘platinum’, for example** |
| **Gold** | **5** | **5% per year** | **Gold would have more demand than standard, but its moderate compared to platinum** |
| **Platinum** | **3** | **10% per year** | **This is an offer that is high-end and would be preferred, regardless of price as it provides more features** |
| **Super Platinum** | **2** | **>5% per year** | **Only a few select would choose this as this is very exclusive so minimal changes are expected** |
| **Zone** | **30** | **2-5% per year** | **Operational zones/delivery areas – growth depends on new areas** |
| **Service** | **50** | **10% per year** | **These are the types of services that are provided (e.g. scanning, imaging) and other jobs that FRED do** |
| **Invoice** | **200** | **15% per year** | **There would be one or more per contract/service** |
| **Employee** | **20** | **5% per year** | **Increase depending on new recruits** |

**Document Attributes:**

**FRED:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **FredID** | **The unique identifier for the FRED drone** | **Integer** | **10** | **Positive integers only** | **Must be unique and positive** | **Auto-increment** | **-** | **Primary key** | **-** | **Primary Key Integrity** |
| **SerialNumber** | **The manufacturer’s number for the specific FRED drone** | **Varchar** | **20** | **Any non-empty string** | **Must be unique and not null  Must not be longer than determined length size** | **-** | **-** | **UNIQUE** | **-** | **Domain integrity** |
| **DateCommissioned** | **The date that the drone was commissioned** | **Date** | **Date** | **Any valid past date** | **Must be today or a previous date** | **-** | **Yes** | **-** | **-** | **Domain integrity** |
| **Status** | **Status update on whether the drone is inactive, active etc.** | **Varchar** | **30** | **“Inactive”, “Active”,**  **“In Maintenance”** | **Must match the predefined values  Must not be longer than determined length size** | **Inactive** | **-** | **-** | **-** | **Domain integrity** |
| **LastCommunication** | **Date on when there was last contact for the drone** | **Date** | **Date** | **Any valid date** | **Cannot be in the future** | **Current date** | **-** | **-** | **-** | **Domain integrity** |
| **BatteryLevel** | **The battery level of the drone** | **Integer** | **10** | **0-100** | **Must be in between 0-100** | **100** | **-** | **-** | **-** | **Domain integrity** |

**Data:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **DataID** | **The unique identifier for the data entry** | **Integer** | **10** | **Positive integers only** | **Must be unique and positive** | **Auto-increment** | **-** | **Primary key** | **-** | **Primary key integrity – uniqueness** |
| **FredID** | **The foreign key reference to the FRED drone who inputted the data entry** | **Integer** | **10** | **Must exist in FRED table** | **Foreign key to FRED (FredID)** | **-** | **-** | **Foreign Key** | **Fred** | **Referential integrity – FredID in FRED table** |
| **ContractID** | **The foreign key reference for the contract assigned with the data** | **Integer** | **10** | **Must exist in Contract table** | **Foreign key to Contract (ContractID)** | **-** | **-** | **Foreign Key** | **Contract** | **Referential integrity – ContractID in Contract table** |
| **Timestamp** | **The timestamp of the data created** | **Timestamp** | **Timestamp** | **Must be valid timestamp** | **Cannot be future date** | **Current timestamp** | **-** | **-** | **-** | **Domain integrity** |
| **Latitude** | **The latitude of the FRED drone** | **Decimal** | **7,5** | **-90.00000 to 90.00000** | **Must be within global lat range and must follow predefined value** | **00.00000** | **-** | **-** | **-** | **Domain integrity** |
| **Longitude** | **The longitude of the FRED drone** | **Decimal** | **8,5** | **-180.00000 to 180.00000** | **Must be within global long range and must follow predefined value** | **000.00000** | **-** | **-** | **-** | **Domain integrity** |
| **Altitude** | **The altitude of the FRED drone** | **Decimal** | **4,1** | **> 0 or as per region**  **000.0** | **Must follow predefined value** | **000.0** | **-** | **-** | **-** | **Domain integrity** |
| **Temperature** | **The temperature of the environment** | **Decimal** | **5,2** | **Practical temperature values in Celsius**  **000.00-999.99** | **e.g. -50°C to 100°C**  **Must follow predefined value** | **000.00** | **-** | **-** | **-** | **Domain integrity** |
| **Humidity** | **The humidity of the environment** | **Decimal** | **5,2** | **000.00-999.99** | **Must follow predefined value** | **000.00** | **-** | **-** | **-** | **Domain integrity** |
| **LightMeasurement** | **The light measurement of the environment** | **Decimal** | **5,2** | **000.00-999.99** | **Must follow predefined value** | **000.00** | **-** | **-** | **-** | **Domain integrity** |
| **OrganicData** | **The organic data entry of the environment** | **Varchar** | **100** | **Any text string** | **Free text but not null  Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |

**Maintenance:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **MaintenanceID** | **The unique identifier for the maintenance service** | **Integer** | **10** | **Positive integers only** | **Must be unique and positive** | **Auto-increment** | **-** | **Primary key** | **-** | **Primary key integrity** |
| **PartID** | **The foreign key referencing the part used (if applicable)** | **Integer** | **10** | **Must exist in Part table** | **Foreign key to Part (PartID)** | **- NULLABLE** | **Yes** | **Foreign Key** | **Part** | **Referential integrity – PartID in Part table** |
| **Date** | **The date of the maintenance session** | **Date** | **Date** | **Any valid date** | **Must be todays date or in the past** | **Current date** | **-** | **-** | **-** | **Domain integrity** |
| **Description** | **The description of what happened during the maintenance session** | **Varchar** | **255** | **Any characters** | **Not empty  Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **PartsUsed** | **The parts that may have been used during the maintenance session** | **Varchar** | **100** | **Any characters** | **Optional   Must not be longer than determined length size** | **-** | **Yes** | **-** | **Part** | **Domain integrity** |
| **Cost** | **The cost of the maintenance** | **Decimal** | **5,2** | **000.00** | **Must follow predefined value** | **000.00** | **-** | **-** | **-** | **Domain integrity** |
| **NextDate** | **The next date that might occur for the maintenance** | **Date** | **Date** | **Any future date** | **Must be after ‘Date’** | **Current date + 30 days** | **Yes** | **-** | **-** | **Domain integrity** |

**Part:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **PartID** | **The unique identifier for the part** | **Integer** | **10** | **Positive integers only** | **Must be unique and positive** | **Auto-increment** | **-** | **Primary key** | **-** | **Primary key integrity** |
| **PartName** | **The name of the part** | **Varchar** | **30** | **Any text** | **Cannot be blank or null** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **Description** | **The description of what the part is and how it’s used** | **Varchar** | **255** | **Any text** | **Optional  Must not be longer than determined length size** | **NULL** | **Yes** | **-** | **-** | **Domain integrity** |
| **Category** | **The category that the part falls in** | **Varchar** | **30** | **Free text** | **Cannot be null  Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **Cost** | **The cost of the part** | **Decimal** | **5,2** | **000.00** | **Must follow predetermined value** | **000.00** | **-** | **-** | **-** | **Domain integrity** |
| **StockAmount** | **The amount of stock left for the part** | **Integer** | **10** | **0 and above** | **Must be > = 0** | **0** | **-** | **-** | **-** | **Domain integrity** |

**Maintainer:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **MaintainerID** | **The unique identifier for the maintenance assistant who conducts maintenance sessions** | **Integer** | **10** | **Positive integers only** | **Must be unique and positive** | **Auto-increment** | **-** | **Primary key** | **-** | **Primary key integrity** |
| **Name** | **The name of the maintenance company** | **Varchar** | **50** | **Any text** | **Cannot be null or empty**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **ContactPerson** | **The name of the person who represents the company** | **Varchar** | **100** | **Any text** | **Cannot be null or empty**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **Email** | **The email of the company** | **Varchar** | **255** | **Valid email format** | **Must follow email format** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **Phone** | **The phone number of the company** | **Varchar** | **20** | **Valid phone format** | **Must follow predetermined value** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **Address** | **The companies’ address** | **Varchar** | **255** | **Any text** | **Cannot be empty**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **ContractStartDate** | **The start date of the contract** | **Date** | **Date** | **Any valid date** | **Must be today or past date** | **Current date** | **-** | **-** | **-** | **Domain integrity** |
| **ContractEndDate** | **The end date of the contract** | **Date** | **Date** | **Any valid date in future** | **Must be future date if applicable** | **-** | **Yes** | **-** | **-** | **Domain integrity** |

**MaintenanceSchedule:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **MaintainerID** | **Foreign key reference** | **Integer** | **10** | **Positive integers only – must exist in Maintainer table** | **Must be unique and positive – foreign key to Maintainer (MaintainerID)** | **Auto-increment** | **-** | **Foreign key** | **Maintainer** | **Primary key integrity** |
| **MaintenanceID** | **Foreign key reference** | **Integer** | **10** | **Positive integers only – must exist in Maintenance table** | **Must be unique and positive – foreign key to Maintenance (MaintenanceID)** | **-** | **-** | **Foreign key** | **Maintenance** | **Referential integrity – MaintenanceID in Maintenance table** |
| **Requirements** | **The requirements necessary to fix the drone** | **Varchar** | **255** | **Any text** | **Cannot be empty or null**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **MaintenancePeriod** | **The maintenance period** | **Varchar** | **50** | **‘Weekly’, ‘Monthly’, ‘Annually’** | **Must follow predefined values**  **Must not be longer than determined length size** | **Monthly** | **-** | **-** | **-** | **Domain integrity** |
| **EstimatedDuration** | **The estimated duration in minutes (e.g. 60 minutes = 1 hour)** | **Integer** | **10** | **0-320 (minutes)** | **Cannot be less than 0** | **0** | **-** | **-** | **-** | **Domain integrity** |

**Part\_Supplier:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **PartID** | **A foreign key reference to the part table** | **Integer** | **10** | **Positive integers only – must exist in Part table** | **Must be unique and positive – foreign key to Part (PartID)** | **Auto-increment** | **-** | **Primary key/Foreign Key** | **Part** | **Primary key/Referential integrity (PK/FK)  PartID in Part table** |
| **SupplierID** | **A foreign key reference to the part table** | **Integer** | **10** | **Positive integers only – must exist in Supplier table** | **Must be unique and positive – foreign key to Supplier (SupplierID)** | **Auto-increment** | **-** | **Primary key/Foreign Key** | **Supplier** | **Primary key/Referential integrity (PK/FK)   SupplierID in Supplier table** |

**Supplier:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **SupplierID** | **The unique identifier for the supplier** | **Integer** | **10** | **Positive integers only** | **Must be unique and positive** | **Auto-increment** | **-** | **Primary key** | **-** | **Primary Key Integrity** |
| **Name** | **The name of the supplier company** | **Varchar** | **50** | **Any text** | **Must not be empty or null**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **ContactPerson** | **The name of the contact person representing the company** | **Varchar** | **30** | **Any text** | **Must not be empty or null**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **Email** | **The email of the company** | **Varchar** | **255** | **Valid email format** | **Must suit the format for email @  and .com/.co.nz/govt.nz etc.**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **Phone** | **The phone number of the company** | **Varchar** | **20** | **Valid phone number format** | **Must suit the format for phone numbers** | **-** | **Yes** | **-** | **-** | **Domain integrity** |
| **Address** | **The address of the company** | **Varchar** | **255** | **Any text** | **Must not be null or empty and should include at least a street number, name, suburb, city and post code.**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |

**Application Session:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **ApplicationSessionID** | **The unique identifier for the application session made by a maintainer** | **Integer** | **10** | **Positive integers only** | **Must be unique and positive** | **Auto-increment** | **-** | **Primary key** | **-** | **Primary Key Integrity** |
| **MaintainerID** | **A foreign key reference to the maintainer logging a session** | **Integer** | **10** | **Positive integers only – must exist in Maintainer table** | **Must be unique and positive – foreign key to Maintainer (MaintainerID)** | **-** | **-** | **Foreign Key** | **Maintainer** | **Referential integrity – MaintainerID in Maintainer table** |
| **UsageDescription** | **Description of how the application was used** | **Varchar** | **255** | **Any text** | **Must not be empty or null**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **LoginTime** | **The login time that the maintainer access it** | **Date** | **Date** | **Must be valid date time** | **Must be today or in past** | **Current date** | **-** | **-** | **-** | **Domain integrity** |
| **LogoutTime** | **The logout time that the maintainer logged out of the application** | **date** | **Date** | **Must be valid date time** | **Must be date time in the future** | **-** | **Yes** | **-** | **-** | **Domain Integrity** |

**Service:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **ServiceID** | **The unique identifier for the supplier** | **Integer** | **10** | **Positive integers only** | **Must be unique and positive** | **Auto-increment** | **-** | **Primary key** | **-** | **Primary Key Integrity** |
| **FredID** | **A foreign key reference to the FRED table** | **Integer** | **10** | **Positive integers only – must exist in FRED table** | **Must be unique and positive – foreign key to Fred (FredID)** | **-** | **-** | **Foreign Key** | **FRED** | **Referential integrity – FredID in FRED table** |
| **Description** | **The description of what the service the FRED is carrying out** | **Varchar** | **255** | **Any text** | **Optional**  **Must not be longer than determined length size** | **-** | **Yes** | **-** | **-** | **Domain integrity** |
| **Category** | **The type of service done** | **Varchar** | **30** | **Any text** | **Must not be empty or null**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |

**Employee**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **EmployeeID** | **The unique identifier for the** | **Integer** | **10** | **Positive integers only** | **Must be unique and positive** | **Auto-increment** | **-** | **Primary key** | **-** | **Primary Key Integrity** |
| **Name** | **The name of the employee** | **Varchar** | **30** | **Any text** | **Must not be empty or null**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **Email** | **The employees email address** | **Varchar** | **255** | **Valid email format** | **Must follow email format**  **@**  **.com/.org/.co.nz**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **Phone** | **The employees phone number** | **Varchar** | **20** | **Digits only – valid phone format** | **Must follow valid phone number - optional** | **-** | **Yes** | **-** | **-** | **Domain integrity** |
| **Position** | **The employees position** | **Varchar** | **30** | **Any text** | **Must not be empty or null**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **Salary** | **The salary of the employee** | **Decimal** | **10, 2** | **00000000.00 – 99999999.99** | **Optional** | **0.00** | **Yes** | **-** | **-** | **Domain integrity** |

**Engineer**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **EmployeeID** | **A foreign key reference to the Employee** | **Integer** | **10** | **Positive integers only – must exist in Employee table** | **Must be unique and positive – foreign key to Employee (EmployeeID)** | **Auto-increment** | **-** | **Primary key/Foreign key** | **Employee** | **Primary Key/Referential Integrity (PK/FK) EmployeeID in Employee table** |
| **SpecializesIn** | **A description explaining what the engineer specializes in** | **Varchar** | **100** | **Any text** | **Optional**  **Must not be longer than determined length size** | **-** | **Yes** | **-** | **-** | **Domain integrity** |
| **Certification** | **A description of any certifications the engineer has** | **Varchar** | **100** | **Any text** | **Optional**  **Must not be longer than determined length size** | **-** | **Yes** | **-** | **-** | **Domain integrity** |

**Salesperson**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **EmployeeID** | **A foreign key reference to the Employee** | **Integer** | **10** | **Positive integers only – must exist in Employee table** | **Must be unique and positive – foreign key to Employee (EmployeeID)** | **Auto-increment** | **-** | **Primary key/Foreign key** | **Employee** | **Primary Key/Referential Integrity (PK/FK) EmployeeID in Employee table** |
| **CommissionRate** | **Referring to the commission rate the salesperson follows** | **Decimal** | **7,2** | **00000.00 – 99999.99** | **Must be within predefined value parameters** | **00000.00** | **-** | **-** | **-** | **Domain integrity** |
| **MaxDiscountPercent** | **Represents the max discount percent a salesperson can apply for a contract** | **Integer** | **5,2** | **000.00% - 100.00%** | **Must be within 0 to 100** | **000.00** | **-** | **-** | **-** | **Domain integrity** |
| **TargetSales** | **The target sales the salesperson might have** | **Integer** | **10** | **0 - 9999999999** | **Must be within predefined value parameters - optional** | **0** | **Yes** | **-** | **-** | **Domain integrity** |

**Admin:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **EmployeeID** | **A foreign key reference to the Employee** | **Integer** | **10** | **Positive integers only – must exist in Employee table** | **Must be unique and positive – foreign key to Employee (EmployeeID)** | **Auto-increment** | **-** | **Primary key/Foreign key** | **Employee** | **Primary Key/Referential Integrity (PK/FK) EmployeeID in Employee table** |
| **AuthorizationLevel** | **Describes the level of authorization the admin has** | **Varchar** | **30** | **Any text** | **Must not be empty or null**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **CanChangePrices** | **The Boolean/bit that determines if the administrator has the right to change prices** | **Bit** | **Bit** | **0 or 1 0 = no 1 = yes** | **Must fit the predetermined value** | **0** | **-** | **-** | **-** | **Domain integrity** |

**Director**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **DirectorID** | **The unique identifier for the** | **Integer** | **10** | **Positive integers only** | **Must be unique and positive** | **Auto-increment** | **-** | **Primary key** | **-** | **Primary Key Integrity** |
| **DirectorName** | **The name of the director** | **Varchar** | **50** | **Any text** | **Must not be empty or null**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **Email** | **The director’s email** | **Varchar** | **255** | **Valid email format** | **Must follow email format**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **Phone** | **The phone number of the director** | **Varchar** | **20** | **Digits only – valid phone format** | **Must follow valid phone number** | **-** | **Yes** | **-** | **-** | **Domain integrity** |
| **Position** | **The directors’ position** | **Varchar** | **50** | **Any text** | **Must not be empty or null**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |

**Contract:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **ContractID** | **The unique identifier for the** | **Integer** | **10** | **Positive integers only** | **Must be unique and positive** | **Auto-increment** | **-** | **Primary key** | **-** | **Primary Key Integrity** |
| **Salesperson: EmployeeID** | **A foreign key reference to the Salesperson inherited from Employee** | **Integer** | **10** | **Positive integers only – must exist in Salesperson: Employee table** | **Must be unique and positive – foreign key to Salesperson: Employee (Salesperson : EmployeeID)** | **-** | **-** | **Primary key/Foreign key** | **Salesperson : Employee** | **Primary Key/Referential Integrity (PK/FK) Saleperson : EmployeeID in Employee table** |
| **Admin: EmployeeID** | **A foreign key reference to the Admin inherited from Employee** | **Integer** | **10** | **Positive integers only – must exist in Admin: Employee table** | **Must be unique and positive – foreign key to Admin: Employee (Admin : EmployeeID)** | **-** | **-** | **Primary key/Foreign key** | **Admin : Employee** | **Primary Key/Referential Integrity (PK/FK) Admin : EmployeeID in Employee table** |
| **StartDate** | **The date where the contract started** | **Date** | **Date** | **Any valid date** | **Must be today or past date** | **Current Date** | **-** | **-** | **-** | **Domain integrity** |
| **EndDate** | **The date where the contract ended** | **Date** | **Date** | **Any valid date in future** | **Must be future date if applicable** | **-** | **Yes** | **-** | **-** | **Domain integrity** |
| **PaymentPeriod** | **The payment period of the contract** | **Varchar** | **10** | **“Weekly’, ‘Monthly’, ‘Annually’** | **Must follow the predefined value** | **Monthly** | **-** | **-** | **-** | **Domain integrity** |
| **DiscountPercent** | **The discount percentage for the payment applied** | **Decimal** | **5,2** | **000.00% - 999.99%** | **Must follow the predefined value** | **000.00** | **Yes** | **-** | **-** | **Domain integrity** |
| **TotalValue** | **The total value price of the contract itself** | **Decimal** | **5,2** | **00000.00** | **Must follow the predetermined values** | **00000.00** | **-** | **-** | **-** | **Domain integrity** |
| **IsExclusive** | **A Boolean/bit that determines if the contract is exclusive or not** | **Bit** | **Bit** | **0 or 1**  **0 = no**  **1 = yes** | **Must fit the bit predefined value** | **0** | **-** | **-** | **-** | **Domain integrity** |

**Invoice:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **InvoiceID** | **The unique identifier for the** | **Integer** | **10** | **Positive integers only** | **Must be unique and positive** | **Auto-increment** | **-** | **Primary key** | **-** | **Primary Key Integrity** |
| **ContractID** | **A foreign key reference to the Contract** | **Integer** | **10** | **Positive integers only – must exist in Contract table** | **Must be unique and positive – foreign key to Contract (ContractID)** | **-** | **-** | **Foreign key** | **Contract** | **Referential Integrity -ContractID in Contract table** |
| **IssueDate** | **The date the invoice was issued** | **Date** | **Date** | **Any valid date** | **Must be today or past date** | **Current date** | **-** | **-** | **-** | **Domain integrity** |
| **DueDate** | **The date that the payment is due** | **Date** | **Date** | **Any valid date in future** | **Must be future date if applicable** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **SubtotalAmount** | **The payment amount before tax** | **Decimal** | **10, 2** | **00000000.00 – 99999999.99** | **Must follow the predefined value standard** | **0.00** | **-** | **-** | **-** | **Domain integrity** |
| **TaxAmount** | **The amount of tax applied** | **Decimal** | **10, 2** | **00000000.00 – 99999999.99  subtotalAmount \* 0.15** | **Must follow the predefined value standard** | **0.00** | **-** | **-** | **-** | **Domain integrity** |
| **TotalAmount** | **The total amount for the bill with tax** | **Decimal** | **10,2** | **00000000.00 – 99999999.99**  **subtotalAmount + TaxAmount** | **Must follow the predefined value standard** | **0.00** | **-** | **-** | **-** | **Domain integrity** |
| **Status** | **The status of the payment (if it’s paid or overdue)** | **Varchar** | **20** | **‘Not paid’, ‘Paid’, ‘Overdue’, ‘Canceled’** | **Must follow the predefined value standard**  **Must not be longer than determined length size** | **Not paid** | **-** | **-** | **-** | **Domain integrity** |

**Zone:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **ZoneID** | **The unique identifier for the** | **Integer** | **10** | **Positive integers only** | **Must be unique and positive** | **Auto-increment** | **-** | **Primary key** | **-** | **Primary Key Integrity** |
| **ContractID** | **A foreign key reference to the Contract** | **Integer** | **10** | **Positive integers only – must exist in Contract table** | **Must be unique and positive – foreign key to Contract (ContractID)** | **-** | **-** | **Foreign key** | **Contract** | **Referential Integrity -ContractID in Contract table** |
| **FredID** | **A foreign key reference to FREDs** | **Integer** | **10** | **Positive integers only – must exist in FRED table** | **Must be unique and positive – foreign key to FRED (FredID)** | **-** | **-** | **Foreign key** | **FRED** | **Referential Integrity -FredID in FRED table** |
| **ZoneName** | **The name of the zone that a contract/FRED is assigned to** | **Varchar** | **50** | **Any text** | **Must not be empty or null**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **EnvironmentType** | **The type of environment that the zone has** | **Varchar** | **50** | **Any text** | **Must not be empty or null**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **Area** | **The size of the region in hectares** | **Decimal** | **7, 2** | **00000.00-99999.99** | **Must follow predetermined value** | **00000.00** | **-** | **-** | **-** | **Domain integrity** |
| **Boundary** | **The boundary coordinates** | **Decimal** | **10,2** | **00000000.00 – 99999999.99** | **Must follow predetermined value** | **00000000.00** | **-** | **-** | **-** | **Domain integrity** |
| **Country** | **The country where the zone is** | **Varchar** | **50** | **Any text** | **Must not be empty or null**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |

**Subscriber:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **SubscriberID** | **The unique identifier for the** | **Integer** | **10** | **Positive integers only** | **Must be unique and positive** | **Auto-increment** | **-** | **Primary key** | **-** | **Entity integrity – uniqueness** |
| **SubscriptionID** | **A foreign key reference to the Subscription** | **Integer** | **10** | **Positive integers only – must exist in Subscription table** | **Must be unique and positive – foreign key to Subscription (SubscriptionID)** | **-** | **-** | **Foreign key** | **Contract** | **Referential Integrity -ContractID in Contract table** |
| **ContractID** | **A foreign key reference to the Contract** | **Integer** | **10** | **Positive integers only – must exist in Contract table** | **Must be unique and positive – foreign key to Contract (ContractID)** | **-** | **-** | **Foreign key** | **Contract** | **Referential Integrity -ContractID in Contract table** |
| **Name** | **The name of the subscriber company/organization** | **Varchar** | **50** | **Any text** | **Must not be empty or null**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **SubscriberType** | **Whether the subscriber is a private company etc.** | **Varchar** | **50** | **Any text** | **Must not be empty or null**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **ContactPerson** | **The name of the contact person assigned by the company** | **Varchar** | **100** | **Any text** | **Must not be empty or null**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **Email** | **The email of the company** | **Varchar** | **255** | **Valid email format** | **Must suit the format for email @  and .com/.co.nz/govt.nz etc.**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **Phone** | **The phone number of the company** | **Varchar** | **20** | **Valid phone number format** | **Must suit the format for phone numbers** | **-** | **Yes** | **-** | **-** | **Domain integrity** |
| **Address** | **The address of the company** | **Varchar** | **255** | **Any text** | **Must not be null or empty and should include at least a street number, name, suburb, city and post code.**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **SubscriptionType** | **The type of subscription they have purchased** | **Varchar** | **30** | **“Standard, Gold, Platinum, SuperPlatinum”** | **Must follow predetermined values** | **Standard** | **-** | **-** | **-** | **Domain integrity** |
| **AccountMade** | **The date of when the account was created** | **date** | **Date** | **Any valid date (not in future)** | **Must be current date or past date** | **Current Date** | **-** | **-** | **-** | **Domain integrity** |

**Subscription:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **SubscriptionID** | **The unique identifier for the** | **Integer** | **10** | **Positive integers only** | **Must be unique and positive** | **Auto-increment** | **-** | **Primary key** | **-** | **Entity integrity – uniqueness** |
| **ContractID** | **A foreign key reference to the Contract** | **Integer** | **10** | **Positive integers only – must exist in Contract table** | **Must be unique and positive – foreign key to Contract (ContractID)** | **-** | **Yes** | **Foreign key** | **Contract** | **Referential Integrity -ContractID in Contract table** |
| **Name** | **The name of the subscription type** | **Varchar** | **30** | **Any text** | **Must not be empty or null**  **Must not be longer than determined length size** | **-** | **-** | **-** | **-** | **Domain integrity** |
| **Description** | **The description of the subscription** | **Varchar** | **255** | **Any text** | **Optional**  **Must not be longer than determined length size** | **-** | **Yes** | **-** | **-** | **Domain integrity** |
| **MaxFREDs** | **The maximum amount of FREDs that the subscription can have and deploy (depending on the contract)** | **Integer** | **10** | **0000000000- 9999999999** | **Must follow predetermined value** | **0** | **Yes** | **-** | **-** | **Domain integrity** |
| **MaxZone** | **The maximum amount of zones that the subscription type can have – there are limits based on the tier type** | **Integer** | **10** | **0000000000-999999999999** | **Must follow predetermined value** | **0** | **Yes** | **-** | **-** | **Domain integrity** |
| **MaxZoneArea** | **The maximum area in sqm that the zone can have** | **Integer** | **10** | **0000000000-9999999999** | **Must follow predetermined value** | **0** | **Yes** | **-** | **-** | **Domain integrity** |
| **BasePrice** | **The base price of the subscription** | **Decimal** | **10,2** | **00000000.00 – 99999999.99** | **Must follow predetermined value** | **0.00** | **-** | **-** | **-** | **Domain integrity** |
| **DiscountPercent** | **The full discount percent that’s applied for the subscription** | **Decimal** | **5,2** | **000.00% - 999.99%** | **Must follow predetermined value** | **000.00** | **Yes** | **-** | **-** | **Domain integrity** |

**Standard**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **SubscriptionID** | **A foreign key reference to Subscription** | **Integer** | **10** | **Positive integers only – must exist in Subscription table** | **Must be unique and positive – foreign key to Subscription (SubscriptionID)** | **Auto-increment** | **-** | **Foreign key** | **Subscription** | **Primary Key Integrity + Referential Integrity -SubscriptionID in Subscription table** |

**Gold**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **SubscriptionID** | **A foreign key reference to Subscription** | **Integer** | **10** | **Positive integers only – must exist in Subscription table** | **Must be unique and positive – foreign key to Subscription (SubscriptionID)** | **Auto-increment** | **-** | **Foreign key** | **Subscription** | **Primary Key Integrity + Referential Integrity -SubscriptionID in Subscription table** |

**Platinum**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **SubscriptionID** | **A foreign key/primary key reference to Subscription** | **Integer** | **10** | **Positive integers only – must exist in Subscription table** | **Must be unique and positive – foreign key to Subscription (SubscriptionID)** | **Auto-increment** | **-** | **Foreign key** | **Subscription** | **Primary Key Integrity + Referential Integrity -SubscriptionID in Subscription table** |
| **DataID** | **A foreign key reference to Data (DataID) as the platinum subscription can view data generated by FREDs** | **Integer** | **10** | **Positive integers only – must exist in Data table** | **Must be unique and positive – foreign key to Data (DataID)** | **-** | **-** | **Foreign Key** | **Data** | **Referential Integrity – DataID in Data table** |
| **Admin : EmployeeID** | **A foreign key reference to Admin (Admin : EmployeeID) as they can negotiate the prices of this subscription** | **Integer** | **10** | **Positive integers only – must exist in Employees subtype table, Admin** | **Must be unique and positive – foreign key to Employee (Admin : EmployeeID)** | **-** | **-** | **Foreign key** | **Admin : Employee** | **Referential Integrity – Admin : EmployeeID in Employees subtype table, Admin** |
| **CanViewData** | **A bit describing if the subscription ‘can view data’ (by default it can, but they may not have accessed it yet)** | **Bit** | **Bit** | **0 or 1 (No or Yes)** | **Must follow predetermined value** | **0** | **-** | **-** | **-** | **Domain Integrity** |

**SuperPlatinum**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Description** | **Data Type** | **Length** | **Value Range** | **Validation Rules** | **Default Value** | **Nulls** | **Keys** | **Entity Ref** | **Integrity** |
| **Platinum SubscriptionID** | **A foreign key reference to Subscription** | **Integer** | **10** | **Positive integers only – must exist in Subscription table** | **Must be unique and positive – foreign key to Subscription (SubscriptionID)** | **Auto-increment** | **-** | **Foreign key** | **Subscription** | **Primary Key Integrity + Referential Integrity -SubscriptionID in Subscription table** |
| **DataID** | **A foreign key reference to Data (DataID) as the platinum subscription can view data generated by FREDs** | **Integer** | **10** | **Positive integers only – must exist in Data table** | **Must be unique and positive – foreign key to Data (DataID)** | **-** | **-** | **Foreign Key** | **Data** | **Referential Integrity – DataID in Data table** |
| **Admin : EmployeeID** | **A foreign key reference to Admin (Admin : EmployeeID) as they can negotiate the prices of this subscription** | **Integer** | **10** | **Positive integers only – must exist in Employees subtype table, Admin** | **Must be unique and positive – foreign key to Employee (Admin : EmployeeID)** | **-** | **-** | **Foreign key** | **Admin : Employee** | **Referential Integrity – Admin : EmployeeID in Employees subtype table, Admin** |
| **CanOwnData** | **A bit describing if the subscription ‘can own data’ (by default it can, but they may not have accessed it yet)** | **Bit** | **Bit** | **0 or 1 (No or Yes)** | **Must follow predetermined value** | **0** | **-** | **-** | **-** | **Domain Integrity** |

**Data Dictionary Rationale:**

The data dictionary shows an organized approach to understanding the structure and use of data within the system. This dictionary defines each entity and their attributes, providing definitions which help provide understanding for the development of the database.

**Concepts of Data Modelling:**

1. Normalization Support

Using a data dictionary isolates each entity into their respective tables, which supports normalization since each attribute is store where its needed, making it easier to also visualize consistency

1. Clear Entity/Attribute Definitions

The dictionary clearly identifies each entity table and their attributes, with descriptive columns to provide generous context.

1. Alignment with the ER model

The layout of the dictionary allows an easy way to translate it into an ERD and database design. The relationships are shown through the foreign keys, which also helps with referential integrity

**Concepts of Data Management Principles include:**

1. Data Accessibility

The use of these detailed descriptions for the attributes in each entity makes usability more approachable for developers and end-users.

1. Data Accuracy

Utilizing a data dictionary like this ensures each attribute definition is particular and accurate, making sure it’s absolutely clear on what is expected for the database. This improves how reliable the data entries will be throughout the entire system.

# NaLER Analysis:

**Step 1: Diagram Notation Legend**

|  |  |
| --- | --- |
| Symbol/Element | Description |
| Rectangle box | Entity/Class with the name at the top |
| Key symbol | A primary key attribute – represents unique identifiers for each entity |
| Paper with green arrow | A foreign key attribute – represents references to primary keys from another table |
| Paper | An attribute |
| Continuous line with crows’ foot (--<) | One-to-many relationship |
| Solid line with triangle (--▷) | Specialization/Inheritance relationship |
| Dotted line (- - -) | Association relationship |
| N | Nullable attribute |
| U | Unique attribute |
| Attribute name | Data type (size) |

**Step 2: Syntax Check Based on Legend**

1. **Missing relationship labels:**

* Relationships between entities don’t have verb phrases to describe the relationship

1. **Relationship notation not consistent**

* some relationships use crows foot notation and others don’t
* some don’t show cardinality

**Step 3: Entity Analysis with Sentences:**

**FRED**

|  |  |  |
| --- | --- | --- |
| Primary Key | Attributes | Relationships |
| Each FRED is uniquely identified by **FredID** | - Each FRED has a **serial number**  - Each FRED has a **date** **commissioned**  - Each FRED has a **status**  - Each FRED has a **last** **communication** **date**  - Each FRED has a **battery** **level** | - Each FRED is associated with one or more **services**  - Each FRED is associated with one **zone**  - Each FRED generates more than one set row of **data**  - Each FRED undergoes **maintenance** |

**Service**

|  |  |  |  |
| --- | --- | --- | --- |
| Primary Key | Foreign Key | Attributes | Relationships |
| Each Service is uniquely identified by ServiceID | A service is associated with one FRED (FredID) | - Each service has a **description (**optional)  - Each service has a **category** | Each service is carried out by a FRED |

**Maintenance**

|  |  |  |  |
| --- | --- | --- | --- |
| Primary Key | Foreign Key | Attributes | Relationships |
| Each maintenance is uniquely identified by MaintenanceID | A maintenance session may be associated with a Part (PartID) (optional/nullable) | - Each maintenance has a **date** - Each maintenance has a **description** - Each maintenance might have **parts used** (optional) - Each maintenance has a **cost applied** - each maintenance may have a **next date** (optional) | - Each maintenance is scheduled through one maintenance schedule - Each maintenance may use a part - A maintenance session is done for a FRED |

**Part**

|  |  |  |
| --- | --- | --- |
| Primary Key | Attributes | Relationships |
| Each part is uniquely identified by PartID | - Each part has a part **name** - Each part has a **description** of what it is used for (optional) - Each part is associated with a **category**  - Each part has a **cost** - Each part has a **stock** **amount** (managed by the supplier) | - Each part is provided by one or many suppliers through the ‘Part\_Supplier’ relationship - Each part may be used in one or many Maintenance tasks |

**Supplier**

|  |  |  |
| --- | --- | --- |
| Primary Key | Attributes | Relationships |
| Each supplier is uniquely identified by SupplierID | - Each supplier has a company **name** - Each supplier has a **contact** **person** - Each supplier has an **email** - Each supplier has a **phone** - Each supplier has an **address** | - Each supplier provides one or many parts through the Part\_Supplier relationship |

**Maintenance Schedule**

|  |  |  |  |
| --- | --- | --- | --- |
| Primary Key | Foreign Key | Attributes | Relationships |
| Each maintenance schedule is identified by MaintainerID and MaintenanceID (PK/FK reference) | | - Each maintenance schedule shows **requirements** - Each maintenance schedule has a **maintenance** **period** - Each maintenance schedule shows the **estimated** **duration** of a maintenance session | - Each maintenance schedule is managed and overlooked by one or more maintainers - Each maintenance schedule is connected to a maintenance session |

**Maintainer**

|  |  |  |
| --- | --- | --- |
| Primary Key | Attributes | Relationships |
| Each maintainer is uniquely identified by MaintainerID | - Each maintainer has a company name - Each maintainer has a contact person - Each maintainer has an email - Each maintainer has a phone - Each maintainer has an address - Each maintainer has a contract start date - each maintainer may have a contract end date (optional) | - Each maintainer may have one or many application sessions - Each maintainer is responsible to overlook one or many maintenance schedules |

**Application Session**

|  |  |  |  |
| --- | --- | --- | --- |
| Primary Key | Foreign Key | Attributes | Relationships |
| Each application session is uniquely identified by **ApplicationSessionID** | An application session is associated with a maintainer who accesses the app (**MaintainerID**) | - Each application session has a usage **description** - Each application session has a **login time** - Each application session has a **logout time** (optional) | - Each application session is controlled with one maintainer |

**Zone**

|  |  |  |  |
| --- | --- | --- | --- |
| Primary Key | Foreign Key | Attributes | Relationships |
| Each zone is uniquely identified by **ZoneID** | - Each zone is associated with a contract who assigned to it (**ContractID**) - Each zone is associated with a FRED who is working in it (**FredID**) | - Each zone has a zone **name** - Each zone has an **environment** **type** - Each zone has an **area** size - Each zone has a **boundary** size its limited to - Each zone is within a **country** | - Each zone is associated with one **FRED** - One to many zones are associated with one **contract** |

**Data**

|  |  |  |  |
| --- | --- | --- | --- |
| Primary Key | Foreign Key | Attributes | Relationships |
| Each data set is uniquely identified by **DataID** | - Each data set is created by a FRED (**FredID**) - Each data set is associated through to a contract (**ContractID**) | - Each data set has a **timestamp** - Each data set has a set ‘**Latitude’** - Each data set has a set ‘**Longitude’** - Each data set has a set ‘**Altitude’** - Each data set has a set ‘**Temperature’** - Each data set has a set ‘**Humidity’** - Each data set has a set ‘**LightMeasurement’** - Each data set has a set ‘**Organic** **Data’** | - One or many sets of data is associated with one **contract** - One or many sets of data is created by a **FRED**  - Sets are accessed/viewed by ‘**Platinum’** - Sets are owned by ‘**SuperPlatinum’** |

**Contract**

|  |  |  |  |
| --- | --- | --- | --- |
| Primary Key | Foreign Key | Attributes | Relationships |
| Each contract is uniquely identified by **ContractID** | - Each contract is organized by a **Salesperson** (Salesperson : EmployeeID)  - Each contract is organized by an **Admin** (Admin : EmployeeID) | - Each contract has a **start date** - Each contract has an **end date** - Each contract has a **payment period** - Each contract has a **discount percent** - Each contract has a **total value** - Each contract may or may not be **exclusive** | - Each contract may have one or more **invoices**  - Each contract is assigned with one **admin**  - Each contract is assigned with one **salesperson** - Each contract may be assigned with one or more **zones** - Each contract is assigned with one to many **data** sets - Each contract is assigned with one **subscriber** - Each contract is assigned with one **subscription** |

**Invoice**

|  |  |  |  |
| --- | --- | --- | --- |
| Primary Key | Foreign Key | Attributes | Relationships |
| Each invoice is uniquely identified by **InvoiceID** | Each invoice is assigned with a contract (**ContractID**) | - Each invoice has an **issue date** - Each invoice has a **due date** - Each invoice has a **subtotal amount** - Each invoice has a **tax amount** - Each invoice has a **total amount** (given from subtotal and tax) - Each invoice has a **status** | - One or more invoices are assigned to a **contract** |

**Subscription**

|  |  |  |  |
| --- | --- | --- | --- |
| Primary Key | Foreign Key | Attributes | Relationships |
| Each subscription is uniquely identified by **SubscriptionID** | Each subscription may be assigned to a Contract (**ContractID**) – optional | - Each subscription has a **name** - Each subscription may have a **description** (optional)  - Each subscription has **MaxFREDs** (optional)  - Each subscription has **MaxZone** (optional)  - Each subscription has a (optional) - Each subscription has a **base** **price** - Each subscription has a **discount** **percent** | - A subscription may be ‘standard’ - A subscription may be a ‘gold’ - A subscription may be a ‘platinum’ - A subscription may be a ‘super platinum’ - A subscription is assigned to a **subscriber** - A subscription is assigned to a **contract** |

**Subscriber**

|  |  |  |  |
| --- | --- | --- | --- |
| Primary Key | Foreign Key | Attributes | Relationships |
| A subscriber is uniquely identified by **SubscriberID** | - A subscriber is associated with a subscription (**SubscriptionID**)  - A subscriber is associated with a contract (**ContractID**) | - A subscriber has a **company** **name** - A subscriber has a **subscriber** **type** (private company etc.) - A subscriber has a **contact** **person** - A subscriber has an **email** - A subscriber has a **phone number** - A subscriber has an **address** - A subscriber has a **subscription type** | - Each subscriber is associated with one **contract** - One subscriber is associated with one **subscription** |

**Standard**

|  |  |  |
| --- | --- | --- |
| Primary Key | Foreign Key | Relationships |
| Each standard subscription is identified by Subscription (**SubscriptionID**) (PK/FK reference) | | - Standard is a **subscription** type |

**Gold**

|  |  |  |
| --- | --- | --- |
| Primary Key | Foreign Key | Relationships |
| Each gold subscription is identified by Subscription (**SubscriptionID**) (PK/FK reference) | | - Gold is a **subscription** type |

**Platinum**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Primary Key | Foreign Key | Additional Foreign Keys | Attribute | Relationships |
| Each platinum subscription is identified by Subscription (SubscriptionID) (PK/FK reference) | | - Each platinum subscription can view Data (DataID)  - Each platinum subscription has its price negotiated by an Admin (Admin : EmployeeID) | - The platinum subscription **CanViewData** | - Platinum is a **subscription** type |

**Super** **Platinum**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Primary Key | Foreign Key | Additional Foreign Keys | Attribute | Relationships |
| Each super platinum subscription is identified by PlatinumSubscription (Platinum: **SubscriptionID**) (PK/FK reference) | | - Each super platinum subscription can own Data (**DataID**)  - Each super platinum subscription has its price negotiated by an Admin (Admin : **EmployeeID**) | - The super platinum subscription **CanOwnData** | - Super platinum is a **subscription** type |

**Employee**

|  |  |  |
| --- | --- | --- |
| Primary Key | Attributes | Relationships |
| Each employee is uniquely identified by **EmployeeID** | - Each employee has a **name** - Each employee has an **email** - Each employee has a **phone** **number** (optional) - Each employee has a **position** - Each employee has a **salary** | - An employee may be an **engineer** - An employee may be a **salesperson** - An employee may be an **admin** |

**Admin**

|  |  |  |  |
| --- | --- | --- | --- |
| Primary Key | Foreign Key | Attributes | Relationships |
| Each admin is uniquely identified by the reference of EmployeeID (PK/FK reference) | | - Each admin has an **authorization level** - Each admin may be able to **change prices** (bit) | - Each admin is an **employee** - Each admin is associated with a contract - Each admin can control the price of a ‘Platinum’ subscription - Each admin can control the price of a ‘SuperPlatinum’ subscription |

**Salesperson**

|  |  |  |  |
| --- | --- | --- | --- |
| Primary Key | Foreign Key | Attributes | Relationships |
| Each salesperson is uniquely identified by the reference of EmployeeID (PK/FK reference) | | - Each salesperson has a **commission rate** - Each salesperson has a **max discount percent** - Each salesperson may have a **target sales** (optional) | - Each salesperson is an **employee -** Each salesperson is associated with a contract |

**Engineer**

|  |  |  |  |
| --- | --- | --- | --- |
| Primary Key | Foreign Key | Attributes | Relationships |
| Each engineer is uniquely identified by the reference of EmployeeID (PK/FK reference) | | - Each engineer may **specialize in** something (optional) - Each engineer may have a **certification** (optional) | - An engineer is an employee  - Each engineer can build one or many FRED drones |

**Director**

|  |  |  |
| --- | --- | --- |
| Primary Key | Attributes | Relationships |
| Each director is uniquely identified by DirectorID | - Each director has a **name** - Each director has an **email** - Each director has a **phone number** - Each director has a **position** | - Each director can hire one or many employees |

**Step 4: Populated sentences with Examples:**

**FRED**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| Each FRED is uniquely identified by **FredID** | 701 |
| Each FRED has a **serial** **number** | 1460392842 |
| Each FRED has a **date commissioned** | 12/04/25 |
| Each FRED has a **status** | Deployed |
| Each FRED has a **last communication** date | 25/07/25 |
| Each FRED has a **battery level** | 74 |

**Service**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| Each Service is uniquely identified by **ServiceID** | 146 |
| A service is associated with one FRED **(FredID)** | 701 |
| Each service has an optional **description** | Tree Trimming |
| Each service has a **category** | Part Service |

**Maintenance**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| Each maintenance is uniquely identified by **MaintenanceID** | 363 |
| Each maintenance session may use a Part (PartID) | 135 |
| Each maintenance has a **date** | 12/03/2025 |
| Each maintenance has a **description** | Fixing the front left propeller |
| Each maintenance might have **parts used** (optional) | Propeller |
| Each maintenance has a **cost** applied | 150.00 |
| Each maintenance may have a **next** **date** (optional) | 25/05/2025 (Checkup) |

**Part**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| Each part is uniquely identified by **PartID** | 742 |
| Each part has a part **name** | Propeller |
| Each part has a **description** of what it is used for | Front propeller to create lift and thrust |
| Each part is associated with a **category** | Hardware part |
| Each part has a **cost** | 150.00 |
| Each part has a **stock amount** (managed by the supplier) | 25 |

**Supplier**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| Each supplier is uniquely identified by **SupplierID** | 101 |
| Each supplier has a company **name** | Tech Parts Inc |
| Each supplier has a **contact** **person** | John Doe |
| Each supplier has an **email** | [contact@techparts.co.nz](mailto:contact@techparts.co.nz) |
| Each supplier has a **phone** | 022 583 2953 |
| Each supplier has an **address** | 123 Address St, Industry Town, CA 90120 |

**Maintenance Schedule**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| Each maintenance schedule is identified by **MaintainerID** (PK/FK reference) | 267 |
| Each maintenance schedule is identified by **MaintenanceID** (PK/FK reference) | 138 |
| Each maintenance schedule shows **requirements** | Replace camera as it is damaged from rock that flicked up and broke the lens |
| Each maintenance schedule has a **maintenance period** | 100 days |
| Each maintenance schedule shows the **estimated duration** of a maintenance session | 30 minutes |

**Maintainer**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| Each maintainer is uniquely identified by **MaintainerID** | 342 |
| Each maintainer has a company **name** | ServiceCore Maintenance |
| Each maintainer has a **contact** **person** | Jane Doe |
| Each maintainer has an **email** | [service@servicecore.co.nz](mailto:service@servicecore.co.nz) |
| Each maintainer has a **phone** | 021 204 2630 |
| Each maintainer has an **address** | 342 Address St, Techtown |
| Each maintainer has a contract **start date** | 14/02/25 |
| each maintainer may have a contract **end date** (optional) | 15/02/25 |

**Application Session**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| Each application session is uniquely identified by **ApplicationSessionID** | 856 |
| An application session is associated with a maintainer who accesses the app (**MaintainerID**) | 332 |
| Each application session has a **usage** **description** | Accessing records |
| Each application session has a **login** time | 10/03/2025 : 12:45pm |
| Each application session has a **logout** time | 10/03/2025: 1:15pm |

**Zone**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| Each zone is uniquely identified by **ZoneID** | 243 |
| Each zone is associated with a contract who assigned to it (**ContractID**) | 233 |
| Each zone is associated with a FRED who is working in it (**FredID**) | 555 |
| Each zone has a zone **name** | Richmond |
| Each zone has an **environment** type | Town center, flat lands |
| Each zone has an **area** size | 640 acres |
| Each zone has a **boundary** **size** its limited to | 300km |
| Each zone is within a **country** | New Zealand |

**Data**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| Each data set is uniquely identified by **DataID** | 101 |
| Each data set is created by a FRED (**FredID**) | 232 |
| Each data set is associated through a contract (**ContractID**) | 112 |
| Each data set has a **timestamp** | 20/05/2025 |
| Each data set has a set ‘**Latitude’** | 34.05334 |
| Each data set has a set ‘**Longitude’** | 118.25345 |
| Each data set has a set ‘**Altitude’** | 253.3 |
| Each data set has a set ‘**Temperature’** | 23.5 C |
| Each data set has a set ‘**Humidity’** | 50.5 |
| Each data set has a set ‘**LightMeasurement’** | 845.70 |
| Each data set has a set ‘**Organic Data’** | carbon:12ppm;nitrogen:3ppm |

**Contract**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| Each contract is uniquely identified by **ContractID** | 567 |
| Each contract is organized by a Salesperson (Salesperson : EmployeeID | 332 |
| Each contract is organized by an Admin  (Admin: EmployeeID) | 253 |
| Each contract has a **start date** | 15/05/2025 |
| Each contract has an **end date** | 10/07/2025 |
| Each contract has a **payment period** | Monthly |
| Each contract has a **discount percent** | 10% |
| Each contract has **a total value** | $5000 |
| Each contract may or may not be exclusive (bit) | 1 |

**Invoice**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| Each invoice is uniquely identified by **InvoiceID** | 768 |
| Each invoice is assigned with a contract (**ContractID**) | 252 |
| Each invoice has an **issue date** | 15/03/2025 |
| Each invoice has a **due date** | 20/04/2025 |
| Each invoice has a **subtotal amount** | $1530 |
| Each invoice has a **tax amount** | $229.5 (subtotal x 0.15 |
| Each invoice has a **total amount** (given from subtotal and tax) | 1759.50 |
| Each invoice has a **status** | Paid |

**Subscription**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| Each subscription is uniquely identified by **SubscriptionID** | 936 |
| Each subscription may be assigned to a Contract (**ContractID**) (optional) | 332 |
| Each subscription has a **name** | Gold |
| Each subscription may have a **description** (optional) | Provides decent amount of features for a client |
| Each subscription has MaxFREDs (optional) | 3 |
| Each subscription has MaxZone (optional) | 2 |
| Each subscription has a MaxZoneArea (optional) | 500/sqm |
| Each subscription has a **base price** | 500.00 |
| Each subscription has a **discount percent** | 25.05% |

**Subscriber**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| A subscriber is uniquely identified by **SubscriberID** | 274 |
| A subscriber is associated with a subscription (**SubscriptionID**) | 555 |
| A subscriber is associated with a contract (**ContractID**) | 354 |
| A subscriber has a **company** **name** | TechPro Corporation |
| A subscriber has a **subscriber** **type** (private company etc.) | Private Tech Company |
| A subscriber has a **contact** **person** | Jane Doe |
| A subscriber has an **email** | [techpro@email.com](mailto:techpro@email.com) |
| A subscriber has a **phone number** (optional) | 021 204 2942 |
| A subscriber has an **address** | 30 Address St, DataCity, AU 21903 |
| A subscriber has a **subscription type** | Gold |

**Standard**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| Each standard subscription is identified by Subscription (**SubscriptionID**) (PK/FK reference) | 257 |

**Gold**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| Each gold subscription is identified by Subscription (**SubscriptionID**) (PK/FK reference) | 969 |

**Platinum**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| Each platinum subscription is identified by Subscription (**SubscriptionID**) (PK/FK reference) | 978 |
| Each platinum subscription can view Data in DataID (FK) | 134 |
| Each platinum subscription has cost decided by Admin in Admin : EmployeeID (FK) | 331 |
| Each platinum subscription CanViewData | 1 |

**Super Platinum**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| Each super platinum subscription is identified by Platinum Subscription (Platinum : **SubscriptionID**) (PK/FK reference) | 573 |
| Each super platinum subscription can own data in DataID (FK) | 238 |
| Each super platinum subscription has cost decided by Admin in Admin: EmployeeID | 301 |
| Each super platinum subscription CanOwnData | 1 |

**Employee**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| Each employee is uniquely identified by **EmployeeID** | 673 |
| Each employee has a **name** | Jane Doe |
| Each employee has an **email** | [Employee101@email.com](mailto:Employee101@email.com) |
| Each employee has a **phone number** (optional) | 021 203 2639 |
| Each employee has a **position** | Engineer |
| Each employee has a **salary (**optional**)** | 30.00 |

**Admin**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| Each admin is uniquely identified by the reference of **EmployeeID** (PK/FK reference) | 475 |
| Each admin has an **authorization level** | Highest |
| Each admin may be able to **change prices** (bit) | 1 (yes) |

**Salesperson**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| Each salesperson is uniquely identified by the reference of **EmployeeID** (PK/FK reference) | 214 |
| Each salesperson has a **commission** **rate** | $200.00 |
| Each salesperson has a **max** **discount percent** | 15% |
| Each salesperson may have a **target sales (**optional**)** | 300 |

**Engineer**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| Each engineer is uniquely identified by the reference of **EmployeeID** (PK/FK reference) | 142 |
| Each engineer may **specialize** **in** something (optional) | Mechanics |
| Each engineer may have a **certification** (optional) | Bachelor’s Degree |

**Director**

|  |  |
| --- | --- |
| **Attribute** | **Data Example** |
| Each director is uniquely identified by **DirectorID** | 174 |
| Each director has a **name** | John Doe |
| Each director has an **email** | [Director142@email.com](mailto:Director142@email.com) |
| Each director has a **phone** **number** (optional) | 022 048 2737 |
| Each director has a **position** | Directing Manager |

**Step 5: Ternary and N-ary Relationship Sentences:**

**FRED:**

1: One FRED (FredID) may conduct one or more services (ServiceID)

2: One Service (ServiceID) must be assigned to one FRED (FredID)

3: One FRED is designed/created by one Engineer (Engineer : EmployeeID)

4: One Engineer (EngineerID) can design/create one or many FRED drones (FredID)

5: One FRED (FredID) is associated with one Zone (ZoneID)

6: One Zone (ZoneID) is associated with one Fred (FredID)

7: One FRED (FredID) can generate one or more sets of Data (DataID)

8: One data (DataID) set is created by one FRED (FredID)

9: One FRED (FredID) undergoes one Maintenance (MaintenanceID) session

10: One Maintenance (MaintenanceID) session is conducted for one FRED (FredID)

**Maintenance:**

11: One Maintenance (MaintenanceID) session is assigned to one-to-many MaintenanceSchedule(s) (MaintainerID/MaintenanceID)

12: One MaintenanceSchedule (MaintainerID/MaintenanceID) assigns to one Maintenance (MaintenanceID) session

12: One Maintenance (MaintenanceID) session may use one or more Part(s) (PartID)

13: One Part (PartID) may be used in one Maintenance (MaintenanceID) session

**Part**:

14: One Part (PartID) may be supplied by one or more Supplier(s) (SupplierID)

15: One Supplier (SupplierID) may supply one or more Parts (PartID)

**MaintenanceSchedule:**

16: One MaintenanceSchedule (MaintainerID/MaintenanceID) may be controlled by one Maintainer (MaintainerID)

17: One Maintainer (MaintainerID) may control one or more MaintenanceSchedule(s) (MaintainerID/MaintenanceID)

**Maintainer**:

18: One Maintainer (MaintainerID) will create one ApplicationSession (ApplicationSessionID)

19: One ApplicationSession (ApplicationSessionID) will be created by one Maintainer (MaintainerID)

**Zone**:

20: One Zone (ZoneID) is associated with one Contract (ContractID)

21: One Contract (ContractID) is associated with one or many Zone(s) (ZoneID)

**Data**:

22: One Data (DataID) set is assigned with one Contract (ContractID)

23: One Contract (ContractID) is assigned with one or many Data (DataID) sets

24: One Data (DataID) set is viewed by one Platinum (Platinum : SubscriptionID) subscription

25: One Platinum (Platinum : SubscriptionID) subscription can view one or more sets of Data (DataID)

26: One Data (DataID) set is owned by one SuperPlatinum (SuperPlatinum : SubscriptionID) subscription

27: One SuperPlatinum (SuperPlatinum : SubscriptionID) subscription can own one or more sets of Data (DataID)

**Contract**:

28: One Contract (ContractID) is assigned to one Salesperson (Salesperson : EmployeeID)

29: One Salesperson (Salesperson : EmployeeID) is assigned to one Contract (ContractID)

30: One Contract (ContractID) is assigned to one Admin (Admin : EmployeeID)

31: One Admin (Admin : EmployeeID) is assigned to one Contract (ContractID)

32: One Contract (ContractID) generates one or more Invoice(s) (InvoiceID)

33: One Invoice (InvoiceID) is generated by one Contract (ContractID)

34: One Contract (ContractID) is associated with one Subscriber (SubscriberID)

35: One Subscriber (SubscriberID) is associated with One Contract (ContractID)

36: One Contract (ContractID) is associated with one Subscription (SubscriptionID)

37: One Subscription (SubscriptionID) is associated with one Contract (ContractID)

**Subscription**:

38: One Subscription (SubscriptionID) has one subtype called Standard (Standard : SubscriptionID)

39: One Standard (Standard: SubscriptionID) subscription is a subtype of Subscription (SubscriptionID)

40: One Subscription (SubscriptionID) has one subtype called Gold (Gold : SubscriptionID)

41: One Gold (Gold : SubscriptionID) subscription is a subtype of Subscription (SubscriptionID)

42: One Subscription (SubscriptionID) has one type called Platinum (Platinum : SubscriptionID)

43: One Platinum (Platinum : SubscriptionID) subscription is a subtype of Subscription (SubscriptionID)

44: One Subscription (SubscriptionID) is associated with one Subscriber (SubscriberID)

45: One Subscriber is associated with one Subscription (SubscriptionID)

**Employee**:

46: One Employee (EmployeeID) has one subtype called Admin (Admin : EmployeeID)

47: One Admin (Admin : EmployeeID) is a subtype of Employee (EmployeeID)

48: One Employee (EmployeeID) has one subtype called Salesperson (Salesperson : EmployeeID)

49: One Salesperson (Salesperson : EmployeeID) is a subtype of Employee (EmployeeID)

50: One Employee (EmployeeID) has one subtype called Engineer (Engineer : EmployeeID)

51: One Engineer (EngineerID) is a subtype of Employee (EmployeeID)

**Director**:

52: One Director (DirectorID) hires one or more Admin (Admin : EmployeeID)

53: One Admin (Admin : EmployeeID) can be hired by one Director (DirectorID)

**Admin**:

54: One Admin can set the price of one Platinum (Platinum : SubscriptionID) subscription

55: One Platinum (Platinum : SubscriptionID) has a price set by one Admin (Admin : EmployeeID)

56: One Admin can set the price of one SuperPlatinum (SuperPlatinum : SubscriptionID) subscription

57: One SuperPlatinum (SuperPlatinum : SubscriptionID) has a price set by one Admin (Admin : EmployeeID)

**Platinum**:

58: One Platinum (Platinum : SubscriptionID) has one subtype called SuperPlatinum (SuperPlatinum : SubscriptionID)

59: One SuperPlatinum (SuperPlatinum : SubscriptionID) is a subtype of Platinum (Platinum : SubscriptionID)

# NaLER Rationale:

This analysis written above demonstrates a structured approach to evaluate the information needs within an organization. The analysis showcases the use of data modelling by accurately identifying key entities (i.e., contracts, FREDs, subscriptions, employees etc.) and their corresponding relationships, which is important to note when designing a proficient data model.

Concepts of Data Modelling include:

1. I have identified what the business requires from the system, such as tracking different subscription tiers, the duration of a contract etc. This information shows the scope and purpose of the data model and how everything is used for the database
2. By defining how different sets of data interact with each other, it showcases the structure needed for the Entity-Relationship Modelling. The foundation I’ve created shows that each table is designed as per the requirements, the attributes have been assigned appropriately, and the relationships are realistic to the connections needed for the system.
3. The analysis shows the consideration of dependencies, which is necessary for a relational schema. For example, ‘One FRED unit can be associated with many different contracts overtime’, which supports referential integrity in the database

Concepts of Data Management Principles include:

1. I have ensured that all the entities and attributes are identified and correct, additionally aligning with the defined logic for the system – showing the representation of relationships improves how reliable the data will be for the system.
2. The layout of the NaLER shows how each element is described with purpose and use-case clarity (through the use of data samples), making it easier to visualize how the system will be used for end-users

The NaLER analysis has proposed separate entities for each subscription tier, however since all tiers share the same attributes and ONLY have differences by the values (maximum number of zones allowed, for example) it would be more beneficial to stick with the one table, ‘Subscription’, having the attributes that are originally defined in the subtype classes.

Overall, by defining the entities and their attributes through sentences, rather than visually presenting it, provides a more substantial layout to figure out how exactly the system works and how each table relationship binds the system together. Additionally, showcasing data samples makes it easier to prove the usability and purpose of each table and their columns, showing how each table is beneficial to define a certain aspect of the system.  
Providing sentences to explain the relationships also makes it simpler to process how each table can communicate and connect with other tables, showing how unified the tables in the system is to provide efficient service for clients.  
For those who need a written explanation on how to create the ERD, utilizing the NaLER analysis makes it notably effortless to design – proving how straightforward the system can be seen through written details.

# Assignment 2 Part 2 – Physical Design:

# References:

**McFadyen, R. (n.d.). *7.3.1: Atomic attributes*. Relational Databases and Microsoft Access 365.** [**https://ecampusontario.pressbooks.pub/relationaldatabasesandmicrosoftaccess365/chapter/\_\_unknown\_\_-59/#:~:text=A%20simple%2C%20or%20atomic%2C%20attribute,decomposed%20into%20other%20smaller%20components**](https://ecampusontario.pressbooks.pub/relationaldatabasesandmicrosoftaccess365/chapter/__unknown__-59/#:~:text=A%20simple%2C%20or%20atomic%2C%20attribute,decomposed%20into%20other%20smaller%20components)**.**

Brumm, B. (2023, July 16). *Normalization in SQL DBMS: 1NF, 2NF, 3NF, and BCNF examples*. PopSQL. <https://popsql.com/blog/normalization-in-sql>

Upadhyay, M. (2025, May 14). *First Normal Form (1NF)*. GeeksforGeeks. <https://www.geeksforgeeks.org/first-normal-form-1nf/>

Upadhyay, M (2025, May 13). *Boyce-Codd Normal Form (BCNF).* GeeksforGeeks***.***<https://www.geeksforgeeks.org/boyce-codd-normal-form-bcnf/>