ASSIGNMENT 2

Author: George Price

Student ID: 12129164

Date: Sunday, 23 May 2021

Unit Coordinator: Elizabeth Tansley

Unit Code: COIT11237 (HT1, 2021)

Unit: Database Design & Implementation

Entity Relationship Model

For this assignment's ERD, I used Visio to produce the model flow:

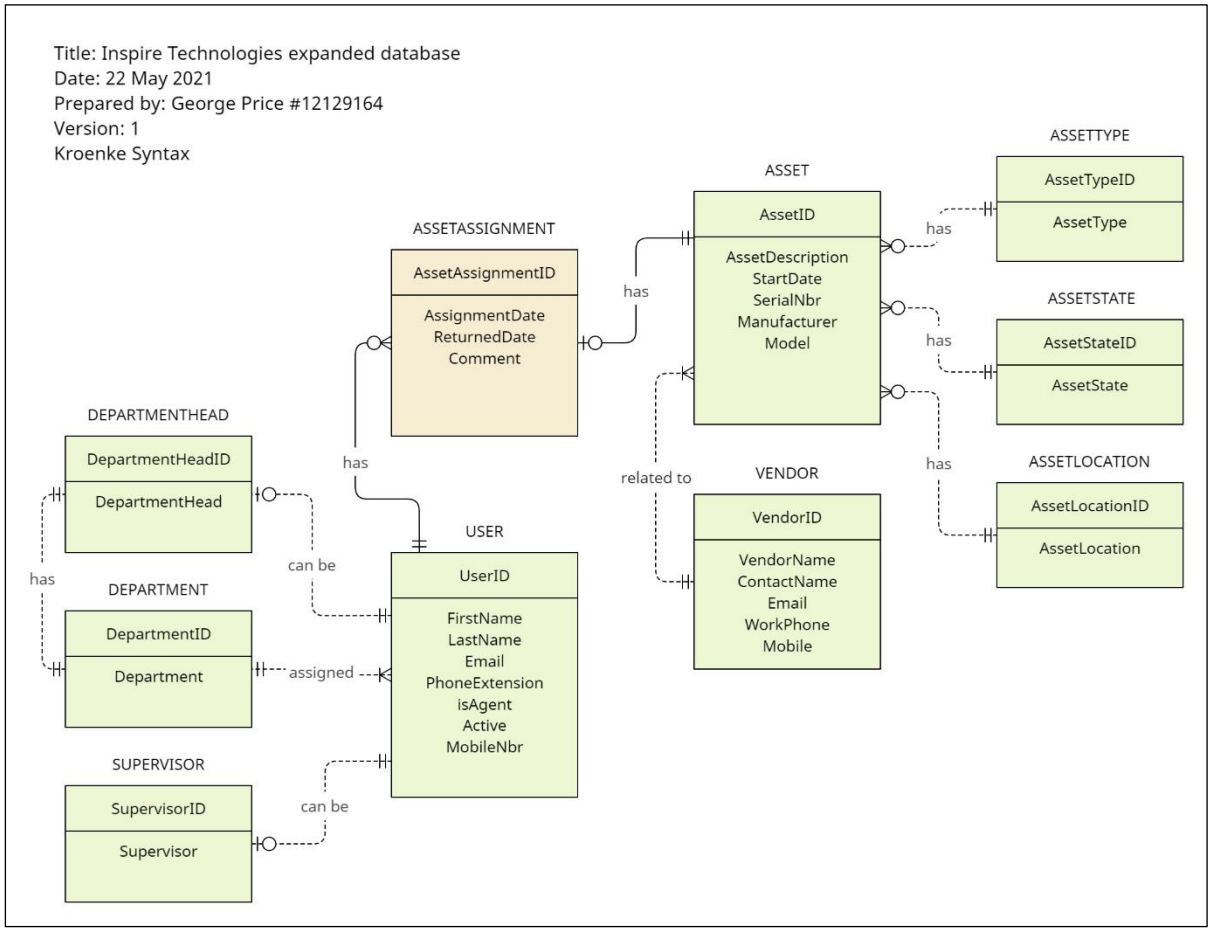

Figure 1. Entity Relationship model as per part 1.

Assumptions

For this assignment's Entity Relationship model (Figure 1), some assumptions were made concerning entity relationships, attributes, and cardinality:

1. **Asset Types.** Here it is assumed that Asset Types should be made their own entity table, so Inspire Technologies staff can update OR add new Asset Types easily. It is feasible to assume that new technologies may be used, or Asset descriptions changed. Every Asset must have an Asset Type, but many Asset Types exist.
2. **Asset States.** As with the above assumptions, it is likely that in the future, Asset State options may evolve. Additionally, an Asset State is expected to change over time, so making them their own entity provides for this. Every Asset must have an Asset State, and many Asset States can exist.
3. **Asset Locations.** It can be assumed that Inspire Technologies may open new branches, or move current ones, and in this way the Asset Locations would need to be updated, and new locations added. The Asset Locations are placed on their own entity table to easily allow for this. An Asset without a location does not make sense, so every Asset must have an Asset Location, but only one.
4. **Vendors.** It is proposed that Vendors should be their own entity, normalised and ID independent. This way each Asset that exists can have a VendorID for warranty and other purposes, that can easily be ripple updated. Each Asset has to have a Vendor, but one or many Vendors can be associated with one or more Assets. It is also assumed a new Vendor cannot be added unless an Asset is entered along with it.
5. **Asset Assignment**. It is assumed that Asset Assignment should be its own entity but be ID dependent on a UserID and an AssetID as composite identifiers. This entity could be weak, so if an Asset Assignment is deleted, neither the User nor the Asset would be. So, an Asset Assignment must have a User and an Asset to justify its existence, but an Asset or User exists without an Asset Assignment, also a User may have many Asset Assignments (a query would be needed to view this, as there is currently no implementation to view multiple Asset Assignments from the User's row of the User table).
6. **Supervisor.** As a User *can be* a Supervisor, and a Supervisor will supervise another user, it is assumed that it is best to make Supervisor its own entity that must be a User. This way multiple Users can have the same SupervisorID, be a Supervisor itself, or neither if that is ever required.
7. **Department Head.** Because each Department must have a Department Head that is a User, and every User be in a Department, it is assumed that multiple users may be in the same Department, including the Department Head itself. So, a separate entity for Department Head is implemented that is dependent on having a DepartmentID and UserID for each record to exist. Department Heads can change, and so can Departments, this makes it easier to update these changes with the existing records.
8. **User.** Because a User can be a Supervisor, that supervises other Users, it is assumed that one or many Users can supervise one or many *other* Users. It is also assumed that one user may have many Asset Assignments at a time, so no indexed unique relationship was made in the User attribute.
9. **Asset.** It is assumed an Asset cannot be in more than one Asset Assignment at any given time, or may not be assigned at all.

Logical Design

Here is the logical design for the Entity Relationship Diagram (Figure 1):

User (UserID, FirstName, LastName, Email, PhoneExtension, isAgent, Active, MobileNbr, *DepartmentID*, *SupervisorID*)

foreign key (DepartmentID) references Department.DepartmentID

foreign key (SupervisorID) references Supervisor.SupervisorID

Department (DepartmentID, Department, *DepartmentHeadID*)

foreign key (DepartmentHeadID) references DepartmentHead.DepartmentHeadID

DepartmentHead (DepartmentHeadID, DepartmentHead, *UserID*)

foreign key (UserID) references User.UserID

Supervisor (SupervisorID, Supervisor, *UserID*)

foreign key (UserID) references User.UserID

Asset (AssetID, AssetDescription, StartDate, SerialNbr, Manufacturer, Model, *AssetTypeID*, *AssetStateID*, *AssetLocationID*, *AssetAssignmentID*, *VendorID*)

foreign key (AssetTypeID) references AssetType.AssetTypeID

foreign key (AssetStateID) references AssetState.AssetStateID

foreign key (AssetLocationID) references AssetLocation.AssetLocationID

foreign key (AssetAssignmentID) references AssetAssignment.AssetAssignmentID

foreign key (VendorID) references Vendor.VendorID

AssetAssignment (AssetAssignmentID, AssignmentDate, ReturnedDate, Comment, *AssetID*, *UserID*)

foreign key (AssetID) references Asset.AssetID

foreign key (UserID) references User.UserID

AssetType (AssetTypeID, AssetType)

AssetState (AssetStateID, AssetState)

AssetLocation (AssetLocationID, AssetLocation)

VendorID (VendorID, VendorName, ContactName, Email, WorkPhone, Mobile)

Physical Design

Here is the physical design for the Assignment table:

Column name	Type/length	Identifier	Required	Default value	Data constraints
AssetAssignmentID	Autonumber	Yes	Yes	n/a	n/a
AssignmentDate	Date/time	No	Yes	Current date/time	Must not be in the future
ReturnedDate	Date/time	No	No	No default	Should only be filled when asset is returned
Comment	Short Text	No	No	No default	Max of 255 characters
AssetID	Long Integer	Yes	Yes	No default	Must exist in the Asset table
UserID	Long Integer	Yes	Yes	No default	Must exist in the User table