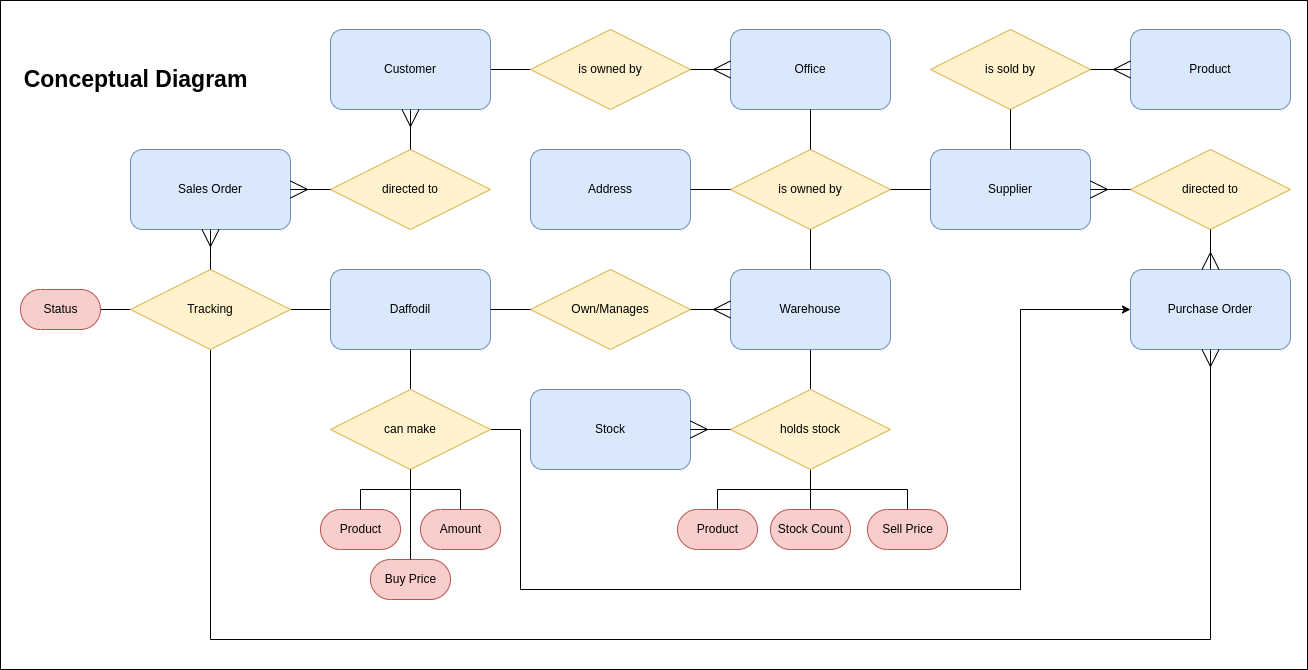
**The Conceptual Diagram**

When designing the conceptual diagram. I had a quick read through the requirements we devised to first create the all entities starting with 'Daffodil'. I then went through the requirement one by one to create junction (connection) for each requirement next to their entities. The entities that ended up with identical or near identical names were placed close to each other and then connected to the relevant entity with only one junction table. I then attempted to merge all of these series of connections together and where it is tough to merge a separate junction of an identical name were created (see "directed to"). Finally i added the attributes making sure no repeated were made and where an attribute is required in two junctions they were loosely accessible by both/all.

The conceptual diagram was designed to only serve as guidance for the Entity Relationship Diagram.



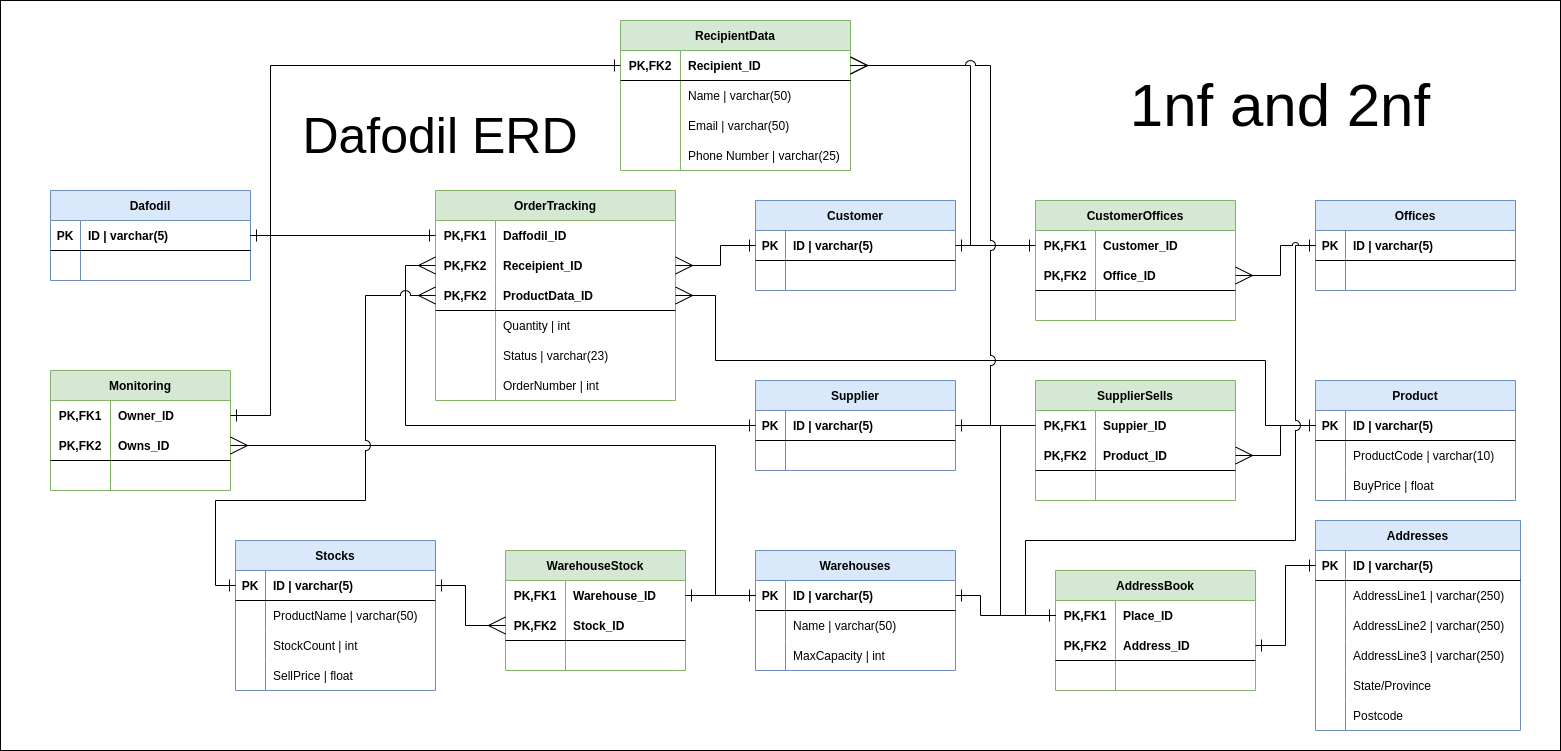
**The Entity Relationship Diagram**

I started off the ERD by modelling the entities and their PKs and adding plausible attributes (as none were presented) such as name Address, etc. I then made junctions identical to the conceptual diagram (with name changes where appropriate) making sure the duplicates from the conceptual diagram were merged here.

I then gave all entities an ID row as private keys with 5 characters. and then gave the relations a template of 1, 2 or 3 PKs, FKs numbered (depending on the amount of connectors and nature of said connections from the conceptual ) and then given a name once again based on the amount of connectors and the nature of the connectors

I then went back to the requirements and went through them individually to connect the junctions to the entities. With all junctions made i took this time to add any other possible attributes to get the table ready for 1nf normalisation.

I used a temporary excel file with sample data to visualise the data when the connections were hard to understand whilst designing some of the connectors. It also helped during normalising 1 and 2 normal form.



**How i checked 1nf (with examples)**

with aid from this video (https://youtu.be/J-drts33N8g?si=SR6GccZDZOyjlqh8) I wrote the following requirements for 1nf.

**- All rows must be unique (no duplicate rows)**

To achieve this requirement I created new junctions and sometimes to hold identical data held by two or more entities. This led to the creation of Recipient Data, Addressbook, etc. Where data can be shared in an junction that already exists, the fitting data is moved to the shared junction and removed from the entities. e.g. OrderTracking.

**- Each cell must only contain a single value (not a list)**

This was only utilised in Addresses where previously, the data was Address, it was changed to reflect the recommendations / guidelines of Royal Mail being 3 addresslines followed by state/town/province and then the postcode.

**- Each value should be non divisible (cant be split down further)**

Although previously i had used this to split Name in ReceipientData, i reverted the change as it was more likely Daffodil would save organisation names rather than specific individuals.

By the end i was satisfied that the database met all 1nf requirements with no duplicate data anywhere.

**How i checked 2nf (with examples)**

Using the same video I wrote the following requirements

**- Database must be in First Normal Form**

The database was checked to be in first normal form.

**- Non partial dependency**

All non-prime attributes should be fully functionally dependent on the candidate key

At the state the ERD was in, i only utilised this principle on OrderTracking. Where the Supplier and Customer ID were merged and combined into Recipient ID to match "ReciepientData" and then also for buy price which was not required individually in stocks and thus was moved to product.

This means it can conceptually be used with a join command when needed but is stored in a separate relation.

**- 2.5nf and 3nf**

It is plausible that the ERD meets the requirement for 2.5 and 3nf but due to my limited comprehension of them i am not currently capable of confirming it.

**References**

Kurdi, D. (2021). *UK Address Guidelines*. [online] IdealPostcodes. Available at: <https://ideal-postcodes.co.uk/guides/good-addressing-guidelines> [Accessed 8 Nov. 2023].

Learn Learn Scratch Tutorials (2020). *1st, 2nd and 3rd Normal Form (Database Normalisation)*. *YouTube*. Available at: <https://www.youtube.com/watch?v=J-drts33N8g>. [Accessed 8 Nov. 2023].

Wikipedia Contributors (2019). *Database normalization*. [online] Wikipedia. Available at: <https://en.wikipedia.org/wiki/Database_normalization>. [Accessed 28 Oct. 2023].