

DB2023_Final

by Luke Redmore

Submission date: 23-Apr-2023 11:14PM (UTC+0100)

Submission ID: 203080737

File name: 41367_Luke_Redmore_DB2023_Final_486136_1113857817.docx (1.02M)

Word count: 2326

Character count: 12224



[10]

DATABASES COURSEWORK

Northern Lights PVT

ABSTRACT

Documentation of my entity relationship diagram(s), along with documentation on my database. Along the way there are some IEEE cited explanations into some of techniques used as well as some general definitions

Luke Redmore

Database design and Web concepts 2023

Table of Contents

Introduction	1
Firstly, I wish to identify the types of users in my database.....	1
Outline of departmental operations:	2
Database plan	3
The ER diagram:	3
The staff part.....	4
The department, branch, and the bookings.	5
Adhering to normal form conventions:	7
The database!	7
Input Validation and Tables:	7
Relationship report	9
Reports and queries	10
Forms	12
Macros used!	13
References	15

Introduction

In this document I will be designing and demonstrating the use of a small dbms (database management system). in this case Microsoft access, and its applications for SMB's (small and medium businesses).

I will outline the types of users and demonstrate the plan for my relational database in the form of an entity relationship diagram. I will then from this produce an access database with a series of functionalities as outlined in the brief, such as menus, queries, and calculations from said queries.

Firstly, I wish to identify the types of users in my database.

However before diving in and doing so I think defining the types of users that will access my database is important.

First you have a Database Administrator (DBA), this will be a person or a team that define the three levels of the database, the DBAs are also responsible for the creation and management of accounts that will allow users to access the database with the minimum level of permissions needed. A DBA is also responsible for providing security to the database and they only allow the authorised users to access or modify any database systems.

The DBAs are also utterly responsible for any performance issues or security breaches to any database systems. They will monitor the recovery drives and provide any technical support to other users.

DBAs also maintain the physical and digital systems of the database should any software or hardware malfunction. [1]

Database designers, these users design the structure of the database this includes tables, indexes, the views triggers, and stored procedures with of course the various constraints of which will usually be enforced before database creation, they also control what data is to be stored and how it will then be related. [1]

It is this user levels responsibility to understand the requirements of various user groups and to produce a design that satisfies them all. [1]

Naive or parametric end user, this group does not infer any knowledge of the machinations of a database they simply access the database to perform their given roles. [1]

I decided that staff in the administration department will be Database administrators, this will allow them to make changes to other departments tables, the theory being if only one department keeps track of the database there is less likely to be error, this will be strictly for administrative managers, another responsibility of this department will be to ensure each venue is assigned the correct type of department and resources, they will also delegate external resources such as catering.

Some of my administrative staff will also be database designers, this is as I feel this department should also be considering the types of users that they have created and to produce the designs needed to satisfy them.

The head of a department will also be given DBA permissions as this will allow them to coordinate with administrative staff to ensure that branches department has the correct amount of resources represented.

General level staff such as lighting engineers or sound engineers will be classed as a casual user with regards to access level, this is as they may frequently view the database to ensure the equipment, they are packing for a venue is correct or not scheduled for maintenance.

Finally, the customers that have used this company to book and organise an event will be classed as a naïve end user, this is as they will use a view of the database to check details but will likely not have any prior knowledge on the workings of a dbms, (database management system).

I decided to break down the brief into four department types of which each branch will have one or more of each department.

These being:

Sound, light and visuals, custodial and maintenance, and administration

Outline of departmental operations:

Sound engineering: For checking acoustic viability of the location and installing and managing sound outputs and inputs.

Light and visual: Responsible

Database plan

Now before I start diving into access and lay down my first tables and forms, I think it's a good idea to have a plan.

This plan will entail the software will design the entity relationship diagram with, and of course the type of ER (Entity relationship) diagram I will use.

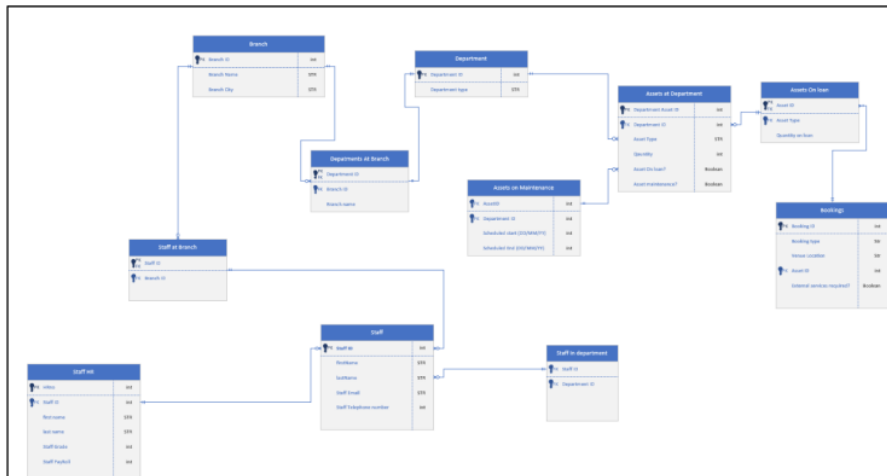
Firstly however, what even is an "entity relationship diagram"?

Well, an entity can be anything with a physical existence say a person, car, house, or employee. Or perhaps even something with a conceptual existence like a job or a company. [2]

Now any entity you create say, employee, will have various properties, such as employeeID, name, job role etc, these properties are called attributes. [2]

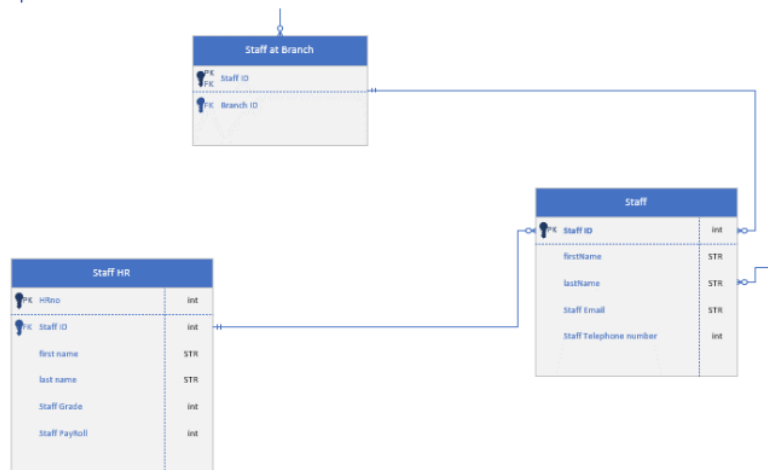
I decided upon using Crows foot notation to represent the degree of relationships between my entities.

The ER diagram:



Seen above is a screen snip of my ER diagram in crows foot notation.

The staff part.



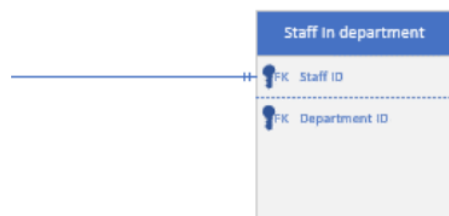
Seen above is a close-up view of my staff table(s), the degree of relationships here is as follows.

There are many staff feeding into one instance of "staff HR" here staff ID is acting as a foreign key which enforces the established link between Staff and "Staff HR".

In the staff at Branch table, you see the field "Staff ID" acting as both a primary and foreign key, as such this table is an example of a weak entity as it is dependent on the existence of Staff, please also note the presence of the foreign key "staff Department" under HR this is simply there as it is an important piece of admin for a HR form.

"A weak entity {Staff at branch} is dependent on a Strong entity {Staff} usually a weak entity will use a partial discriminator key, but the concept of dependence here and not being able to be uniquely identified still stand". [3]

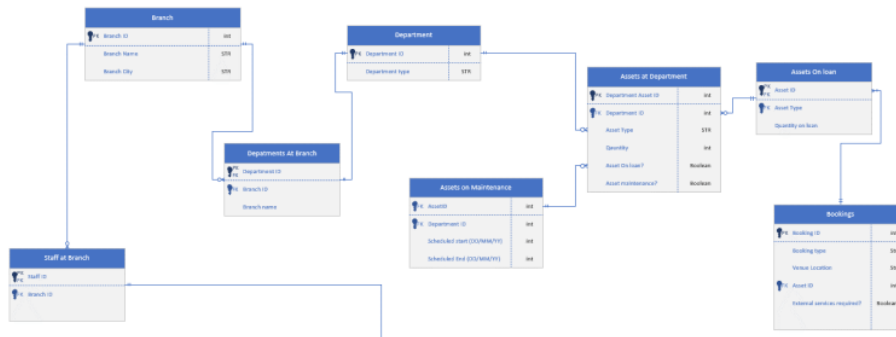
"For notes sake a partial key would be a set of attributes that ultimately form a unique identity in a Dbms" [4]



Note "Staff in department" on the left, this is another example of a Many to one relationship.

I/e many staff are in one department at a given time.

The department, branch, and the bookings.



Seen above here are my Branch and department entities, it all starts from the aforementioned “Staff at Branch” entity this forms a many to one relation with “Branch” in that order, as many Staff will work in each branch, you will note Branch ID is used as the foreign key which enforces this instance.

Logically Branch is then related to “Departments at Branch” in a one-to-many relationship as I feel each Branch may have many departments, to enforce this instance Branch ID is again being employed as a foreign key. The “departments at Branch” entity now forms a one-to-one relationship with “department” as this is merely a “bridging table” I will explain later. Here department ID is acting as a foreign and primary key to ensure data integrity and enforcing this relationship.

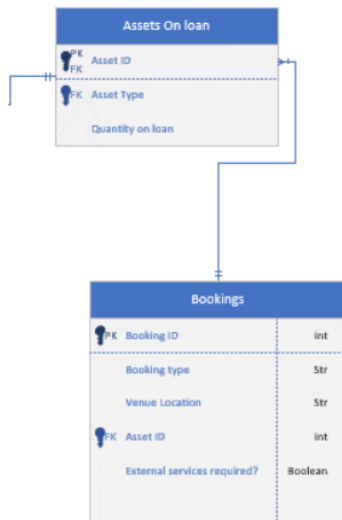
From there I decided to create the department entity and produce another two entities from there. The first being Assets at department, this has a relationship to department of one-to-many, i/e there are many assets at a given department where appropriate. However, there is yet another layer of abstraction, do bear with me.

This entity yet again splits into two others, “Assets on loan” and of course the ever almighty “Bookings” table where the customer view is really focused upon.

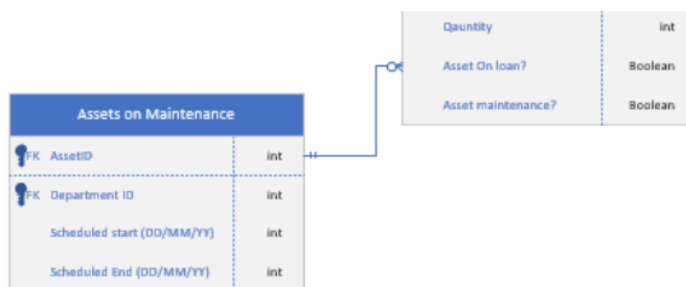
Now the “assets on loan” entity is naturally related to “Assets at department” in a many-to-one manner stemming from assets at department. The thinking here is that there are many assets that can be at an instance on loan.

I decided to stem bookings off assets on loan in a many-to-one relationship. This as I felt that in each instance there will be many assets per a booking, say 3 or more speakers with some lights.

So, in terms of the keys enforcing these relationships are as follow:



Asset ID stems from the assets table, as such it behaves as a primary key and a foreign key, it forms a composite primary key, this being the combination of two or more columns that together uniquely identify each row in the table, when both keys are applied (as required) it guarantees the uniqueness of that row. [5]



Lastly here is my assets on maintenance table. AssetID forms a foreign key paired with department ID which together form a foreign composite primary key which guarantees uniqueness in a given instance of Assets on maintenance, it is related from the Assets In department, in a many to one manner. As many assets may be in each instance under maintenance.

Adhering to normal form conventions:

In the diagram you may note the presence of “bridging” tables such as Staff at Branch, Departments at Branch and of course Staff at Department.

Well, these exist so that my database will conform to a set of standards that have been in place since E.F Codd himself produced the concept for the relational database! These being 1st, 2nd, and 3rd normal form.

To reach the desired 3rd normalisation we must first adhere to of course, 1st and second normal form! This is done by first ensuring that each attribute in the relation is atomic and cannot be further simplified into any other meaningful components [6]. After it has been first normalised, we must then ensure that there are no ‘non’ prime attributes dependant on any subset of candidate keys. Put simply that every atomic attribute only depends on one and only one primary key, composite or otherwise. [7]

Finally, once we have formatted the tables as so above, we can then ensure that they conform to third normal form. It should be mentioned that the primary reason for using this standard is to reduce the level of redundancy as we increase the levels of normal form, we simply wish to reduce the risk of update anomalies. And to ensure the database is in a consistent state. [8]

The greatest risk to 2nd normal form is a transitive anomaly, where indirect relationships between attributes create issues, as such by ensuring that only prime attributes form dependencies in relations, we can eliminate this risk and push the table into 3rd normal form. [8]

The database!

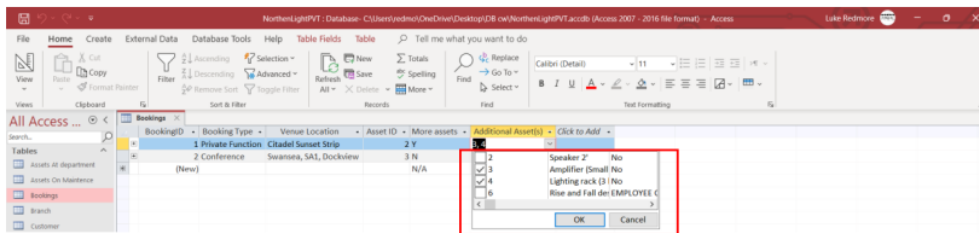
Input Validation and Tables:

Bookings...

BookingID	Booking Type	Venue Location	Asset ID	More assets	Additional Asset(s)
	Private Function	Citadel Sunset Strip	2 Y		3, 4
	2 Conference	Swansea, SA1, Dockview	3 N		
	(New)			N/A	

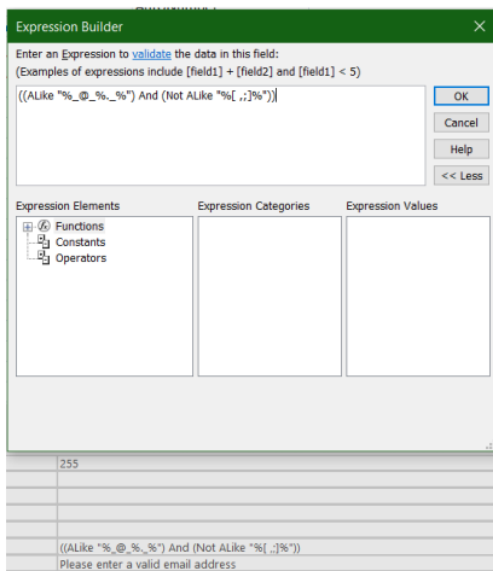
To ensure that all possible needs for any venue are met I realised that perhaps only one asset per booking would create issues down the line. So, I created a lookup field ‘Additional assets’ that enable multiple Asset’s to be assigned under the same booking.

Here is an example of that working...



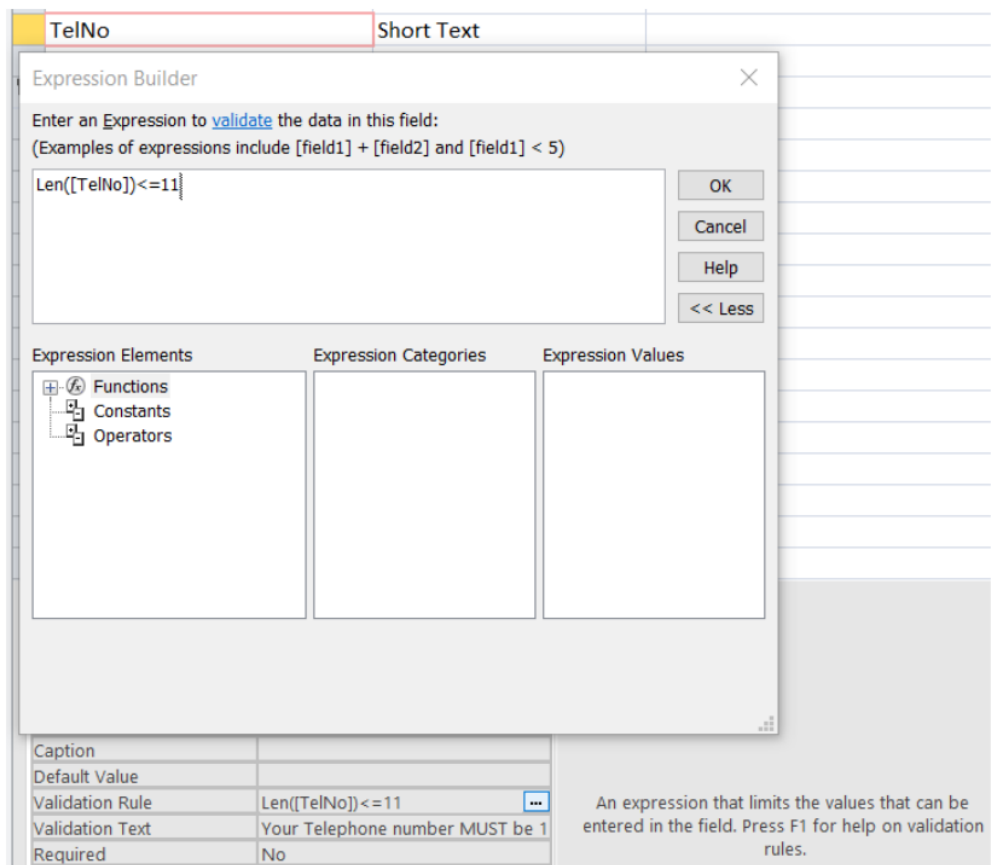
To provide input validation for the 'More assets' Field I used the syntax:

Input validation on emails and phone numbers for the customer table:



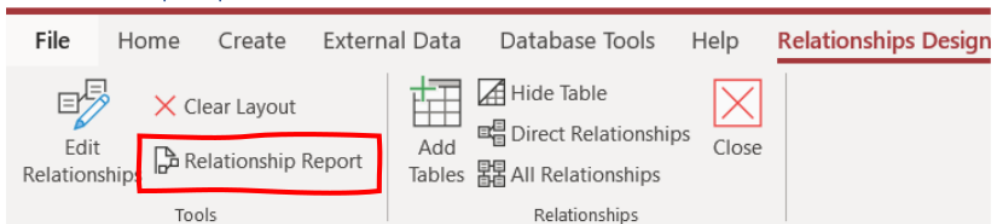
*This requires that the email String has ascii characters and an @ symbol and is not akin to anything that is not that so special characters like * or a space cause an error to be thrown! Regarding ASCII, I used the operand ALIKE not LIKE as emails can have ASCII wildcards which throw errors when the like pattern match is executed.*

For phone numbers, note that this is to validate standard UK number lengths and may throw errors if another area code is used!

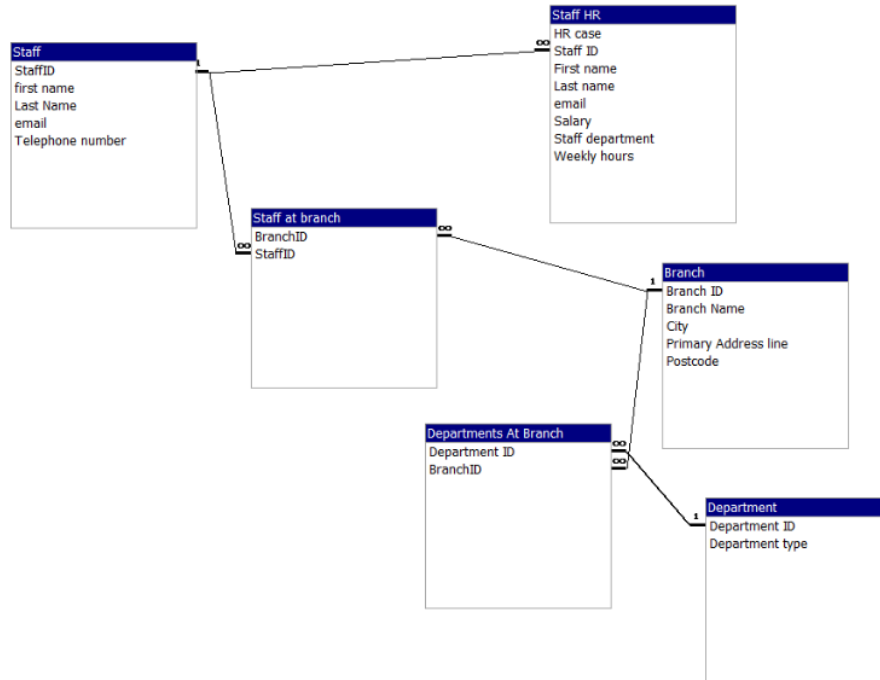


This just ensures that the length is less than or equal to 11 numbers.

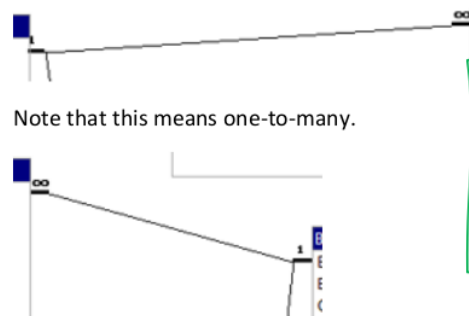
Relationship report



To generate my relationships report I used the feature highlighted in red.



To show consistency with my ER diagram, here is a generated report of my entities with their given relationships.



Note that this means one-to-many.

And this being many-to-one, seeing as I discussed in depth the relationships between entities in my ER diagram above. I will merely leave this here as a database view of those in action!

Reports and queries

```

Format for Staff At Branch
alter table [staff at branch] drop column [first name]
    
```

Note that this was to format my staff at branch table for sake of data integrity and concise nature of my data.

Assets at department:

Assets At department Query

```
SELECT [Assets At department].[Department Asset ID], [Assets At department].[Quantity], [Assets At department].[Asset Scheduled for maintenance?], [Assets On Maintenance].[Quantity for maintenance], ([Quantity]-[Quantity for maintenance]) AS [total left(After maintenance)]
FROM [Assets At department] INNER JOIN [Assets On Maintenance] ON ([Assets At department].[Department Asset ID]) = ([Assets On Maintenance].[AssetID])
```

Assets At department Query

Department Asset ID	Quantity	Asset Scheduled for maintenance?	Quantity for maintenance	total left(After maintenance)
2	5	Yes	3	2
6	30	Yes	15	15

(Query for bookings and multiple assets per function) Assets At department Query

Field:	Department Asset ID	Quantity	Asset Scheduled for ma	Quantity for maintenanc	total left(After maintenance): [Quantity]-[Quantity for maintenance]
Table:	Assets At department	Assets At department	Assets At department	Assets On Maintenance	
Sort:					
Show:	☒	☒	☒	☒	☒
Calculate:					

Here is the query design for my maintenance table, note that there are two sources used here 'Assets at department' and 'Assets on maintenance' these two sources are used to show the quantity overall, and the amount that will be on maintenance.

The field of note here however is 'total left (After maintenance)' this is a calculated field. And to allow the database engine to call upon field [quantity] and then subtract that value with [quantity for maintenance] giving the total amount left that are available to be rented out. (Calculated field highlighted red)

Here's how that looks:

Department Asset ID	Quantity	Asset Scheduled for maintenance?	Quantity for maintenance	total left(After maintenance)
2	5	Yes	3	2
6	30	Yes	15	15

Forms

[illegible]

Customer Form

CustomerID

Customer Name John Shepard

TelNo 07312118007

Email shepardcommander@SSV2.net

Booking ID Private Function

Add New Customer

Macros used!

You'll note that the Bookings form opens when this access database image is opened, how did I do this? Well with Macros!

autoexec

OpenForm

Form Name Booking Form

View Form

Filter Name

Where Condition

Data Mode Edit

Window Mode Normal

+ Add New Action

For simplicity's sake, here is an overview of my macro, note that the action dictated is 'openForm' 'Booking Form'.

So how do I get it to run when the database is opened?

Well by calling the macro 'autoexec' this harks back to the old bios days but put simply this will be executed before any queries or VBA codes are run! [9]

Gallery of all tables used in database:

Assets At department	Assets On Maintenance	Bookings	Branch	Customer	Department	Departments At Branch	Staff	Staff at branch	Staff HR
Department Asset ID	Department ID	Asset Type	Quantity	Asset On loan?	Asset Scheduled for maintenance?	Click to Add			
1 Sound engineering	3	Speaker 2'	5	No	Yes				
3 Sound engineering		Amplifier (Small)	10	No	No				
4 Light and visual		Lighting rack (3 lights, rgb)	10	No	No				
6 Employee support		Rise and Fall desk	30	EMPLOYEE ONLY	Yes				
(New)				N/A	N/A				

Assets At department	Assets On Maintenance	Bookings	Branch	Customer	Department	Departments At Branch	Staff	Staff at branch	Staff HR
AssetID	DepartmentID	Quantity for maintenance	Scheduled Start (DD/MM/YY)	Scheduled End (DD/MM/YY)	Click to Add				
1 Sound engineering	3		21/06/2023	10/07/2023					
6 Employee support	15		25/04/2023	30/04/2023					

Assets At department	Assets On Maintenance	Bookings	Branch	Customer	Department	Departments At Branch	Staff	Staff at branch	Staff HR
BookingID	Booking Type	Venue Location	Asset ID	More assets	Additional Asset(s)	Click to Add			
1 Private Function		Citadel Sunset Strip	2 Y	3, 4					
2 Conference		Swansea, SA1, Dockview	3 N						
(New)			N/A						

Assets At department	Assets On Maintenance	Bookings	Branch	Customer	Department	Departments At Branch	Staff	Staff at branch	Staff HR
Branch ID	Branch Name	City	Primary Address line	Postcode	Click to Add				
1 Swansea-Bay	Swansea	23, Dock view Road	SA1 4J5						
2 Swansea-Centre	Swansea	07, Quay Parade	SA1 3EE						
3 Vale of Neath	Neath	8, Old Market Street	SA11 3NA						
4 Glamorgan Valleys-Clyne	Neath	Lletty Dafydd	SA11 4BW						
5 RedTech Headquarters	Cardiff	Bridge St, Cardiff	CF10 2TR						
(New)									

Assets At department	Assets On Maintenance	Bookings	Branch	Customer	Department	Departments At Branch	Staff	Staff at branch	Staff HR
CustomerID	Customer Name	TelNo	Email	Booking ID	Click to Add				
1 John Shepard	07312118007	shepardcomma	Private Function						
10 Luke Redmore	1876113457	redmoreluke@	Conference						
(New)									

Assets At department	Assets On Maintenance	Bookings	Branch	Customer	Department	Departments At Branch	Staff	Staff at branch	Staff HR
Department ID	Department type	Click to Add							
1 Sound engineering									
2 Light and visual									
3 Administration									
4 Custodial									
5 External services									
6 Legal services									
7 Employee support									
(New)									

Assets At department	Assets On Maintenance	Bookings	Branch	Customer	Department	Departments At Branch	Staff	Staff at branch	Staff HR
Department ID	BranchID	Click to Add							
1	1								
1	5								
2	1								
2	5								
3	1								
4	5								
5	5								
6	5								

StaffID	first name	Last Name	email	Telephone number	Click to Add
1	Luke	Redmore	luke.redmore@redtech.com	0911500011	
2	David	Kingston	david.kingston@redtech.com	0911500012	
3	Jess	Lewis	jess.lewis@redtech.com	0911500123	
4	Jackson	Sexton	Sexton.J@redtech.com	0911500124	
(New)					

BranchID	StaffID	Click to Add
1	2	

HR case	Staff ID	First name	Last name	email	Salary	Staff department	Weekly hours	Click to Add
1	2	Luke	Redmore	luke.redmore@redtech.com	£23,000.00	Sound engineering		
2	2	David	Kingston	david.kingston@redtech.com	£23,000.00	Sound engineering		
3	3	Jess	Lewis	jess.lewis@redtech.com	£23,000.00	Light and visual	0	
(New)					£0.00	Administration	0	
						Custodial		
						External services		
						Legal services		
						Employee support		

References

- [1] geeksforgeeks, "Different types of database users," geeksforgeeks, 02 December 2022. [Online]. Available: <https://www.geeksforgeeks.org/different-types-of-database-users/>. [Accessed 07 March 2023].
- [2] geeksforgeeks, "Introduction of ER Model," geeksforgeeks, 17 January 2023. [Online]. Available: <https://www.geeksforgeeks.org/introduction-of-er-model/>. [Accessed 10 March 2023].
- [3] GeeksforGeeks, "Difference between Strong and Weak Entity," GeeksforGeeks, 22 February 2023. [Online]. Available: <https://www.geeksforgeeks.org/difference-between-strong-and-weak-entity/>. [Accessed 24 March 2023].

- [4] GeeksforGeeks, "Partial, Unique, Secondary, Composite and Surrogate keys in DBMS," GeeksforGeeks, 22 September 2020. [Online]. Available: <https://www.geeksforgeeks.org/partial-unique-secondary-composite-and-surrogate-keys-in-dbms/>. [Accessed 24 March 2023].
- [5] GeeksforGeeks, "Composite Key in SQL," GeeksforGeeks, 2021 April 26. [Online]. Available: <https://www.geeksforgeeks.org/composite-key-in-sql/>. [Accessed 30 March 2023].
- [6] O. library, "7.3.1: Atomic Attributes," Open library, [Online]. Available: https://ecampusontario.pressbooks.pub/relationaldatabasesandmicrosoftaccess365/chapter/_unknown__-59/. [Accessed 22 March 2023].
- [7] A. Kozubek, "Normalization in Relational Databases: First Normal Form (1NF), Second Normal Form (2NF), and Third Normal Form (3NF)," Vertabelo, 1 December 2020. [Online]. Available: <https://vertabelo.com/blog/normalization-1nf-2nf-3nf/>. [Accessed 22 March 2023].
- [8] GeeksforGeeks, "Third Normal Form (3NF)," GeeksforGeeks, 31 July 2019. [Online]. Available: <https://www.geeksforgeeks.org/third-normal-form-3nf/>. [Accessed 2 April 2023].
- [9] Microsoft, "Create a macro that runs when you open a database," Microsoft, 2021. [Online]. Available: <https://support.microsoft.com/en-us/office/create-a-macro-that-runs-when-you-open-a-database-98ba1508-dcc6-4e0f-9698-a4755e548124>. [Accessed 21 April 2023].
- [10] PNGEGG, Artist, *Microsoft application logo, Microsoft Access Microsoft Office 2016 Database*] *Microsoft Word, MS Access, text, trademark png*. [Art]. 2016.

FINAL GRADE

GENERAL COMMENTS

47 /100

Instructor

Generic Assessment Criteria	47
1. Engagement with Literature Skills	
Not Applicable	
2. Knowledge and Understanding Skills	
It is a good attempt trying to provide a brief explanation about database management system, however description about the scenario is missing. Suitable design methodologies for creating the database system must be employed and justified. After your initial investigation, the following user requirements were to be highlighted: . Different type of users who might access the Database Management System were fairly identified. The database administrators who is responsible for the design of the database system and staff working	13

in the organization should have different access rights. - Information about branch manager and his privileges are missing. - Identification of different type of departments each branch might require is missing essential features. - Primary key and foreign key in UML diagram are nicely represented. But the labels in the UML diagram are not clearly visible.	
3. Cognitive and Intellectual Skills	
- Evidence showing the justification for the links between tables is missing. - Verification of third normal form in table design is not been provided. - screen shots provided for creating table is missing few descriptions. - usage of an MS access functionality to produce few calculations in forms and generating invoice as a query result is missing. However, some additional querying features and it's screen dumps could have been provided.	5

4. Practical Application Skills	
· Implementation of Northern lights database should meet the requirements using the Entity Relationship Model that you have developed. · Demonstration of the database system is fairly done.	2 5
5. Transferable Skills for Life and Professional Practice	
· The front page includes the name and topic · Some of the screen shots along with the description are missing essential justification. · Your writing should reflect a professional tone. Avoid using slang or informal language that may come across as unprofessional. Try to use more formal language in your writing. · Proper grammar and punctuation are essential when writing in a professional setting. Make sure you proofread for any errors before submitting it. · A brief user guide for navigation between the system is missing.	4

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14

PAGE 15

PAGE 16

PAGE 17
