

|  |
| --- |
| DAT601 Milestone 3 |
| Physical Database Design |
| Name: Rhylei Tremlett  ID: 13513865  Date: 25/06/2023 |

Table of Contents

[Translate Logical Model to Physical Model 1](#_Toc138622032)

[Estimate Required Disk Space 2](#_Toc138622033)

[SQL Queries 3](#_Toc138622034)

[Query A 3](#_Toc138622035)

[Query B 3](#_Toc138622036)

[Query C 4](#_Toc138622037)

[Query D 4](#_Toc138622038)

[Query E 5](#_Toc138622039)

[Query F 5](#_Toc138622040)

[Query G 6](#_Toc138622041)

[Query H 6](#_Toc138622042)

[Query I 6](#_Toc138622043)

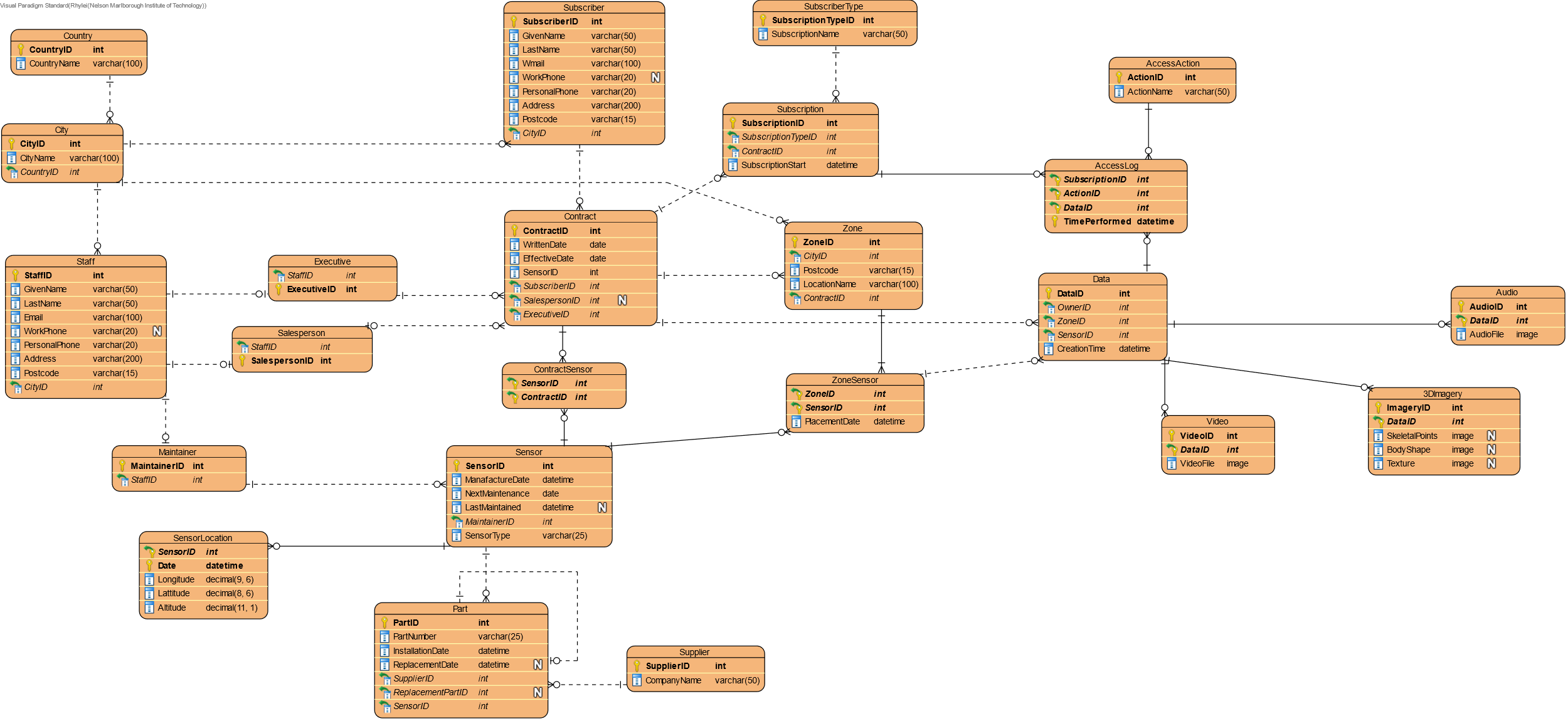
[Query J 7](#_Toc138622044)

[Issues Faced 8](#_Toc138622045)

# Translate Logical Model to Physical Model

I decided to try to use a different naming convention. I chose to use a convention similar to the Microsoft AdventureWorks database. Most notably, I have changed my column names from snake case to pascal case.

No other changes have been made other than the automatic conversion of datatype to MSSQL done by Visual Paradigm.





# Estimate Required Disk Space



I have chosen not to calculate the data recorded by the sensors as it is impossible to guess the number of records. Also, my design for data is terrible and inflates the size to 800 terabytes when I include it.

The ballpark estimate for the database without including data is 30GB. With data, it could be anywhere from 100 terabytes to 10 petabytes.

# SQL Queries

This section contains my queries, a breif description of their purpose, an example of their result.

Double-clicking the files should allow you to open them in an editor.

## Query A

“A salesperson subscribes to a new standard subscription to a 3D sensor. The transaction receives the salesperson Id, a discount %, all subscriber details, and a 3D sensor ID.”

My transaction creates a new record for the subscriber and then uses the ID of the newly made record to create a contract record and a subscription record.

My transaction doesn’t include any discounts because I did not model that in my database.



## Query B

“For each salesperson list the subscribers they have sold a subscription to. The transaction receives the salesperson's name as input, and presents each name, address, and the % they were discounted.”

My transaction shows all of the customers who were sold a subscription by a salesperson. Like Query A, this transaction doesn’t include discounts.

The result below is for a salesperson named “Hector Bim”.

A screenshot of a computer

Description automatically generated with medium confidence



## Query C

“Write SQL to be used to insert data from a 3D sensor to its stored data on the Spaces database. The transaction receives the 3D sensor ID and all the data from a data stream. That is made up of one or more records of 3D human imagery – texture, body shape and estimated skeletal points, Human voice, Shared audio from other sources, time, longitude, latitude, and altitude.”

My transaction creates a new data record using the provided sensor ID. The transaction then creates each of the data subtypes using the provided data.



## Query D

“List the location in latitude, longitude coordinates, of each 3D sensor that is currently in a contract (subscribed to). The transaction presents the Contracting organisation or person’s name, a 3D sensor ID, a Latitude, and a Longitude.”

My transaction retrieves the location of each sensor as well as the customer who owns it and the ID of their contract. My database also includes an altitude column.

Sensors appear more than once because I could not constrain Mockaroo to use a foreign key only once.

A screenshot of a computer

Description automatically generated with medium confidence



## Query E

“For a contract list all the data collected. The transaction receives the contracting organisation'sname and presents for each collected data record, the contracting organisation's name, a 3D sensor ID, 3D human imagery – texture, body shape AND estimated skeletal points, Human voice, Shared audio from other sources, time, longitude, latitude, and altitude”

My transaction retrieves all of the data belonging to a specific contract. There are more columns hidden to the right.

A screenshot of a computer

Description automatically generated with low confidence



## Query F

“For each 3D sensor present the list of subscribers who are viewing a live 3D video stream. The transaction lists 3D sensor ID, Subscriber Name, Stream ID.”

I didn’t model video streams into my database because I don’t know what it is or how to model it, so here is the next closest thing. My database stores a log of all the actions performed by a user, including viewing data.

My transaction retrieves for each sensor when a resource (I can’t specify video) was most recently accessed. The procedure returns the sensor Id, the subscriber's name, the action they took (view), the time performed and the data ID they accessed.

A picture containing text, screenshot, number, font

Description automatically generated



## Query G

“For a given 3D sensor list all the suppliers of parts. The transaction receives the 3D sensor ID, and presents the Supplier Name and, Part Name.”

My transaction retrieves a list of suppliers for all of the parts that aren’t replaced, which belong to the sensor. My database doesn’t have a ‘Part Name’ and instead has a part number.

A screenshot of a computer

Description automatically generated with medium confidence



## Query H

“Update the location and Zone of a 3D sensor. The transaction receives the 3D sensor ID, a location and a Zone expressed as a list of coordinates in latitude, longitude pairs. It updates the location of the 3D sensor and its corresponding Zone. (This transaction may require more than one update query.)”

This query just isn’t possible for my database in the way described above, I can achieve a similar result, however.

My transaction inserts a new record into sensor location that contains the updated longitude, latitude and altitude. The transaction then updates the zone sensor join table to update the zone the sensor is now in.



## Query I

“Delete the data collected for a given Contract. The transaction receives a Contract ID, the data collected for a Contract is deleted.”

My transaction utilises a while loop to emulate the functionality of a cascade delete.



## Query J

“Display the total cost of all parts replaced in maintenance of a 3D sensor. The transaction displays the 3D sensor ID, Total Cost of replaced parts, for every 3D sensor.”

My transaction retrieves all parts that have been replaced in every sensor. My database does not include any costs of parts, however it would be trivial to include them in the select list if they were there.

A screenshot of a computer

Description automatically generated with medium confidence



# Issues Faced

While generating test data and creating the procedures to execute, I ran into an issue where data was converted to an integer on a column that takes varchar. The data that I’ve generated is not numeric, and it includes symbols and text. SQL Server then would throw an error as it fails the conversion.

Another problem I had was with Mocakroo, I realised when testing the insert script that I had a lot of foreign key constraint errors. This turned out to be because of one statement failing, which caused the rest to fail because the rows were missing. The problem that occurred was that I had improperly generated data for one of my very simple tables (executive, two columns). I had the data in the wrong columns, and this meant that the row referenced later in the script didn’t exist.

I faced many issues during the queries where my database design was lacking the fundamental data required for the query. It is most notable in query H. I realised that my database design was a decent distance off the mark and decided to scrape together what I could for the queries.

Because my design ended up being of such poor quality, I decided not to do the transaction analysis, views and roles. I am aware of the marks it will cost me, but I believe that completing that work on such a poor design of mine merited no benefit.

Another result of the poor design is that my database size estimate was redundant as I could not get an estimate for how much size the sensor data could take up. This is because the size of the data is arbitrary and based on my interpretation of the scenario. Since I still don’t understand it, I couldn’t properly model the data and, therefore, cannot estimate its size.

What I have learned is that even though I read through the transactions in the beginning for my conceptual model, I did not check again at the later stages. This resulted in my design veering off the track.

I have also learned the importance of clarification. Had I spent more time trying to understand the scenario, I could have created a better design and would not have faced these issues and mistakes.

In future, I plan to try and implement a process similar to the unified process in my work, where I can work on development and design at the same time to make sure that everything stays on track.