

CEGEG129: Spatial Databases and Data Management
Assignment 2 – Spatial Data Management

Table of Contents

Part A – Database Creation 2

 Revised Conceptual ERD 2

 Logical ERD 3

 SQL Scripts..... 4

 SQL Scripts – Table Population 11

 QGIS Map 18

 FME Screenshot of 3D Data 19

 Functional Requirements 20

Part B – Option 1: GeoMedia 3D..... 27

References 29

Appendix..... 31

Part A – Database Creation

1 – Revised Conceptual ERD

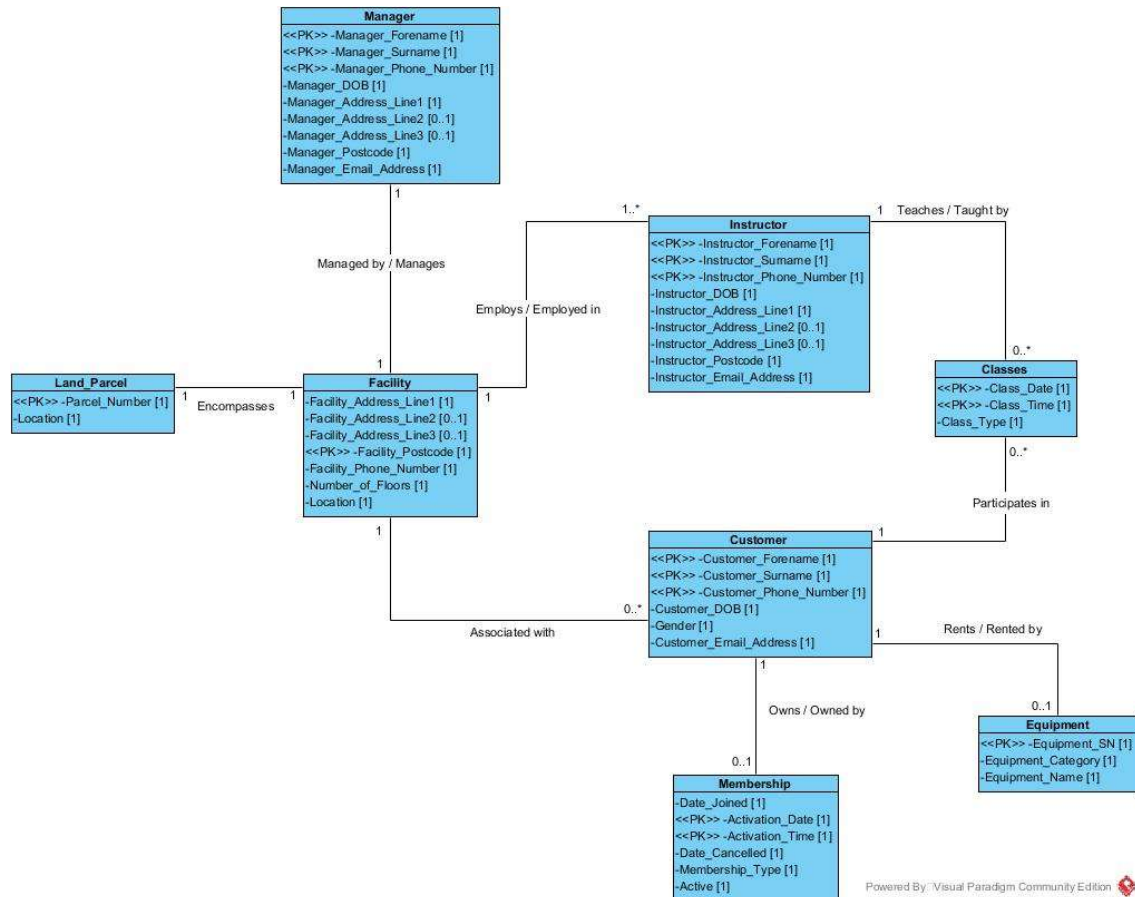


Figure 1: Revised Conceptual Entity Relationship Diagram¹

¹ This conceptual ERD differs slightly to the diagram submitted as part of assignment one as a result of the feedback received. The original conceptual ERD from assignment one is presented in the appendix with a list of changes made.

2 – Logical ERD

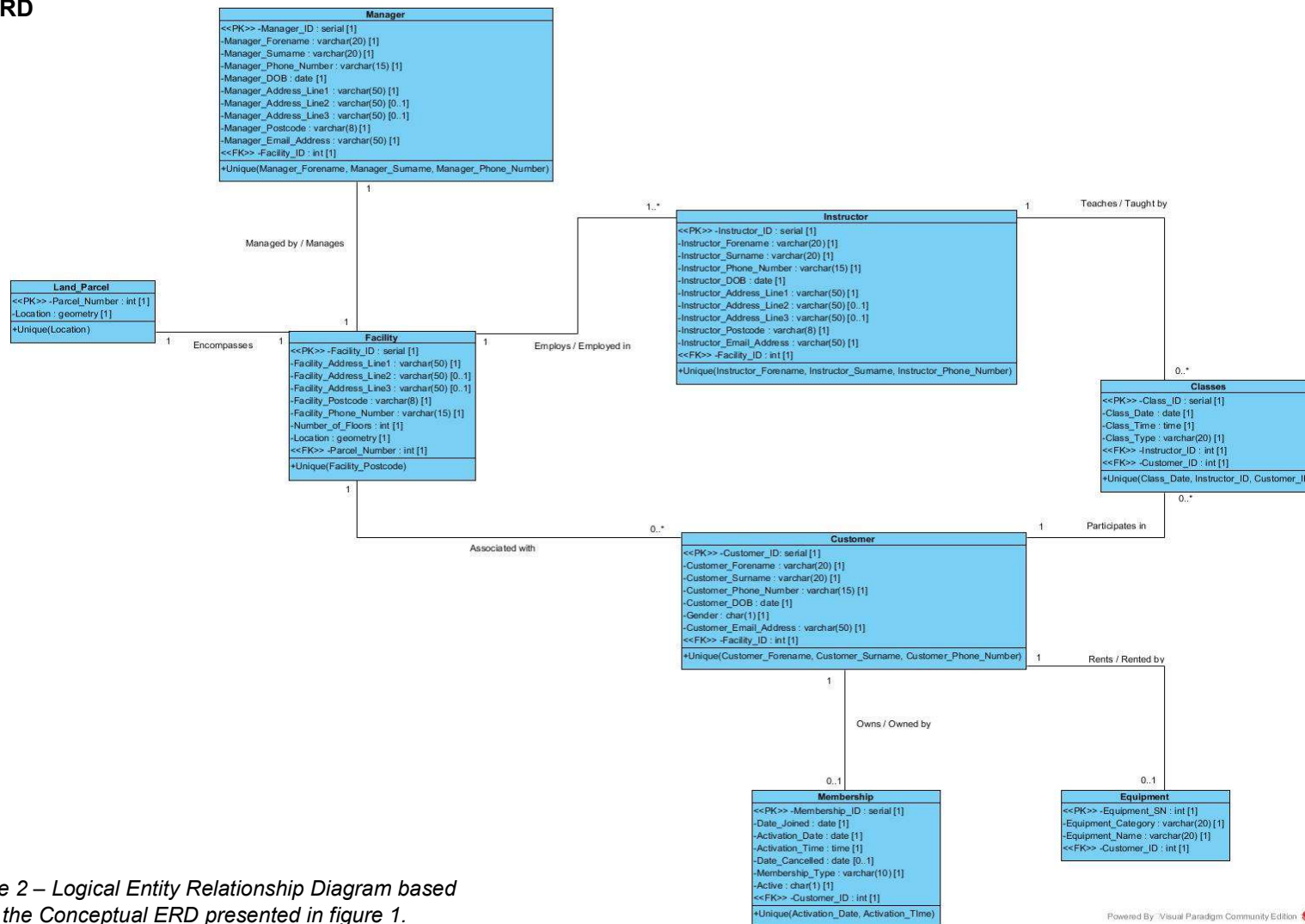


Figure 2 – Logical Entity Relationship Diagram based upon the Conceptual ERD presented in figure 1.

3 – SQL Scripts

Table 1 – SQL scripts to create tables and their respective primary key, foreign key and unique key constraints.

Entity Name	Create Table Script	Primary Key Constraints	Foreign Key Constraints	Unique Key Constraints
Land Parcel	<pre>CREATE TABLE public.Land_Parcel(Parcel_Number INTEGER NOT NULL, Location GEOMETRY);</pre>	<pre>ALTER TABLE public.Land_Parcel ADD CONSTRAINT Land_Parcel_pk PRIMARY KEY (Parcel_Number);</pre>		<pre>ALTER TABLE public.Land_Parcel ADD CONSTRAINT Land_Parcel_unique UNIQUE (location);</pre>
Facility	<pre>CREATE TABLE public.Facility (Facility_ID SERIAL NOT NULL, Facility_Address_Line1 VARCHAR(50) NOT NULL, Facility_Address_Line2 VARCHAR(50), Facility_Address_Line3 VARCHAR(50), Facility_Postcode VARCHAR(8) NOT NULL, Facility_Phone_Number VARCHAR(15) NOT NULL, Number_of_Floors INTEGER NOT NULL, Location GEOMETRY, Parcel_Number INTEGER);</pre>	<pre>ALTER TABLE public.Facility ADD CONSTRAINT Facility_pk PRIMARY KEY (Facility_ID);</pre>	<pre>ALTER TABLE public.Facility ADD CONSTRAINT Facility_Land_Parcel_fk FOREIGN KEY (Parcel_Number) REFERENCES public.Land_Parcel (Parcel_Number);</pre>	<pre>ALTER TABLE public.Facility ADD CONSTRAINT Facility_unique UNIQUE (Facility_Postcode);</pre>

Manager	<pre>CREATE TABLE public.Manager (Manager_ID SERIAL NOT NULL, Manager_Forename VARCHAR(20) NOT NULL, Manager_Surname VARCHAR(20) NOT NULL, Manager_Phone_Number VARCHAR(15) NOT NULL, Manager_DOB DATE NOT NULL, Manager_Address_Line1 VARCHAR(50) NOT NULL, Manager_Address_Line2 VARCHAR(50), Manager_Address_Line3 VARCHAR(50), Manager_Postcode VARCHAR(8) NOT NULL, Manager_Email_Address VARCHAR(50) NOT NULL, Facility_ID INTEGER);</pre>	<pre>ALTER TABLE public.Manager ADD CONSTRAINT Manager_pk PRIMARY KEY (Manager_ID);</pre>	<pre>ALTER TABLE public.Manager ADD CONSTRAINT Manager_Facility_fk FOREIGN KEY (Facility_ID) REFERENCES public.Facility (Facility_ID);</pre>	<pre>ALTER TABLE public.Manager ADD CONSTRAINT Manager_unique UNIQUE (Manager_Forename, Manager_Surname, Manager_Phone_Number);</pre>
Instructor	<pre>CREATE TABLE public.Instructor (Instructor_ID SERIAL NOT NULL, Instructor_Forename VARCHAR(20) NOT NULL, Instructor_Surname VARCHAR(20) NOT NULL, Instructor_Phone_Number VARCHAR(15) NOT NULL, Instructor_DOB DATE NOT NULL, Instructor_Address_Line1 VARCHAR(50) NOT NULL,</pre>	<pre>ALTER TABLE public.Instructor ADD CONSTRAINT Instructor_pk PRIMARY KEY (Instructor_ID);</pre>	<pre>ALTER TABLE public.Instructor ADD CONSTRAINT Instructor_Facility_fk FOREIGN KEY (Facility_ID) REFERENCES public.Facility (Facility_ID);</pre>	<pre>ALTER TABLE public.Instructor ADD CONSTRAINT Instructor_unique UNIQUE (Instructor_Forename, Instructor_Surname, Instructor_Phone_Number);</pre>

	Instructor_Address_Line2 VARCHAR(50) , Instructor_Address_Line3 VARCHAR(50) , Instructor_Postcode VARCHAR(8) NOT NULL , Instructor_Email_Address VARCHAR(50) NOT NULL , Facility_ID INTEGER) ;			
Customer	CREATE TABLE public.Customer (Customer_ID SERIAL NOT NULL , Customer_Forename VARCHAR(20) NOT NULL , Customer_Surname VARCHAR(20) NOT NULL , Customer_Phone_Number VARCHAR(15) NOT NULL , Customer_DOB DATE NOT NULL , Gender CHAR(1) NOT NULL , Customer_Email_Address VARCHAR(50) NOT NULL , Facility_ID INTEGER) ;	ALTER TABLE public.Customer ADD CONSTRAINT Customer_pk PRIMARY KEY (Customer_ID) ;	ALTER TABLE public.Customer ADD CONSTRAINT Customer_Facility_fk FOREIGN KEY (Facility_ID) REFERENCES public.Facility (Facility_ID) ;	ALTER TABLE public.Customer ADD CONSTRAINT Customer_unique UNIQUE (Customer_Forename , Customer_Surname , Customer_Phone_Number) ;
Membership	CREATE TABLE public.Membership (Membership_ID SERIAL NOT NULL , Date_Joined DATE NOT NULL , Activation_Date DATE NOT NULL , Activation_Time TIME NOT NULL , Date_Cancelled DATE ,	ALTER TABLE public.Membership ADD CONSTRAINT Membership_pk PRIMARY KEY (Membership_ID) ;	ALTER TABLE public.Membership ADD CONSTRAINT Membership_Customer_fk FOREIGN KEY (Customer_ID) REFERENCES public.Customer (Customer_ID) ;	ALTER TABLE public.Membership ADD CONSTRAINT Membership_unique UNIQUE (Activation_Date , Activation_Time) ;

	Membership_type VARCHAR(10) NOT NULL, Active CHAR(1) NOT NULL, Customer_ID INTEGER);			
Classes	CREATE TABLE public.Classes (Class_ID SERIAL NOT NULL, Class_Date DATE NOT NULL, Class_Time TIME NOT NULL, Class_Type VARCHAR(20) NOT NULL, Instructor_ID INTEGER, Customer_ID INTEGER);	ALTER TABLE public.Classes ADD CONSTRAINT Classes_pk PRIMARY KEY (Class_ID);	ALTER TABLE public.Classes ADD CONSTRAINT Classes_Instructor_fk FOREIGN KEY (Instructor_ID) REFERENCES public.Instructor (Instructor_ID); ALTER TABLE public.Classes ADD CONSTRAINT Classes_Customer_fk FOREIGN KEY (Customer_ID) REFERENCES public.Customer(Customer_ID);	ALTER TABLE public.Classes ADD CONSTRAINT Class_Instructor_unique UNIQUE (Class_Date, Instructor_ID, Customer_ID);
Equipment	CREATE TABLE public.Equipment (Equipment_SN INTEGER NOT NULL, Equipment_Category VARCHAR(20) NOT NULL, Equipment_Name VARCHAR(20) NOT NULL, Customer_ID INTEGER);	ALTER TABLE public.Equipment ADD CONSTRAINT Equipment_pk PRIMARY KEY (Equipment_SN);	ALTER TABLE public.Equipment ADD CONSTRAINT Equipment_Customer_fk FOREIGN KEY (Customer_ID) REFERENCES public.Customer(Customer_ID);	

3 (continued) – Additional SQL scripts. Check Constraints used to maintain data integrity.

Table 2 – SQL scripts to create check constraints for the appropriate tables.

Regular Expression sources: Stack Overflow User (2011), Stack Overflow User (2016), Wright (2008).

Entity Name	Check Constraint
Facility	<pre>ALTER TABLE public.Facility ADD CONSTRAINT Facility_floor_check CHECK (Number_of_Floors > 0); ALTER TABLE public.Facility ADD CONSTRAINT Facility_phone_check CHECK (Facility_Phone_Number ~* '^(((\+44\s?\d{4}) (?0\d{4})\?)\s?\d{3}\s?\d{3}) ((\+44\s?\d{3}) (?0\d{3})\?)\s?\d{3}\s?\d{4}) ((\+44 \s?\d{2}) (?0\d{2})\?)\s?\d{4}\s?\d{4}))(\s?\#(\d{4} \d{3}))?\$'); ALTER TABLE public.Facility ADD CONSTRAINT Facility_postcode_check CHECK (Facility_Postcode ~* '^ ?([BEGLMNSWbeglmnsW][0-9][0-9]?) ([A-PR-UWYZa-pr-uwyz][A-HK-Ya-hk-y][0-9][0-9]?) ([ENWenw][0-9][A- HJKSTUWa-hjkstuw]) ([ENWenw][A-HK-Ya-hk-y][0-9][ABEHMNPRVWXyabehmnprvwxy])) ?[0-9][ABD-HJLNP-UW- Zabd-hjlnp-uw-z]{2}\$');</pre>
Manager	<pre>ALTER TABLE public.Manager ADD CONSTRAINT Manager_email_check CHECK (Manager_Email_Address ~* '^ [A- Za-z0-9._%~]+@[A-Za-z0-9.-]+[.] [A-Za-z]+\$'); ALTER TABLE public.Manager ADD CONSTRAINT Manager_phone_check CHECK (Manager_Phone_Number ~* '^(((\+44\s?\d{4}) (?0\d{4})\?)\s?\d{3}\s?\d{3}) ((\+44\s?\d{3}) (?0\d{3})\?)\s?\d{3}\s?\d{4}) ((\+44 \s?\d{2}) (?0\d{2})\?)\s?\d{4}\s?\d{4}))(\s?\#(\d{4} \d{3}))?\$'); ALTER TABLE public.Manager ADD CONSTRAINT Manager_postcode_check CHECK (Manager_Postcode ~* '^ ?([BEGLMNSWbeglmnsW][0-9][0-9]?) ([A-PR-UWYZa-pr-uwyz][A-HK-Ya-hk-y][0-9][0-9]?) ([ENWenw][0-9][A- HJKSTUWa-hjkstuw]) ([ENWenw][A-HK-Ya-hk-y][0-9][ABEHMNPRVWXyabehmnprvwxy])) ?[0-9][ABD-HJLNP-UW- Zabd-hjlnp-uw-z]{2}\$');</pre>

Instructor	<pre>ALTER TABLE public.Instructor ADD CONSTRAINT Instructor_email_check CHECK (Instructor_email_Address ~* '^[A-Za-z0-9._%~]+@[A-Za-z0-9.-]+[.][A-Za-z]+\$');</pre> <pre>ALTER TABLE public.Instructor ADD CONSTRAINT Instructor_postcode_check CHECK (Instructor_Postcode ~* '^ ? ([BEGLMNSWbeglmnsw] [0-9] [0-9] ?) (([A-PR-UYZa-pr-uwyz] [A-HK-Ya-hk-y] [0-9] [0-9] ?) (([ENWenw] [0-9] [A-HJKSTUWa-hjkstuw]) ([ENWenw] [A-HK-Ya-hk-y] [0-9] [ABEHMNPRVWXYabehmnprvwxY]))) ? [0-9] [ABD-HJLNP-UW-Zabd-hjlnp-uw-z] {2} \$');</pre> <pre>ALTER TABLE public.Instructor ADD CONSTRAINT Instructor_phone_check CHECK (Instructor_Phone_Number ~* '^ (((\+44\s?\d{4}) (\?0\d{4})\?)\s?\d{3}\s?\d{3}) ((\+44\s?\d{3}) (\?0\d{3})\?)\s?\d{3}\s?\d{4}) ((\+44\s?\d{2}) (\?0\d{2})\?)\s?\d{4}\s?\d{4})) (\s?\#(\d{4} \d{3}))?\$');</pre>
Customer	<pre>ALTER TABLE public.Customer ADD CONSTRAINT Customer_phone_check CHECK (Customer_Phone_Number ~* '^ (((\+44\s?\d{4}) (\?0\d{4})\?)\s?\d{3}\s?\d{3}) ((\+44\s?\d{3}) (\?0\d{3})\?)\s?\d{3}\s?\d{4}) ((\+44\s?\d{2}) (\?0\d{2})\?)\s?\d{4}\s?\d{4})) (\s?\#(\d{4} \d{3}))?\$');</pre> <pre>ALTER TABLE public.Customer ADD CONSTRAINT Customer_gender_check CHECK (Gender = 'M' OR Gender = 'F');</pre> <pre>ALTER TABLE public.Customer ADD CONSTRAINT Customer_email_check CHECK (Customer_Email_Address ~* '^[A-Za-z0-9._%~]+@[A-Za-z0-9.-]+[.][A-Za-z]+\$');</pre>
Membership	<pre>ALTER TABLE public.Membership ADD CONSTRAINT Membership_check CHECK (Membership_Type = 'Premium' OR Membership_Type = 'Child' OR Membership_Type = 'Elderly');</pre> <pre>ALTER TABLE public.Membership ADD CONSTRAINT Membership_join_cancel_check CHECK (Date_Joined <= Date_Cancelled);</pre> <pre>ALTER TABLE public.Membership ADD CONSTRAINT Membership_active_check CHECK (Active = 'Y' OR Active = 'N');</pre>

3 (Continued) – Indexes produced to increase the speed of queries.

Table 3 – SQL scripts to create indexes for the appropriate tables.

Entity Name	Index	Spatial / Non-Spatial
Land Parcel	<code>CREATE INDEX Land_parcel_gidx ON public.land_parcel USING GIST(location);</code>	Spatial
Facility	<code>CREATE INDEX Facility_gidx ON public.Facility USING GIST(location);</code>	Spatial
	<code>CREATE INDEX Facility_idx on public.Facility(Facility_ID);</code>	Non-Spatial
Customer	<code>CREATE INDEX Customer_idx on public.Customer(Customer_ID);</code>	Non-Spatial
Instructor	<code>CREATE INDEX Instructor_idx on public.Instructor(Instructor_ID);</code>	Non-Spatial

4 – SQL Insert Statements

Table 4 – SQL scripts to insert data into the tables

Table Name	SQL Insert Statements
Land Parcel	<pre> INSERT INTO public.Land_Parcel(Parcel_Number, Location) VALUES (001, st_geomfromtext('POLYGON((525675 184877,525694 184877,525694 184854,525675 184854,525675 184877))',27700)), (002, st_geomfromtext('POLYGON((525681 185224,525716 185243,525729 185219,525694 185200,525681 185224))',27700)), (003, st_geomfromtext('POLYGON((524764 185315,524892 185357,524907 185311,524779 185269,524764 185315))',27700)); </pre>
Facility	<pre> INSERT INTO public.Facility(Facility_Address_Line1, Facility_Address_Line2, Facility_Address_Line3, Facility_Postcode, Facility_Phone_Number, Number_of_Floors, Location, Parcel_Number) VALUES ('LDN GYM LTD', '1 Crown CL', 'London', 'NW6 1XZ', '020 7946 0119', 2, st_geomfromtext('POLYHEDRALSURFACE(((525676 184875 0,525693 184875 0,525693 184862 0,525676 184862 0,525676 184875 0)), ((525676 184875 0,525676 184875 10,525693 184875 10,525693 184875 0,525676 184875 0)), ((525676 184862 0,525676 184862 10,525693 184862 10,525693 184862 0,525676 184862 0)), ((525693 184862 0,525693 184862 10,525693 184875 10,525693 184875 0,525693 184862 0)), ((525676 184862 0,525676 184862 10,525676 184875 10,525676 184875 0,525676 184862 0)), ((525676 184875 10,525693 184875 10,525693 184862 10,525676 184862 10,525676 184875 10)))',27700), 001), </pre>

	<pre> ('LDN GYM LTD', '25 Alvanley Gardens', 'London', 'NW6 1JD', '020 7946 0403', 2, st_geomfromtext('POLYHEDRALSURFACE(((525691 185227 0,525708 185237 0,525717 185221 0,525699 185212 0,525691 185227 0)), ((525691 185227 0,525691 185227 10,525708 185237 10,525708 185237 0,525691 185227 0)), ((525699 185212 0,525699 185212 10,525717 185221 10,525717 185221 0,525699 185212 0)), ((525717 185221 0,525717 185221 10,525708 185237 10,525708 185237 0,525717 185221 0)), ((525699 185212 0,525699 185212 10,525691 185227 10,525691 185227 0,525699 185212 0)), ((525691 185227 10,525708 185237 10,525717 185221 10,525699 185212 10,525691 185227 10)))',27700), 002), ('LDN GYM LTD', 'Gondar Gardens', 'London', 'NW6 1HA', '020 7946 0133', 2, st_geomfromtext('POLYHEDRALSURFACE(((524791 185320 0,524874 185347 0,524885 185310 0,524803 185284 0,524791 185320 0)), ((524791 185320 0,524791 185320 10,524874 185347 10,524874 185347 0,524791 185320 0)), ((524803 185284 0,524803 185284 10,524885 185310 10,524885 185310 0,524803 185284 0)), ((524885 185310 0,524885 185310 10,524874 185347 10,524874 185347 0,524885 185310 0)), ((524803 185284 0,524803 185284 10,524791 185320 10,524791 185320 0,524803 185284 0)), ((524791 185320 10,524874 185347 10,524885 185310 10,524803 185284 10,524791 185320 10)))',27700), 003); </pre>
--	---

Manager	<pre> INSERT INTO public.Manager (Manager_Forename, Manager_Surname, Manager_Phone_Number, Manager_DOB, Manager_Address_Line1, Manager_Address_Line2, Manager_Address_Line3, Manager_Postcode, Manager_Email_Address, Facility_ID) VALUES ('Jack', 'Gallivan', '07700 900260', '02/05/1990', '62 Camden High St', null, 'London', 'NW1 0LT', 'jack.gallivan@gmail.com', 1), ('Alexandros', 'Petrakis', '07700 900155', '13/03/1987', '174 Royal College Street', null, 'London', 'NW1 0SP', 'alex.petrakis@gmail.com', 2), ('Jack', 'Jones', '07700 900497', '12/12/1995', 'Flat 3 Princess Park Manor', 'Royal Drive', 'London', 'N11 3FL', 'jack.jones@gmail.com', 3); </pre>
Instructor	<pre> INSERT INTO public.Instructor (Instructor_Forename, Instructor_Surname, Instructor_Phone_Number, Instructor_DOB, Instructor_Address_Line1, Instructor_Address_Line2, Instructor_Address_Line3, Instructor_Postcode, Instructor_Email_Address, Facility_ID) VALUES ('Dylan', 'Kennedy', '07700 900346', '03/05/1981', '79B Walm Lane', null, 'London', 'NW2 4QL', 'dylan.kennedy@gmail.com', 1), ('Kristina', 'Williams', '07700 900657', '20/05/1982', '130 Eversholt Street', 'Kings Cross', 'London', 'NW1 1DL', 'kristina.williams@gmail.com', 1), ('Brian', 'Roper', '07700 900176', '15/03/1992', '26C Blenheim Gardens', null, 'London', 'NW2 4NS', 'brian.roper@gmail.com', 1), ('Sam', 'Leck', '07700 900606', '17/05/1992', '38 Burtonhole Lane', null, 'London', 'NW7 1AL', 'sam.leck@gmail.com', 1), ('Gary', 'Wellington', '07700 900616', '24/05/1995', '132 Eversholt Street', 'Kings Cross', 'London', 'NW1 1DL', 'gary.wellington@gmail.com', 2), </pre>

	<pre>('Stefan', 'Jones', '07700 900469', '16/08/1985', '8 Brampton Grove', null, 'London', 'NW4 4AG', 'stfean.jones@gmail.com', 2), ('Sergei', 'Foss', '07700 900607', '20/02/1987', '145A Queens Crescent', 'Belize Park', 'London', 'NW5 4ED', 'sergei.foss@gmail.com', 2), ('Ellis', 'Knox', '07700 900965', '29/03/1990', '129 Paragon Court', 'Holders Hill Road', 'London', 'NW4 1LH', 'ellis.knox@gmail.com', 3), ('Louie', 'Donnovan', '07700 900523', '18/08/1993', '8 8hase Road', null, 'London', 'NW10 6QD', 'louie.donnovan@gmail.com', 3);</pre>
Customer	<pre>INSERT INTO public.Customer (Customer_Forename, Customer_Surname, Customer_Phone_Number, Customer_DOB, Gender, Customer_Email_Address, Facility_ID) VALUES ('Miranda', 'Hume', '07700 900448', '02/03/1952', 'F', 'mirander.hume@gmail.com', 1), ('Paige' , 'Mckee', '07700 900016', '22/09/1980', 'F', 'paige.mckee@gmail.com', 1), ('Troy', 'Bassett', '07700 900740', '24/02/2001', 'M', 'troy.bassett@gmail.com', 2), ('Sally', 'Lara', '07700 900077', '19/05/1987', 'F', 'sally.lara@gmail.com', 1), ('Brogan', 'Kumar', '07700 900443', '10/05/1992', 'M', 'brogan.kumar@gmail.com', 3), ('Cindy', 'Ellison', '07700 900910', '14/02/1994', 'F', 'cindy.ellison@gmail.com', 2), ('Neil', 'Sawyer', '07700 900177', '08/10/1948', 'M', 'neil.sawyer@gmail.com', 2), ('Viki', 'Maag', '07700 900257', '26/12/1982', 'F', 'viki.maag@gmail.com', 1), ('Cindie', 'Kaye', '07700 900271', '09/09/1987', 'F', 'cindie.kaye@gmail.com', 3), ('Jake', 'Marsden', '07700 900904', '07/09/1992', 'M', 'jake.marsden@gmail.com', 1), ('Gill', 'Dickerson', '07700 900167', '21/09/1952', 'F', 'gill.dickerson@gmail.com', 3),</pre>

	<pre>('Jordana', 'Vallee', '07700 900635', '05/03/1980', 'F', 'jordana.vallee@gmail.com', 1), ('Tom', 'Brown', '07700 900130', '30/03/1980', 'M', 'tom.brown@gmail.com', 2), ('Kirsty', 'Landes', '07700 900451', '13/04/1986', 'F', 'kirsty.landes@gmail.com', 1), ('Holly', 'Bading', '07700 900426', '30/06/2001', 'F', 'holly.bading@gmail.com', 3), ('Lewis', 'Davies', '07700 900224', '17/04/1993', 'M', 'lewis.davies@gmail.com', 2), ('Adam', 'Vaughan', '07700 900310', '05/03/2002', 'M', 'adam.vaughan@gmail.com', 2), ('Tom', 'Kenvyn', '07700 900521', '24/04/1982', 'M', 'tom.kenvyn@gmail.com', 3), ('Corey', 'Newman', '07700 900687', '26/04/2001', 'M', 'corey.newman@gmail.com', 2), ('Jack', 'Plumley', '07700 900125', '14/04/2000', 'M', 'jack.plumley@gmail.com', 2);</pre>
Membership	<pre>INSERT INTO public.Membership(Date_Joined, Activation_Date, Activation_Time, Date_Cancelled, Membership_Type, Active, Customer_ID) VALUES ('14/12/2017', '14/01/2018', '13:15:58', null, 'Elderly', 'Y', 1), ('25/01/2018', '25/01/2018', '13:21:57', null, 'Premium', 'Y', 2), ('26/01/2018', '26/02/2018', '10:44:26', null, 'Child', 'Y', 3), ('27/12/2017', '27/02/2018', '14:05:39', null, 'Premium', 'Y', 4), ('22/01/2018', '22/03/2018', '11:37:42', null, 'Premium', 'Y', 5), ('08/01/2018', '08/01/2018', '07:38:42', '07/02/2018', 'Premium', 'N', 6), ('13/12/2017', '13/01/2018', '09:22:42', '13/02/2018', 'Elderly', 'N', 7), ('20/01/2018', '20/01/2018', '15:14:10', '15/02/2018', 'Premium', 'N', 8),</pre>

	<pre> ('24/12/2017', '24/01/2018', '13:56:02', null, 'Premium', 'Y', 9), ('08/01/2018', '08/02/2018', '14:44:06', '08/03/2018', 'Premium', 'N', 10), ('19/12/2017', '19/01/2018', '10:23:29', null, 'Elderly', 'Y', 11), ('31/12/2017', '31/01/2018', '14:14:35', null, 'Premium', 'Y', 12), ('06/03/2017', '06/03/2018', '16:25:00', null, 'Premium', 'Y', 13), ('02/01/2018', '12/03/2018', '09:08:17', null, 'Premium', 'Y', 14), ('14/01/2018', '14/03/2018', '14:32:57', null, 'Child', 'Y', 15), ('05/01/2018', '05/01/2018', '11:08:46', null, 'Premium', 'Y', 16), ('10/01/2018', '15/02/2018', '16:03:36', '12/03/2018', 'Child', 'N', 17), ('02/01/2018', '18/02/2018', '13:16:07', null, 'Premium', 'Y', 18), ('24/01/2018', '24/02/2018', '11:35:08', null, 'Child', 'Y', 19), ('03/01/2018', '28/03/2018', '10:58:02', null, 'Child', 'Y', 20); </pre>
Classes	<pre> INSERT INTO public.Classes (Class_Date, Class_Time, Class_Type, Instructor_ID, Customer_ID) VALUES ('11/01/2018', '16:00:00', 'Power Lifting', 1, 1), ('30/01/2018', '10:30:00', 'HIIT Cardio', 7, 3), ('03/03/2018', '17:45:00', 'Olympic Lifting', 2, 4), ('06/03/2018', '09:00:00', 'HIIT Cardio', 9, 5), ('29/03/2018', '06:30:00', 'Hypertrophy', 1, 8), ('11/01/2018', '11:30:00', 'Hypertrophy', 4, 10), ('02/02/2018', '17:30:00', 'Power Lifting', 8, 11), ('04/03/2018', '15:30:00', 'Hypertrophy', 5, 13), ('05/03/2018', '10:30:00', 'Power Lifting', 3, 12), ('10/03/2018', '12:15:00', 'HIIT Cardio', 8, 15), ('09/01/2018', '08:30:00', 'Olympic Lifting', 5, 19), </pre>

	<pre>('25/01/2018', '16:00:00', 'Hypertrophy', 5, 17), ('01/03/2018', '14:30:00', 'HIIT Cardio', 6, 3), ('07/03/2018', '14:00:00', 'Olympic Lifting', 6, 3), ('28/03/2018', '15:45:00', 'HIIT Cardio', 4, 2);</pre>
Equipment	<pre>INSERT INTO public.Equipment (Equipment_SN, Equipment_Category, Equipment_Name, Customer_ID) VALUES (4806, 'Safety', 'Weight Lifting Belt', 1), (9838, 'Mobility', 'Lacrosse Ball', 4), (7393, 'Comfort', 'Yoga Matt', 15), (3837, 'Comfort', 'Barbell Pad', 19);</pre>

5 – QGIS Map of Spatial Data

Asset Locations for LDN GYM Ltd.

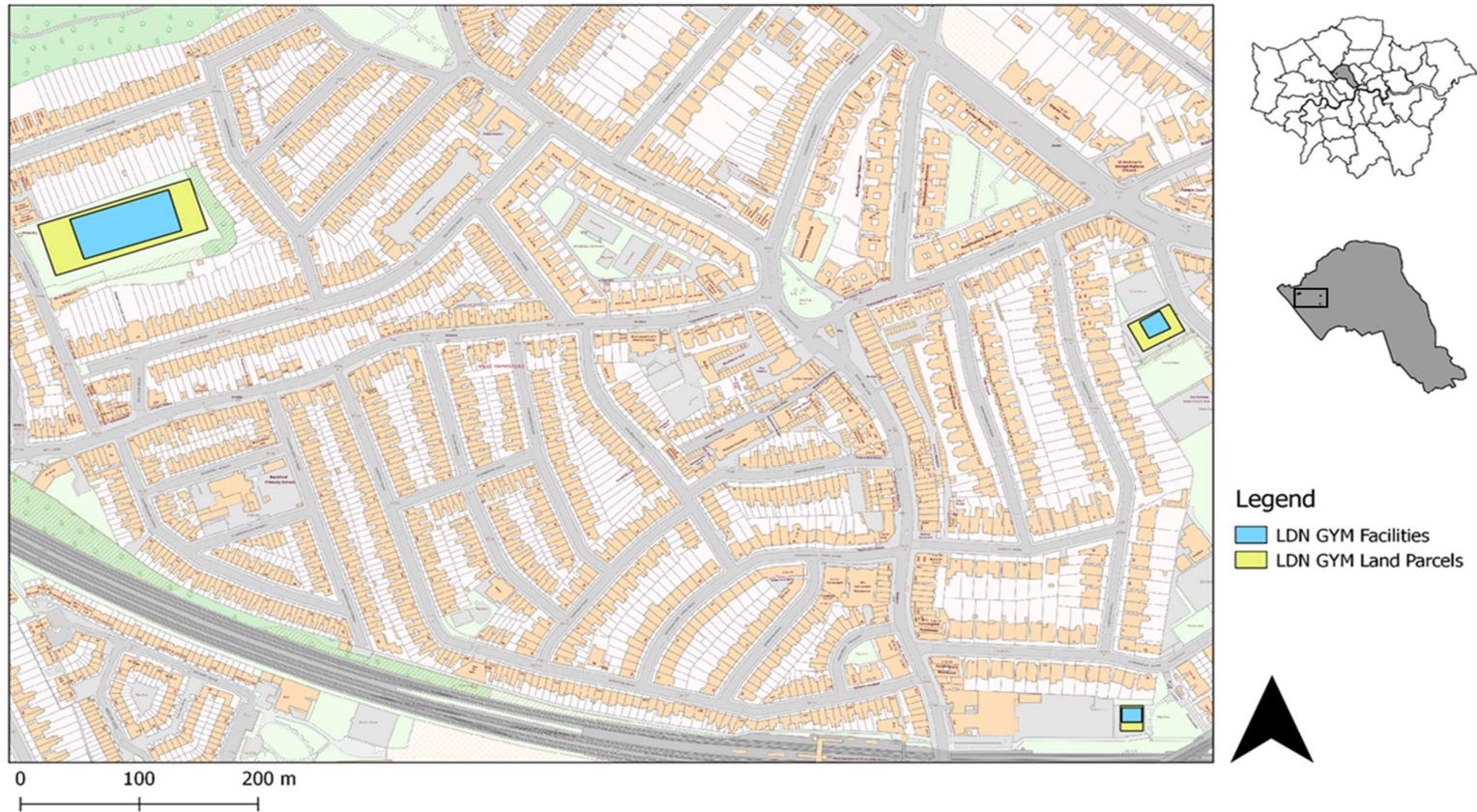


Figure 3 – Map of LDN GYM Ltd. Land Parcels and Facilities created in QGIS utilising a connection to pgAdmin4 user12 database to obtain the relevant spatial data created previously in this report. Contains OS data © Crown copyright and database right (2018). Digimap Licence.

6 – FME Screenshot of 3D Data

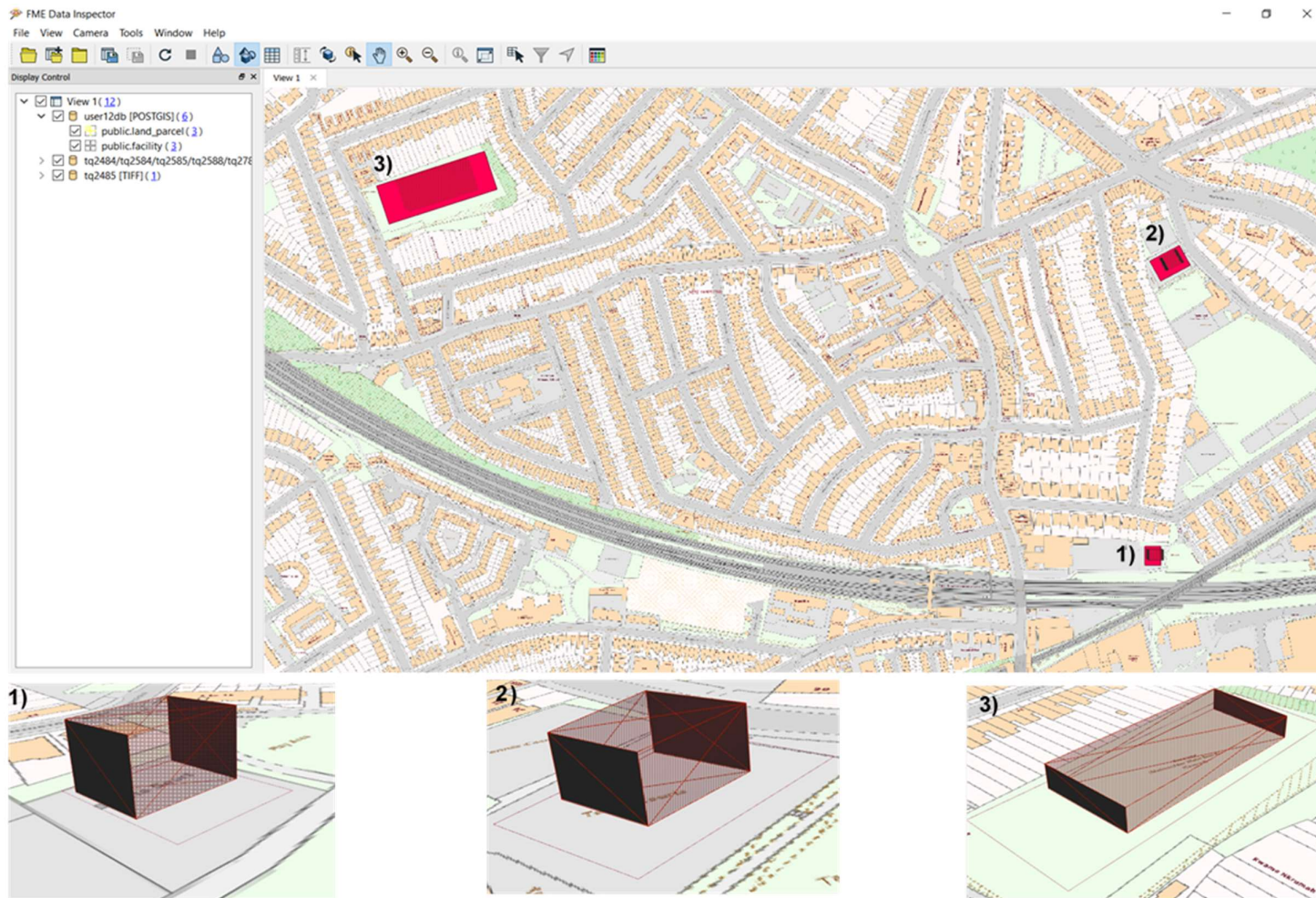
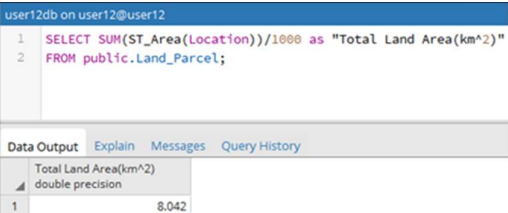
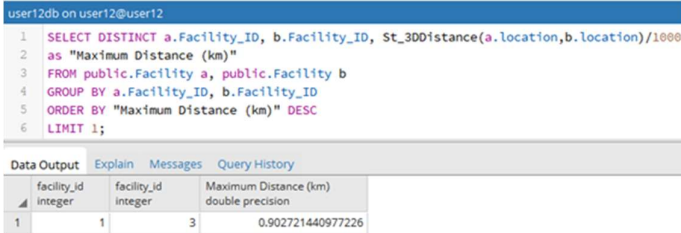


Figure 4 -FME screenshots of 3D Facilities contained within their respective 2D land parcels.
Contains OS data © Crown copyright and database right (2018). Digimap Licence.

7 – List of functional requirements from Assignment 1

Table 5 – Functional requirements from assignment 1 presented with accompanying SQL script to answer the respective requirements. Screenshots of the query being executed in PG Admin 4 also presented.

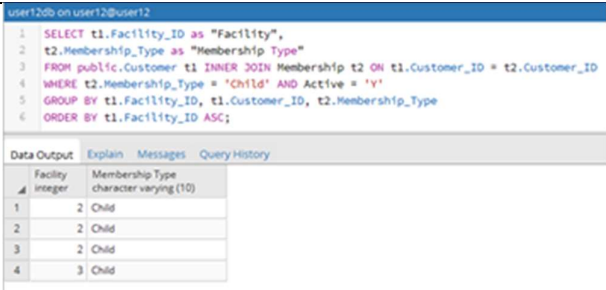
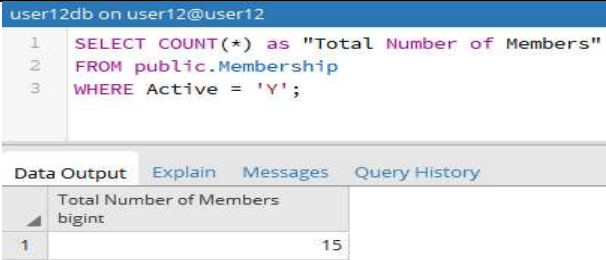
Requirement #	Requirement Description	List of Table(s) Involved	Spatial	Join	SQL Query	Screenshot of results from PG Admin
1	What is the total area of land owned by the business?	Land Parcel	Yes	No	<pre>SELECT SUM(ST_Area(Location)) /1000 as "Total Land Area(km^2)" FROM public.Land_Parcel;</pre>	
2	Which two gyms are the furthest distance away from each other?	Facility	Yes	No	<pre>SELECT DISTINCT a.Facility_ID, b.Facility_ID, St_3DDistance(a.location, b.location)/1000 as "Maximum Distance (km)" FROM public.Facility a, public.Facility b GROUP BY a.Facility_ID, b.Facility_ID ORDER BY "Maximum Distance (km)" DESC LIMIT 1;</pre>	

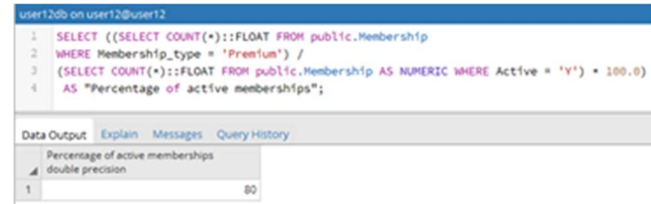
3	How many gyms fall within an area of interest (for example, a London Borough)?	Facility Land Parcel	Yes	No	<pre>/* In this example, the 'area of interest' being used is the London ward of West Hampstead. The query utilised the land Parcel table, rather than the expected Facility table. This is due to ST_Within not supporting 3D geometries. However, each land parcel contains a gym. Therefore, if a land parcel is contained within an area, it guarantees a gym facility will be. */ SELECT Parcel_Number as "Parcel Number", (ST_Within(ST_CENTROID(Loc ation), ST_GEOMFROMTEXT('Polygon((524651 184648,524926 184771,525058 184775,525067 184869,525034 184881,524958 185167, 525084 185214,525370 185249,525416 185225,525673 185339,525857 185274,526170 184976,526328 184681,526105 184735,</pre>	<pre>user12db on user12@user12 1 SELECT Parcel_Number as "Parcel Number", (ST_Within(ST_CENTROID(Location), 2 ST_GEOMFROMTEXT('Polygon((524651 184648,524926 184771,525058 184775,525067 184869,525034 184881,524958 185167, 3 525084 185214,525370 185249,525416 185225,525673 185339,525857 185274,526170 184976,526328 184681,526105 184735, 4 525089 184665,525685 184622,525636 184616,525630 184339,525653 184337,525628 184303,525475 184301,525456 184364, 5 525425 184363,525438 184406,525389 184398,525387 184363,525161 184376,525166 184407,525114 184446,525137 184545, 6 525066 184548,525058 184589,524780 184463,524651 184648)))', 27700))) 7 AS "Gyms In West Hampstead" FROM public.Land_Parcel;</pre> <table><tr><th>Parcel Number</th><th>Gyms in West Hampstead</th></tr><tr><td>1</td><td>true</td></tr><tr><td>2</td><td>true</td></tr><tr><td>3</td><td>false</td></tr></table>	Parcel Number	Gyms in West Hampstead	1	true	2	true	3	false
Parcel Number	Gyms in West Hampstead													
1	true													
2	true													
3	false													

					<pre>525689 184665,525685 184622,525636 184616,525630 184359,525653 184337,525628 184303,525475 184301,525456 184364, 525425 184363,525438 184406,525389 184398,525387 184363,525161 184376,525166 184407,525114 184446,525137 184545, 525066 184548,525058 184589,524780 184463,524651 184648))', 27700))) AS "Gyms in West Hampstead" FROM public.Land_Parcel;</pre>																										
4	What is the proximity of each gym to a proposed gym location defined by GPS coordinates?	Facility	Yes	No	<pre>SELECT Facility_ID, (ST_DISTANCE(Location, ST_GEOFROMTEXT('POINT(526 477 186025)', 27700))/1000) as "Distance(km) to proposed gym location" FROM public.Facility;</pre>	<div>user12db on user12@user12</div> <div><pre>1 SELECT Facility_ID, (ST_DISTANCE(Location, ST_GEOFROMTEXT('POINT(526477 186025)', 27700)))/1000) 2 as "Distance(km) to proposed gym location" FROM public.Facility; 3</pre></div> <div><table><thead><tr><th colspan="2">Data Output</th><th>Explain</th><th>Messages</th><th>Query History</th></tr></thead><tbody><tr><td>facility_id</td><td>integer</td><td>Distance(km) to proposed gym location</td><td>double precision</td><td></td></tr><tr><td>1</td><td>1</td><td>1.39181751677438</td><td></td><td></td></tr><tr><td>2</td><td>2</td><td>1.10104722877813</td><td></td><td></td></tr><tr><td>3</td><td>3</td><td>1.74048642626135</td><td></td><td></td></tr></tbody></table></div>	Data Output		Explain	Messages	Query History	facility_id	integer	Distance(km) to proposed gym location	double precision		1	1	1.39181751677438			2	2	1.10104722877813			3	3	1.74048642626135		
Data Output		Explain	Messages	Query History																											
facility_id	integer	Distance(km) to proposed gym location	double precision																												
1	1	1.39181751677438																													
2	2	1.10104722877813																													
3	3	1.74048642626135																													

5	What is the perimeter of a given land parcel owned by the business?	Land Parcel	Yes	No	<pre>--Parcel Number 003 chosen for this example SELECT ROUND(CAST(SUM(ST_Perimeter(Location)) AS NUMERIC),2) as "Total Perimeter(m)" FROM public.Land_Parcel WHERE Parcel_Number = 003;</pre>	<div>user12db on user12@user12</div> <pre>1 SELECT ROUND(CAST(SUM(ST_Perimeter(Location)) as numeric),2) 2 as "Total Perimeter(m)" 3 FROM public.Land_Parcel WHERE Parcel_Number = 003;</pre> <div>Data Output Explain Messages Query History</div> <table><thead><tr><th></th><th>Total Perimeter(m)</th></tr></thead><tbody><tr><td>1</td><td>366.20</td></tr></tbody></table>		Total Perimeter(m)	1	366.20		
	Total Perimeter(m)											
1	366.20											
6	Which member participates in the most classes during a given date range?	Customer Membership, Class	No	Yes	<pre>-- 01/03/2018 - 10/03/2018 used for this example. SELECT t1.Customer_ID, COUNT(*) AS "Number of classes taken" FROM public.Classes t1 INNER JOIN public.Membership t2 ON t1.Customer_ID = t2.Customer_ID WHERE class_date >= '01/03/2018' AND class_date <= '10/03/2018' AND Active = 'Y' GROUP BY t1.Customer_id ORDER BY COUNT(*) DESC LIMIT 1;</pre>	<div>user12db on user12@user12</div> <pre>1 SELECT t1.Customer_ID, COUNT(*) AS "Number of classes taken" 2 FROM public.Classes t1 INNER JOIN public.Membership t2 ON t1.Customer_ID = t2.Customer_ID 3 WHERE class_date >= '01/03/2018' AND class_date <= '10/03/2018' AND Active = 'Y' 4 GROUP BY t1.Customer_id 5 ORDER BY COUNT(*) DESC 6 LIMIT 1;</pre> <div>Data Output Explain Messages Query History</div> <table><thead><tr><th></th><th>customer_id</th><th>Number of classes taken</th></tr></thead><tbody><tr><td>1</td><td>3</td><td>2</td></tr></tbody></table>		customer_id	Number of classes taken	1	3	2
	customer_id	Number of classes taken										
1	3	2										

7	Which gym facility employs the most instructors?	Facility, Instructor	No	Yes	<pre>/*N.B: By utilising foreign keys, a JOIN is not required. The SQL and result of meeting this requirement both ways is presented below */ --Using a join SELECT t1.Facility_ID AS Facility, COUNT(t2.Instructor_ID) AS "Number of instructors" FROM public.Facility t1 INNER JOIN Instructor t2 ON t1.Facility_ID = t2.Facility_ID GROUP BY t1.facility_Id ORDER BY "Number of instructors" DESC LIMIT 1; --Not using a join SELECT Facility_ID as Facility, COUNT(*) as "Number of instructors" FROM public.Instructor GROUP BY Facility_ID ORDER BY COUNT(*) DESC LIMIT 1;</pre>	<div>user12db on user12@user12</div> <pre>1 --Using a join 2 SELECT t1.Facility_ID AS Facility, COUNT(t2.Instructor_ID) AS "Number of instructors" 3 FROM public.Facility t1 INNER JOIN Instructor t2 ON t1.Facility_ID = t2.Facility_ID 4 GROUP BY t1.facility_Id 5 ORDER BY "Number of instructors" DESC 6 LIMIT 1;</pre> <table><tr><th colspan="3">Data Output</th><th>Explain</th><th>Messages</th><th>Query History</th></tr><tr><th></th><th>facility integer</th><th>Number of instructors bigint</th><th></th><th></th><th></th></tr><tr><td>1</td><td>1</td><td>4</td><td></td><td></td><td></td></tr></table> <div>user12db on user12@user12</div> <pre>1 --Not using a join 2 SELECT Facility_ID as Facility, COUNT(*) 3 as "Number of instructors" FROM public.Instructor 4 GROUP BY Facility_ID 5 ORDER BY COUNT(*) DESC 6 LIMIT 1;</pre> <table><tr><th colspan="3">Data Output</th><th>Explain</th><th>Messages</th><th>Query History</th></tr><tr><th></th><th>facility integer</th><th>Number of instructors bigint</th><th></th><th></th><th></th></tr><tr><td>1</td><td>1</td><td>4</td><td></td><td></td><td></td></tr></table>	Data Output			Explain	Messages	Query History		facility integer	Number of instructors bigint				1	1	4				Data Output			Explain	Messages	Query History		facility integer	Number of instructors bigint				1	1	4			
Data Output			Explain	Messages	Query History																																					
	facility integer	Number of instructors bigint																																								
1	1	4																																								
Data Output			Explain	Messages	Query History																																					
	facility integer	Number of instructors bigint																																								
1	1	4																																								

8	Which gym facility has the most child memberships?	Facility, Customer, Membership	No	Yes	<pre>SELECT t1.Facility_ID as "Facility", t2.Membership_Type as "Membership Type" FROM public.Customer t1 INNER JOIN Membership t2 ON t1.Customer_ID = t2.Customer_ID WHERE t2.Membership_Type = 'Child' AND Active = 'Y' GROUP BY t1.Facility_ID, t1.Customer_ID, t2.Membership_Type ORDER BY t1.Facility_ID ASC;</pre>	
9	What is the total number of gym members across the whole business?	Customer Membership	No	No	<pre>SELECT COUNT(*) as "Total Number of Members" FROM public.Membership WHERE Active = 'Y';</pre>	

10	What percentage of active memberships are classified as premium memberships?	Membership	No	No	<pre>SELECT ((SELECT COUNT(*)::FLOAT FROM public.Membership WHERE Membership_type = 'Premium') / (SELECT COUNT(*)::FLOAT FROM public.Membership AS NUMERIC WHERE Active = 'Y') * 100.0) AS "Percentage of active memberships";</pre>	
Totals	10	N/A	5	3	N/A	N/A

Part B – Option 1 – 3D GIS Support in GeoMedia 3D

In 2010, Intergraph, an American software development company released GeoMedia 3D as an extension to their existing Geographic information System (Intergraph, 2010). In the same year, Intergraph were formally acquired by Hexagon AB, resulting in the creation of the Hexagon Geospatial division, formed from Intergraph's existing geospatial portfolio in 2014 (Hexagon, 2014).

As an extension, the GeoMedia 3D map window seamlessly integrates into the existing GI framework. The inclusion of the third dimension is an advantage over traditional, 2D GI systems, as it enables a more realistic visualisation of the data. (Hexagon Geospatial, 2016).

GeoMedia 3D is commonly utilised in the construction of simple city models as it can be used to create Level of Detail (LOD) 1 building representations, existing as simple, 'block' style buildings, which do not consider roof structures etc. (Hexagon Geospatial, 2016; Ellul and Altenbuchner, 2014). To increase the realism of each building, 'textures', which reference PNG image files can be added (figure 5a). Furthermore, due to attribute based, 3D symbology, manipulation is not limited to the vertical extrusion of points, lines and polygons based on height, but can be manipulated based on any relevant attribute (Gammon, 2014). It is important to note that this can be achieved in most 2D GIS packages. However, 3D symbology combined with appropriate styling allows for a more imposing visualisation (figure 5b).



Figure 5 – Attribute based, 3D symbology in GeoMedia 3D.

Figure 5a (left) - Texture files are combined with LOD1 building representations.

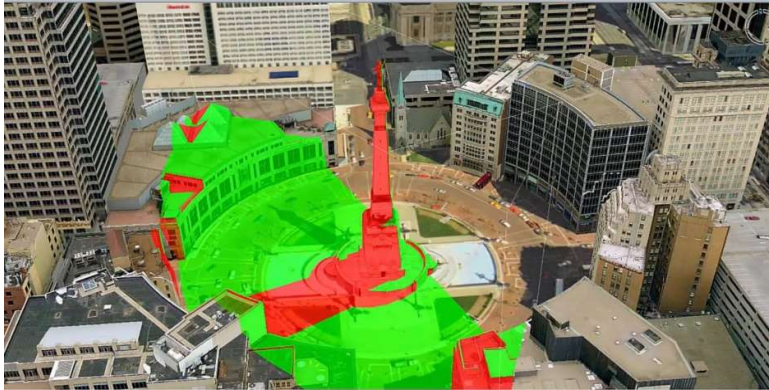
Figure 5b (right) - The size of each fire hydrant is based upon the pressure attribute.

Source: Gammon (2014).

Particularly relevant when producing city models is GeoMedia 3D's ability to conduct 'fly-throughs', which is effectively a simulated tour of an area. City model fly-through's combined with environmental effects allows for an increasingly realistic emulation of the real world, incomparable to a 2D visualisation. The illumination commands within GeoMedia 3D can be used to vary the sun's presence, time of day or year and cast the resultant shadows. Additional environmental effects include rain, snow and fog. (Hexagon Geospatial, 2015).

Photo-realistic 3D terrain can be produced in GeoMedia 3D by draping aerial or satellite imagery over a 3D surface (Bektaş, Çöltekin, and Straumann, 2012). Vector data can also be visualised in this way, but it should be noted that as only one Z value can be associated with any XY coordinates, both the extrusion from footprints and image draping are technically 2.5D and not true 3D (GeoMedia Support User, 2016).

Viewshed analysis conducted in 3D is much more informative than its' 2D counterpart. In GeoMedia, the visibility (or lack of) from 'the eye' can be easily deciphered and saved as an independent GIS layer (figure 6). Real world applications of this vary from assessing the view from a proposed hotel to the placement of security cameras. (Hexagon Geospatial, 2018). Conducting a viewshed analysis in a 2D GIS package has very limited practical application due to the lack of consideration for height (z) and associated poor accuracy of the result. This is because changes in elevation drastically alter line of sight vision.



*Figure 6: Example of viewshed analysis in GeoMedia 3D.
Source: Kita (2015).*

Due to 3D data complexity, storage and computational power requirements vastly increase in comparison with analysing and visualising 2D data. The creation of a 3D Mesh Layer (3DML) in GeoMedia 3D combats this, allowing large 3D datasets such as city models to be visualised and navigated. 3DML files are optimised for display, removing the lag in rendering commonly associated with complex 3D data files in the GUI. Furthermore, this representation of the model is disconnected from the feature class and therefore edits are not linked. (Hexagon Geospatial, 2014).

GeoMedia 3D users can import existing 3D data from other software packages including Trimble SketchUp, CityGML and GoogleEarth (Hexagon Geospatial, 2016). As GeoMedia 3D exists as an add-on to the GeoMedia 2D Desktop package, the user has complete accessibility to any existing connections with the Open Geospatial Consortium (OGC) and associated Web Map Service (WMS) and Web Feature Service (WFS) (Intergraph, 2010).

Much of this research paper has focussed on the use of GeoMedia 3D in respect to an urban landscape, with references to city models etc. However, GeoMedia 3D is not designed to create complex 3D objects, but rather to enable 3D connectivity to its users GIS data (Gammon, 2014). However, as outlined above, more complex models can be imported from a variety of other sources. (Intergraph, 2010).

In comparison to 2D GIS, GeoMedia 3D offers increased visualisation and analysis techniques which are much more applicable in industry (e.g. urban planning utilising 3D viewshed analysis). However, 2D GIS benefits from less complex data than 3D GIS. In the future, technological advancements increasing the power and storage capabilities of computers may negate this issue.

References

- Bektaş, K., Çöltekin, A. and Straumann, R. (2012). Survey of True 3D and Raster Level of Detail Support in GIS Software. [online] Available at: <http://www.geo.uzh.ch/~arzu/publications/bektas-et-al-2012.pdf> [Accessed 22 Mar. 2018].
- Ellul, C. and Altenbuchner, J. (2014). Investigating approaches to improving rendering performance of 3D city models on mobile devices. *Geo-spatial Information Science*, 17(2), pp.73-84.
- Gammon, R. (2014). *Getting to 3D: Part 1*. [online] Sensing Change Blog. Available at: <https://blog.hexagongeospatial.com/getting-geomedia-3d-part-1/> [Accessed 10 Apr. 2018].
- GeoMedia Support User (2016). *Spatial Intersection with shape files - z-coordinate not transferred*. [online] Community.hexagongeospatial.com. Available at: <https://community.hexagongeospatial.com/t5/Support-GeoMedia/Spatial-Intersection-with-shape-files-z-coordinate-not/td-p/21730> [Accessed 21 Mar. 2018].
- Hexagon (2014). *Press Releases | Hexagon*. [online] Web.archive.org. Available at: https://web.archive.org/web/20140119045414/http://investors.hexagon.com/en/news-releases?afw_id=1300899 [Accessed 21 Mar. 2018].
- Hexagon Geospatial (2014). *GeoMedia 3D - Working with 3D Mesh Layer (3DML) Files*. [online] P.widencdn.net. Available at: <https://p.widencdn.net/eufuer> [Accessed 22 Mar. 2018].
- Hexagon Geospatial (2015). *Introduction To GeoMedia 3D: Environmental Effects eTraining*. [online] Community.hexagongeospatial.com. Available at: <https://community.hexagongeospatial.com/hmrkh95973/attachments/hmrkh95973/eTGeoMedia/18/1/Intro%20to%20GeoMedia%203D%20-%20Environmental%20Effects%20Script.pdf> [Accessed 22 Mar. 2018].
- Hexagon Geospatial (2016). *GeoMedia 3D Product Sheet*. [online] Hexagongeospatial.com. Available at: <https://www.hexagongeospatial.com/brochure-pages/geomedia-3d-product-sheet> [Accessed 22 Mar. 2018].
- Hexagon Geospatial (2018). *GeoMedia 3D help*. [online] Hexagongeospatial.fluidtopics.net. Available at: https://hexagongeospatial.fluidtopics.net/reader/o7AqnJn1shAquTj9Yet2Lw/M3qfUX2MT4Jq0spsN65h_Q [Accessed 22 Mar. 2018].
- Intergraph. (2010). *Intergraph® Introduces 3D Capabilities in Geospatial Software*. [online] Available at: <http://www.intergraph.com/assets/pressreleases/2010/09-02-2010.aspx> [Accessed 8 Apr. 2018].
- Kita, D. (2015). *Using GeoMedia 3D to Identify Best Security Camera Placement*. [online] Hexagon Geospatial. Available at: <https://blog.hexagongeospatial.com/using-geomedia-3d-to-identify-best-security-camera-placement/>.
- Stack Overflow User (2011). *How can I create a constraint to check if an email is valid in postgres?*. [online] Stackoverflow.com. Available at: <https://stackoverflow.com/questions/5689718/how-can-i-create-a-constraint-to-check-if-an-email-is-valid-in-postgres> [Accessed 10 Apr. 2018].

Stack Overflow User (2016). *Regular expression for UK based and only numeric phone number in cakephp*. [online] Stackoverflow.com. Available at: <https://stackoverflow.com/questions/11518035/regular-expression-for-uk-based-and-only-numeric-phone-number-in-cakephp> [Accessed 10 Apr. 2018].

Wright, G. (2008). *Regular Expression Library*. [online] Regexlib.com. Available at: [http://regexlib.com/\(X\(1\)A\(Siy_IdgdPq0hBaO9CEsXSLbav4X_64NmcxKx7OLsie2TgHYYKvGG6oe0XhXmjaYEAV3VCvcp0Z8DstKYWDrXgbl_WJnP4QNjxKv0ywxtMupWu_0-UuOSvMklhlmMOKRLGiyRtiExcYypD7--vna8-s8eVg9GxvAHn0NB_y1RF-JyakBft4HVpuJdTyrqa2m0\)\)/REDetails.aspx?regex_id=2113](http://regexlib.com/(X(1)A(Siy_IdgdPq0hBaO9CEsXSLbav4X_64NmcxKx7OLsie2TgHYYKvGG6oe0XhXmjaYEAV3VCvcp0Z8DstKYWDrXgbl_WJnP4QNjxKv0ywxtMupWu_0-UuOSvMklhlmMOKRLGiyRtiExcYypD7--vna8-s8eVg9GxvAHn0NB_y1RF-JyakBft4HVpuJdTyrqa2m0))/REDetails.aspx?regex_id=2113) [Accessed 10 Apr. 2018].

Appendix

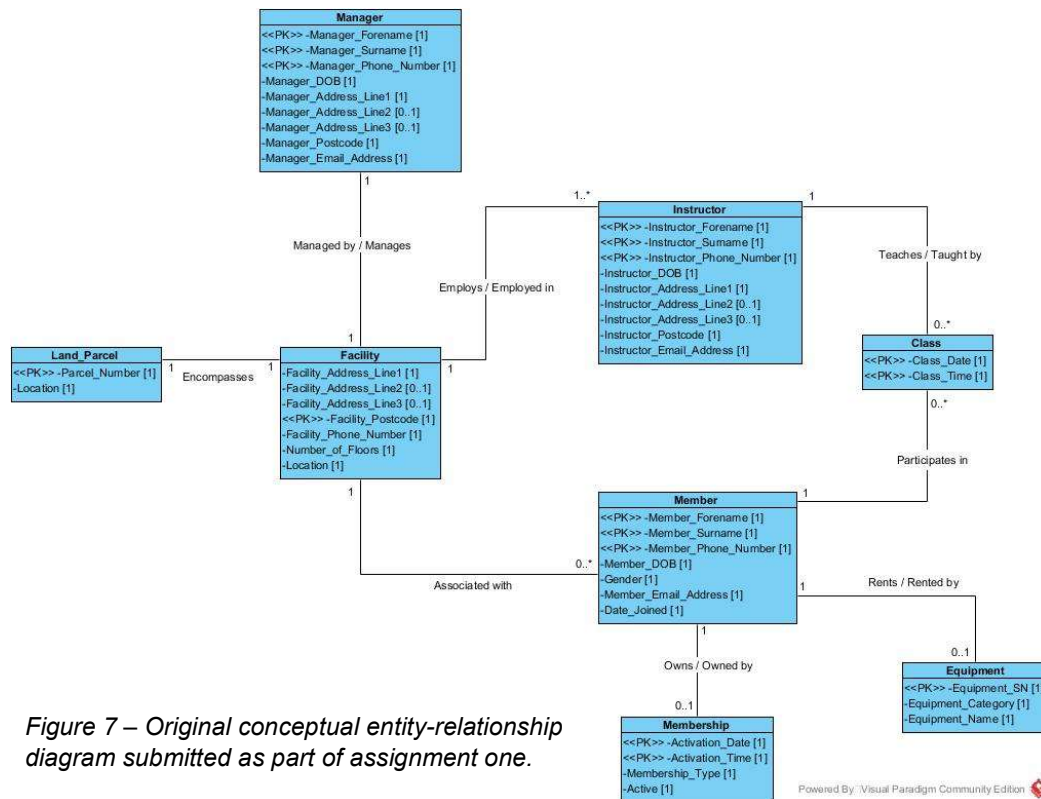


Figure 7 – Original conceptual entity-relationship diagram submitted as part of assignment one.

Figure 1 presents a revised conceptual ERD which differs slightly from the original conceptual ERD submitted in assignment one (Figure 7). The changes are based upon feedback from assignment one and include:

- 'Member' entity was re-named to 'Customer'.
- The 'Date Joined' attribute was removed from the (now) 'Customer' entity.
- 'Date Joined' and 'Date Cancelled' attributes added to the 'Membership' entity.
- 'Class' entity renamed to 'Classes' to avoid complications with reserved words in pgAdmin.²
- 'Class Type' attribute added to the (now) 'Classes' entity.

Please note the following:

The feedback from assignment one inferred to include details of the physical space in which the one-to-one class will be taking place (e.g. an additional attribute to the (now) 'Classes' entity). However, the author did not deem this necessary. This is because each one-to-one class is taking place at a specific gym which can be determined from the Foreign Keys of the Instructor and/or Customer. The class will be mobile, moving throughout the regular gym space as necessary and not contained in a specific area or room. The instructor will meet the customer upon arrival and the class will begin.

² This specific revision was not completed because of feedback from assignment one, but rather an issue that became apparent once writing SQL scripts.