# CW1

219031729

07/03/2022

# Part 2

```
# Load the RPostgreSQL library
library(RPostgreSQL)
```

```
## Loading required package: DBI
```

```
pgsql_drv <- dbDriver("PostgreSQL")

# Connection information
pgsql_user <- "kal41"
pgsql_password <- "219031729"
pgsql_dbname <- "sds27"
pgsql_host <- "pgsql.mcs.le.ac.uk"
pgsql_port <- 5432

# Create the connection
pgsql_conn <- dbConnect(
  pgsql_drv,
  host = pgsql_host, port = pgsql_port,
  user = pgsql_user,
  password = pgsql_password,
  dbname = pgsql_dbname
)

# Remove the connection information
# from the R environment
rm(pgsql_user)
rm(pgsql_password)
rm(pgsql_dbname)
rm(pgsql_host)
rm(pgsql_port)
```

```
#Checking data types for each of the variables in table
#greater_london_osm_point
dbGetQuery(
  conn = pgsql_conn,
  statement = "SELECT column_name, data_type
FROM information_schema.columns
WHERE table_name = 'greater_london_osm_point' ;"
)
```

```
##    column_name         data_type
## 1           id           integer
## 2         geom      USER-DEFINED
## 3       osm_id character varying
## 4         name character varying
## 5      barrier character varying
## 6      highway character varying
## 7          ref character varying
## 8      address character varying
## 9        is_in character varying
## 10       place character varying
## 11    man_made character varying
## 12  other_tags character varying
```

```
#Bicycle parking points in the study area: Havering
Cycle_P_in_Haerving <- dbGetQuery(
  conn = pgsql_conn,
  statement = "SELECT glop.id, glop.osm_id, glop.other_tags,
  gll.oa_code,  gll.lad11cd
FROM greater_london_osm_point glop
INNER JOIN greater_london_loac gll
ON st_intersects(
st_transform(glop.geom, 27700),
st_transform(gll.geom, 27700)
)
WHERE glop.other_tags like '%bicycle_parking%' AND
gll.lad11nm = 'Havering';"
)

Cycle_P_in_Haerving
```

```
##        id     osm_id
## 1   14299  274870159
## 2  190880 3104472651
## 3  190894 3104472670
## 4  194291 3221558719
## 5  194420 3225682668
## 6  195742 3244383768
## 7  212032 3749182495
## 8  224232 4081920759
## 9  155016 2320751625
## 10  73876 1344561848
##
other_tags
## 1
"amenity"=>"bicycle_parking"
## 2
"amenity"=>"bicycle_parking"
## 3
"amenity"=>"bicycle_parking"
## 4                                                    "amenity"=>"bicycle_parking","c
apacity"=>"6","bicycle_parking"=>"stands"
## 5                                                                  "amenity"=>"bic
```

```
ycle_parking","bicycle_parking"=>"stands"
## 6  "access"=>"yes","amenity"=>"bicycle_parking","bicycle_parking"=>"stands","ca
pacity"=>"50","covered"=>"no","fee"=>"no"
## 7
"amenity"=>"bicycle_parking"
## 8
"amenity"=>"bicycle_parking"
## 9                        "amenity"=>"bicycle_parking","covered"=>"yes","capacit
y"=>"24","bicycle_parking"=>"high_density"
## 10                             "amenity"=>"bicycle_parking","covered"=>"yes","ca
pacity"=>"14","bicycle_parking"=>"stands"
##      oa_code   lad11cd
## 1  E00011849 E09000016
## 2  E00011452 E09000016
## 3  E00011481 E09000016
## 4  E00011914 E09000016
## 5  E00011914 E09000016
## 6  E00011935 E09000016
## 7  E00011659 E09000016
## 8  E00011935 E09000016
## 9  E00011935 E09000016
## 10 E00011935 E09000016
```

```
#Bicycle parking points in the study area: Havering
#Converted to human readable coordinate format
EWKT_Cycle_P_in_Haerving <- dbGetQuery(
  conn = pgsql_conn,
  statement = "SELECT  glop.other_tags, gll.lad11cd, gll.oa_code, gll.lad11nm,
ST_AsEWKT(glop.geom) geom_as_wkt
FROM greater_london_osm_point glop
INNER JOIN greater_london_loac gll
ON st_intersects(
  st_transform(glop.geom, 27700),
  st_transform(gll.geom, 27700)
)
WHERE glop.other_tags like '%bicycle_parking%' AND
gll.lad11nm = 'Havering';"
)

EWKT_Cycle_P_in_Haerving %>% select (lad11cd, oa_code, lad11nm,
                                     geom_as_wkt, other_tags)
```

```
##      lad11cd   oa_code  lad11nm                            geom_as_wkt
## 1  E09000016 E00011849 Havering SRID=4326;POINT(0.1925964 51.5257979)
## 2  E09000016 E00011452 Havering  SRID=4326;POINT(0.252502 51.5810805)
## 3  E09000016 E00011481 Havering SRID=4326;POINT(0.2356086 51.5944099)
## 4  E09000016 E00011914 Havering  SRID=4326;POINT(0.248445 51.5545826)
## 5  E09000016 E00011914 Havering  SRID=4326;POINT(0.2474866 51.553287)
## 6  E09000016 E00011935 Havering SRID=4326;POINT(0.2502582 51.5584349)
## 7  E09000016 E00011659 Havering SRID=4326;POINT(0.1912415 51.5898794)
## 8  E09000016 E00011935 Havering SRID=4326;POINT(0.2512451 51.5587753)
## 9  E09000016 E00011935 Havering SRID=4326;POINT(0.2520801 51.5585929)
## 10 E09000016 E00011935 Havering SRID=4326;POINT(0.2518227 51.5587505)
```

```
##
other_tags
## 1
"amenity"=>"bicycle_parking"
## 2
"amenity"=>"bicycle_parking"
## 3
"amenity"=>"bicycle_parking"
## 4                                                    "amenity"=>"bicycle_parking","c
apacity"=>"6","bicycle_parking"=>"stands"
## 5                                                                  "amenity"=>"bic
ycle_parking","bicycle_parking"=>"stands"
## 6  "access"=>"yes","amenity"=>"bicycle_parking","bicycle_parking"=>"stands","ca
pacity"=>"50","covered"=>"no","fee"=>"no"
## 7
"amenity"=>"bicycle_parking"
## 8
"amenity"=>"bicycle_parking"
## 9                        "amenity"=>"bicycle_parking","covered"=>"yes","capacit
y"=>"24","bicycle_parking"=>"high_density"
## 10                             "amenity"=>"bicycle_parking","covered"=>"yes","ca
pacity"=>"14","bicycle_parking"=>"stands"
```

```
#checking if there is any duplicate point
#Based on id, there are unique bicycle parking
IDGrouped_Haerving_Cycle_P <- dbGetQuery(
  conn = pgsql_conn,
  statement = "SELECT glop.id, glop.geom, count(*)
FROM greater_london_osm_point glop
INNER JOIN greater_london_loac gll
ON st_intersects(
  st_transform(glop.geom, 27700),
  st_transform(gll.geom, 27700)
)
WHERE glop.other_tags like '%bicycle_parking%' AND
gll.lad11nm = 'Havering'
GROUP BY glop.id;"
) %>% knitr::kable()
```

```
## Warning in postgresqlExecStatement(conn, statement, ...): RS-DBI driver warnin
g:
## ((null))
```

```
IDGrouped_Haerving_Cycle_P
```

| id | geom | count |
|---|---|---|
| 14299 | 0101000020E610000018E6A9B3FFA6C83F1A6778584DC34940 | 1 |
| 73876 | 0101000020E6100000A1FC38F5DC1DD03FD40FEA2285C74940 | 1 |
| 155016 | 0101000020E6100000F53B5E921422D03F8FA3DEF87FC74940 | 1 |
| 190880 | 0101000020E6100000842C0B26FE28D03FF3EB87D860CA4940 | 1 |

| | | |
|---|---|---|
| 190894 | 0101000020E61000002203D42F6C28CE3F9675A49F15CC4940 | 1 |
| 194291 | 0101000020E61000007767EDB60BCDCF3F20F70890FCC64940 | 1 |
| 194420 | 0101000020E6100000B85F9912A4ADCF3FA726C11BD2C64940 | 1 |
| 195742 | 0101000020E6100000AC9223F83A04D03FF73878CB7AC74940 | 1 |
| 212032 | 0101000020E610000076A911FA997AC83FC4CA0D2B81CB4940 | 1 |
| 224232 | 0101000020E61000009CEFF1536614D03F03A8F3F285C74940 | 1 |

```
#checking if there is any duplicate point
#Based on OA and Supgroup_CD, there are unique bicycle parking
OA_Grouped_Haerving_Cycle_P <- dbGetQuery(
  conn = pgsql_conn,
  statement = "SELECT gll.oa_code, gll.supgrp_cd, count(*)
FROM greater_london_osm_point glop
INNER JOIN greater_london_loac gll
ON st_intersects(
st_transform(glop.geom, 27700),
st_transform(gll.geom, 27700)
)
WHERE glop.other_tags like '%bicycle_parking%' AND
gll.lad11nm = 'Havering'
GROUP BY gll.oa_code, gll.supgrp_cd;"
) %>% knitr::kable()

OA_Grouped_Haerving_Cycle_P
```

| oa_code | supgrp_cd | count |
|---|---|---|
| E00011452 | H | 1 |
| E00011481 | F | 1 |
| E00011659 | H | 1 |
| E00011849 | A | 1 |
| E00011914 | F | 2 |
| E00011935 | H | 4 |

```
#checking if there is any duplicate point
#Based on OA and grp_cd, there are unique bicycle parking

CD_Grouped_Haerving_Cycle_P <- dbGetQuery(
  conn = pgsql_conn,
  statement = "SELECT gll.grp_cd, count(*)
FROM greater_london_osm_point glop
INNER JOIN greater_london_loac gll
ON st_intersects(
st_transform(glop.geom, 27700),
st_transform(gll.geom, 27700)
```

```
)
WHERE glop.other_tags like '%bicycle_parking%' AND
gll.lad11nm = 'Havering'
GROUP BY gll.grp_cd;"
) %>% knitr::kable()

CD_Grouped_Haerving_Cycle_P
```

| grp_cd | count |
|---|---|
| A2 | 1 |
| F2 | 3 |
| H1 | 2 |
| H2 | 4 |

```
#Of the over 750 multipolygons in the area, only 37 intersects with 100 meter buff
er
#around the bicyle parking.
hndBff_Haerving_Cycle_P <- dbGetQuery(
  conn = pgsql_conn,
  statement = "SELECT gll.id, gll.oa_code
FROM greater_london_osm_point glop
INNER JOIN greater_london_loac gll
ON st_intersects(
st_buffer(
st_transform(glop.geom, 27700),
100),
st_transform(gll.geom, 27700)
)
WHERE glop.other_tags like '%bicycle_parking%' AND
gll.lad11nm = 'Havering';"
)
```

```
#Roads that intersect bicycle parking points in  Havering
Road_int_Haerving_Cycle_P <- dbGetQuery(
  conn = pgsql_conn,
  statement = "WITH hr
as(
SELECT glop.geom, glop.osm_id
FROM greater_london_osm_point glop
INNER JOIN greater_london_loac gll
ON st_Within(
st_transform(glop.geom, 27700),
st_transform(gll.geom, 27700)
)
WHERE glop.other_tags like '%bicycle_parking%' AND
gll.lad11nm = 'Havering'
)
SELECT hr.*,  glol.name
FROM hr
INNER JOIN greater_london_osm_line glol
```

```
ON st_intersects(st_transform(glol.geom, 27700),
st_transform(hr.geom, 27700))"
)
```

```
## Warning in postgresqlExecStatement(conn, statement, ...): RS-DBI driver warnin
g:
## ((null))
```

```
Road_19m_buff_Haerving_Cycle_P <- dbGetQuery(
  conn = pgsql_conn,
  statement = "WITH  hr
as(
SELECT glop.geom, glop.osm_id, glop.name
FROM greater_london_osm_point glop
INNER JOIN greater_london_loac gll
ON st_intersects(
st_transform(glop.geom, 27700),
st_transform(gll.geom, 27700)
)
WHERE glop.other_tags like '%bicycle_parking%' AND
gll.lad11nm = 'Havering'
)
SELECT hr.*,  glol.name, glol.highway
FROM hr
INNER JOIN greater_london_osm_line glol
ON st_intersects(st_buffer(st_transform(hr.geom, 27700),19
), st_transform(glol.geom, 27700))"
)
```

```
## Warning in postgresqlExecStatement(conn, statement, ...): RS-DBI driver warnin
g:
## ((null))
```

```
Road_19m_buff_Haerving_Cycle_P
```

```
##                                                  geom     osm_id
## 1   0101000020E610000018E6A9B3FFA6C83F1A6778584DC34940  274870159
## 2   0101000020E610000018E6A9B3FFA6C83F1A6778584DC34940  274870159
## 3   0101000020E610000018E6A9B3FFA6C83F1A6778584DC34940  274870159
## 4   0101000020E610000018E6A9B3FFA6C83F1A6778584DC34940  274870159
## 5   0101000020E610000018E6A9B3FFA6C83F1A6778584DC34940  274870159
## 6   0101000020E610000018E6A9B3FFA6C83F1A6778584DC34940  274870159
## 7   0101000020E610000018E6A9B3FFA6C83F1A6778584DC34940  274870159
## 8   0101000020E610000018E6A9B3FFA6C83F1A6778584DC34940  274870159
## 9   0101000020E610000018E6A9B3FFA6C83F1A6778584DC34940  274870159
## 10  0101000020E610000018E6A9B3FFA6C83F1A6778584DC34940  274870159
## 11  0101000020E610000018E6A9B3FFA6C83F1A6778584DC34940  274870159
## 12  0101000020E610000018E6A9B3FFA6C83F1A6778584DC34940  274870159
## 13  0101000020E6100000842C0B26FE28D03FF3EB87D860CA4940 3104472651
## 14  0101000020E6100000842C0B26FE28D03FF3EB87D860CA4940 3104472651
## 15  0101000020E6100000842C0B26FE28D03FF3EB87D860CA4940 3104472651
```

```
## 16  0101000020E6100000842C0B26FE28D03FF3EB87D860CA4940 3104472651
## 17  0101000020E6100000842C0B26FE28D03FF3EB87D860CA4940 3104472651
## 18  0101000020E6100000842C0B26FE28D03FF3EB87D860CA4940 3104472651
## 19  0101000020E61000002203D42F6C28CE3F9675A49F15CC4940 3104472670
## 20  0101000020E61000002203D42F6C28CE3F9675A49F15CC4940 3104472670
## 21  0101000020E61000002203D42F6C28CE3F9675A49F15CC4940 3104472670
## 22  0101000020E61000002203D42F6C28CE3F9675A49F15CC4940 3104472670
## 23  0101000020E61000002203D42F6C28CE3F9675A49F15CC4940 3104472670
## 24  0101000020E61000002203D42F6C28CE3F9675A49F15CC4940 3104472670
## 25  0101000020E61000002203D42F6C28CE3F9675A49F15CC4940 3104472670
## 26  0101000020E61000002203D42F6C28CE3F9675A49F15CC4940 3104472670
## 27  0101000020E61000007767EDB60BCDCF3F20F70890FCC64940 3221558719
## 28  0101000020E61000007767EDB60BCDCF3F20F70890FCC64940 3221558719
## 29  0101000020E61000007767EDB60BCDCF3F20F70890FCC64940 3221558719
## 30  0101000020E61000007767EDB60BCDCF3F20F70890FCC64940 3221558719
## 31  0101000020E61000007767EDB60BCDCF3F20F70890FCC64940 3221558719
## 32  0101000020E61000007767EDB60BCDCF3F20F70890FCC64940 3221558719
## 33  0101000020E6100000B85F9912A4ADCF3FA726C11BD2C64940 3225682668
## 34  0101000020E6100000B85F9912A4ADCF3FA726C11BD2C64940 3225682668
## 35  0101000020E6100000B85F9912A4ADCF3FA726C11BD2C64940 3225682668
## 36  0101000020E6100000B85F9912A4ADCF3FA726C11BD2C64940 3225682668
## 37  0101000020E6100000B85F9912A4ADCF3FA726C11BD2C64940 3225682668
## 38  0101000020E6100000B85F9912A4ADCF3FA726C11BD2C64940 3225682668
## 39  0101000020E6100000AC9223F83A04D03FF73878CB7AC74940 3244383768
## 40  0101000020E6100000AC9223F83A04D03FF73878CB7AC74940 3244383768
## 41  0101000020E6100000AC9223F83A04D03FF73878CB7AC74940 3244383768
## 42  0101000020E6100000AC9223F83A04D03FF73878CB7AC74940 3244383768
## 43  0101000020E6100000AC9223F83A04D03FF73878CB7AC74940 3244383768
## 44  0101000020E6100000AC9223F83A04D03FF73878CB7AC74940 3244383768
## 45  0101000020E6100000AC9223F83A04D03FF73878CB7AC74940 3244383768
## 46  0101000020E6100000AC9223F83A04D03FF73878CB7AC74940 3244383768
## 47  0101000020E6100000AC9223F83A04D03FF73878CB7AC74940 3244383768
## 48  0101000020E6100000AC9223F83A04D03FF73878CB7AC74940 3244383768
## 49  0101000020E6100000AC9223F83A04D03FF73878CB7AC74940 3244383768
## 50  0101000020E6100000AC9223F83A04D03FF73878CB7AC74940 3244383768
## 51  0101000020E6100000AC9223F83A04D03FF73878CB7AC74940 3244383768
## 52  0101000020E6100000AC9223F83A04D03FF73878CB7AC74940 3244383768
## 53  0101000020E6100000AC9223F83A04D03FF73878CB7AC74940 3244383768
## 54  0101000020E6100000AC9223F83A04D03FF73878CB7AC74940 3244383768
## 55  0101000020E6100000AC9223F83A04D03FF73878CB7AC74940 3244383768
## 56  0101000020E6100000AC9223F83A04D03FF73878CB7AC74940 3244383768
## 57  0101000020E610000076A911FA997AC83FC4CA0D2B81CB4940 3749182495
## 58  0101000020E610000076A911FA997AC83FC4CA0D2B81CB4940 3749182495
## 59  0101000020E610000076A911FA997AC83FC4CA0D2B81CB4940 3749182495
## 60  0101000020E610000076A911FA997AC83FC4CA0D2B81CB4940 3749182495
## 61  0101000020E610000076A911FA997AC83FC4CA0D2B81CB4940 3749182495
## 62  0101000020E610000076A911FA997AC83FC4CA0D2B81CB4940 3749182495
## 63  0101000020E610000076A911FA997AC83FC4CA0D2B81CB4940 3749182495
## 64  0101000020E610000076A911FA997AC83FC4CA0D2B81CB4940 3749182495
## 65  0101000020E610000076A911FA997AC83FC4CA0D2B81CB4940 3749182495
## 66  0101000020E610000076A911FA997AC83FC4CA0D2B81CB4940 3749182495
## 67  0101000020E610000076A911FA997AC83FC4CA0D2B81CB4940 3749182495
## 68  0101000020E61000009CEFF1536614D03F03A8F3F285C74940 4081920759
## 69  0101000020E61000009CEFF1536614D03F03A8F3F285C74940 4081920759
## 70  0101000020E61000009CEFF1536614D03F03A8F3F285C74940 4081920759
```

```
## 71  0101000020E61000009CEFF1536614D03F03A8F3F285C74940 4081920759
## 72  0101000020E61000009CEFF1536614D03F03A8F3F285C74940 4081920759
## 73  0101000020E61000009CEFF1536614D03F03A8F3F285C74940 4081920759
## 74  0101000020E61000009CEFF1536614D03F03A8F3F285C74940 4081920759
## 75  0101000020E61000009CEFF1536614D03F03A8F3F285C74940 4081920759
## 76  0101000020E61000009CEFF1536614D03F03A8F3F285C74940 4081920759
## 77  0101000020E61000009CEFF1536614D03F03A8F3F285C74940 4081920759
## 78  0101000020E61000009CEFF1536614D03F03A8F3F285C74940 4081920759
## 79  0101000020E61000009CEFF1536614D03F03A8F3F285C74940 4081920759
## 80  0101000020E61000009CEFF1536614D03F03A8F3F285C74940 4081920759
## 81  0101000020E61000009CEFF1536614D03F03A8F3F285C74940 4081920759
## 82  0101000020E61000009CEFF1536614D03F03A8F3F285C74940 4081920759
## 83  0101000020E61000009CEFF1536614D03F03A8F3F285C74940 4081920759
## 84  0101000020E61000009CEFF1536614D03F03A8F3F285C74940 4081920759
## 85  0101000020E61000009CEFF1536614D03F03A8F3F285C74940 4081920759
## 86  0101000020E61000009CEFF1536614D03F03A8F3F285C74940 4081920759
## 87  0101000020E61000009CEFF1536614D03F03A8F3F285C74940 4081920759
## 88  0101000020E61000009CEFF1536614D03F03A8F3F285C74940 4081920759
## 89  0101000020E61000009CEFF1536614D03F03A8F3F285C74940 4081920759
## 90  0101000020E6100000F53B5E921422D03F8FA3DEF87FC74940 2320751625
## 91  0101000020E6100000F53B5E921422D03F8FA3DEF87FC74940 2320751625
## 92  0101000020E6100000F53B5E921422D03F8FA3DEF87FC74940 2320751625
## 93  0101000020E6100000F53B5E921422D03F8FA3DEF87FC74940 2320751625
## 94  0101000020E6100000F53B5E921422D03F8FA3DEF87FC74940 2320751625
## 95  0101000020E6100000F53B5E921422D03F8FA3DEF87FC74940 2320751625
## 96  0101000020E6100000F53B5E921422D03F8FA3DEF87FC74940 2320751625
## 97  0101000020E6100000F53B5E921422D03F8FA3DEF87FC74940 2320751625
## 98  0101000020E6100000F53B5E921422D03F8FA3DEF87FC74940 2320751625
## 99  0101000020E6100000A1FC38F5DC1DD03FD40FEA2285C74940 1344561848
## 100 0101000020E6100000A1FC38F5DC1DD03FD40FEA2285C74940 1344561848
## 101 0101000020E6100000A1FC38F5DC1DD03FD40FEA2285C74940 1344561848
## 102 0101000020E6100000A1FC38F5DC1DD03FD40FEA2285C74940 1344561848
## 103 0101000020E6100000A1FC38F5DC1DD03FD40FEA2285C74940 1344561848
## 104 0101000020E6100000A1FC38F5DC1DD03FD40FEA2285C74940 1344561848
## 105 0101000020E6100000A1FC38F5DC1DD03FD40FEA2285C74940 1344561848
## 106 0101000020E6100000A1FC38F5DC1DD03FD40FEA2285C74940 1344561848
## 107 0101000020E6100000A1FC38F5DC1DD03FD40FEA2285C74940 1344561848
## 108 0101000020E6100000A1FC38F5DC1DD03FD40FEA2285C74940 1344561848
## 109 0101000020E6100000A1FC38F5DC1DD03FD40FEA2285C74940 1344561848
## 110 0101000020E6100000A1FC38F5DC1DD03FD40FEA2285C74940 1344561848
## 111 0101000020E6100000A1FC38F5DC1DD03FD40FEA2285C74940 1344561848
## 112 0101000020E6100000A1FC38F5DC1DD03FD40FEA2285C74940 1344561848
##                                name
## 1                              <NA>
## 2                              <NA>
## 3                              <NA>
## 4                              <NA>
## 5                              <NA>
## 6                              <NA>
## 7                              <NA>
## 8                              <NA>
## 9                              <NA>
## 10                             <NA>
## 11                             <NA>
## 12                             <NA>
```

```
## 13                          <NA>
## 14                          <NA>
## 15                          <NA>
## 16                          <NA>
## 17                          <NA>
## 18                          <NA>
## 19                          <NA>
## 20                          <NA>
## 21                          <NA>
## 22                          <NA>
## 23                          <NA>
## 24                          <NA>
## 25                          <NA>
## 26                          <NA>
## 27                          <NA>
## 28                          <NA>
## 29                          <NA>
## 30                          <NA>
## 31                          <NA>
## 32                          <NA>
## 33                          <NA>
## 34                          <NA>
## 35                          <NA>
## 36                          <NA>
## 37                          <NA>
## 38                          <NA>
## 39                          <NA>
## 40                          <NA>
## 41                          <NA>
## 42                          <NA>
## 43                          <NA>
## 44                          <NA>
## 45                          <NA>
## 46                          <NA>
## 47                          <NA>
## 48                          <NA>
## 49                          <NA>
## 50                          <NA>
## 51                          <NA>
## 52                          <NA>
## 53                          <NA>
## 54                          <NA>
## 55                          <NA>
## 56                          <NA>
## 57                          <NA>
## 58                          <NA>
## 59                          <NA>
## 60                          <NA>
## 61                          <NA>
## 62                          <NA>
## 63                          <NA>
## 64                          <NA>
## 65                          <NA>
## 66                          <NA>
## 67                          <NA>
```

```
## 68  Upminster Station Bike Compound
## 69  Upminster Station Bike Compound
## 70  Upminster Station Bike Compound
## 71  Upminster Station Bike Compound
## 72  Upminster Station Bike Compound
## 73  Upminster Station Bike Compound
## 74  Upminster Station Bike Compound
## 75  Upminster Station Bike Compound
## 76  Upminster Station Bike Compound
## 77  Upminster Station Bike Compound
## 78  Upminster Station Bike Compound
## 79  Upminster Station Bike Compound
## 80  Upminster Station Bike Compound
## 81  Upminster Station Bike Compound
## 82  Upminster Station Bike Compound
## 83  Upminster Station Bike Compound
## 84  Upminster Station Bike Compound
## 85  Upminster Station Bike Compound
## 86  Upminster Station Bike Compound
## 87  Upminster Station Bike Compound
## 88  Upminster Station Bike Compound
## 89  Upminster Station Bike Compound
## 90                             <NA>
## 91                             <NA>
## 92                             <NA>
## 93                             <NA>
## 94                             <NA>
## 95                             <NA>
## 96                             <NA>
## 97                             <NA>
## 98                             <NA>
## 99                             <NA>
## 100                            <NA>
## 101                            <NA>
## 102                            <NA>
## 103                            <NA>
## 104                            <NA>
## 105                            <NA>
## 106                            <NA>
## 107                            <NA>
## 108                            <NA>
## 109                            <NA>
## 110                            <NA>
## 111                            <NA>
## 112                            <NA>
##                                                 name      highway
## 1                                  Knightswood Road  residential
## 2                                              <NA>      service
## 3                                   Ingrebourne Way     cycleway
## 4                                      Rainham Road      primary
## 5                                              <NA>      service
## 6                                      Rainham Road      primary
## 7                                              <NA>      footway
## 8                                              <NA>     cycleway
## 9                 London Buses route 103 → Chase Cross         <NA>
```

```
## 10                                Ingreborne Valley Way         <NA>
## 11                             London LOOP (Section 23)         <NA>
## 12        London Buses route 103 → Rainham Interchange         <NA>
## 13                                                <NA>      service
## 14                                                <NA>     cycleway
## 15                                                <NA>      footway
## 16                                                <NA>      footway
## 17                                Ingreborne Valley Way         <NA>
## 18                             London LOOP (Section 22)         <NA>
## 19                                      Arundel Road  residential
## 20                                       Avenue Road     tertiary
## 21                                   Queens Park Road  residential
## 22                                       Avenue Road     tertiary
## 23                                      Station Road     tertiary
## 24                                                <NA>      footway
## 25  London Buses route 496 → Romford, Queen's Hospital         <NA>
## 26                 London Buses route 496 → Harold Wood         <NA>
## 27                                                <NA>         <NA>
## 28                                    Corbets Tey Road    secondary
## 29                   London Buses route 370 → Lakeside         <NA>
## 30              London Buses route 370 → Romford Market         <NA>
## 31              London Buses route 370 → Romford Market         <NA>
## 32                 London Buses route 370 → Corbets Tey         <NA>
## 33                                    Stewart Avenue  residential
## 34                                    Corbets Tey Road    secondary
## 35                   London Buses route 370 → Lakeside         <NA>
## 36              London Buses route 370 → Romford Market         <NA>
## 37              London Buses route 370 → Romford Market         <NA>
## 38                 London Buses route 370 → Corbets Tey         <NA>
## 39                                      Station Road     tertiary
## 40                                      Station Road     tertiary
## 41                                                <NA>      footway
## 42              London Buses route 248 → Romford Market         <NA>
## 43                   London Buses route 370 → Lakeside         <NA>
## 44                                Ingreborne Valley Way         <NA>
## 45                  London Buses route 346 → Upminster         <NA>
## 46                  London Buses route 652 → Upminster         <NA>
## 47                    London Buses route 646 → Cranham         <NA>
## 48                  London Buses route 646 → Noak Hill         <NA>
## 49                    London Buses route 248 → Cranham         <NA>
## 50                    London Buses route 648 → Cranham         <NA>
## 51              London Buses route 648 → Romford Market         <NA>
## 52                   London Buses route 347 → Ockendon         <NA>
## 53             London Buses route 347 → Romford Station         <NA>
## 54              London Buses route 370 → Romford Market         <NA>
## 55              London Buses route 370 → Romford Market         <NA>
## 56                 London Buses route 370 → Corbets Tey         <NA>
## 57                                                <NA>         <NA>
## 58                                                <NA>      footway
## 59                                                <NA>        steps
## 60                                                <NA>         <NA>
## 61                                                <NA>      footway
## 62                                                <NA>      footway
## 63                                                <NA>      footway
## 64                                                <NA>      footway
```

```
## 65                                                 <NA>        steps
## 66                                                 <NA>     cycleway
## 67                                                 <NA>         <NA>
## 68                                  Station Approach unclassified
## 69                                                 <NA>         <NA>
## 70                                                 <NA>         <NA>
## 71                    London, Tilbury & Southend Line         <NA>
## 72                   London Buses route 370 → Lakeside         <NA>
## 73                  London Buses route 346 → Upminster         <NA>
## 74                    London Buses route 646 → Cranham         <NA>
## 75                  London Buses route 646 → Noak Hill         <NA>
## 76                    London Buses route 648 → Cranham         <NA>
## 77              London Buses route 648 → Romford Market         <NA>
## 78              London Buses route 370 → Romford Market         <NA>
## 79           C2C: London - Chafford Hundred - Southend         <NA>
## 80              C2C: Shoeburyness - London (semi-fast)         <NA>
## 81              C2C: London - Shoeburyness (semi-fast)         <NA>
## 82               C2C: Shoeburyness - London (stopping)         <NA>
## 83               C2C: London - Shoeburyness (stopping)         <NA>
## 84                          C2C: Leigh-on-Sea - London         <NA>
## 85                          C2C: London - Leigh-on-Sea         <NA>
## 86           C2C: Southend - Chafford Hundred - London         <NA>
## 87              London Buses route 370 → Romford Market         <NA>
## 88                 London Buses route 370 → Corbets Tey         <NA>
## 89                                                 <NA>         <NA>
## 90                          London, Tilbury & Southend         <NA>
## 91                                                 <NA>      service
## 92                                                 <NA>         <NA>
## 93                          London, TIlbury & Southend         <NA>
## 94                        London, Tilbury & Southend Line         <NA>
## 95              C2C: Shoeburyness - London (semi-fast)         <NA>
## 96               C2C: Shoeburyness - London (stopping)         <NA>
## 97                          C2C: Leigh-on-Sea - London         <NA>
## 98           C2C: Southend - Chafford Hundred - London         <NA>
## 99                                                 <NA>         <NA>
## 100                         London, TIlbury & Southend         <NA>
## 101                                                <NA>         <NA>
## 102                                                <NA>      footway
## 103                       London, Tilbury & Southend Line         <NA>
## 104          C2C: London - Chafford Hundred - Southend         <NA>
## 105             C2C: Shoeburyness - London (semi-fast)         <NA>
## 106             C2C: London - Shoeburyness (semi-fast)         <NA>
## 107              C2C: Shoeburyness - London (stopping)         <NA>
## 108              C2C: London - Shoeburyness (stopping)         <NA>
## 109                         C2C: Leigh-on-Sea - London         <NA>
## 110                         C2C: London - Leigh-on-Sea         <NA>
## 111          C2C: Southend - Chafford Hundred - London         <NA>
## 112                                                <NA>         <NA>
```

```
#Buildings that intersect bicycle parking in Havering
Building_int_Haerving_Cycle_P <- dbGetQuery(
  conn = pgsql_conn,
  statement = "WITH  hm
as(
```

```
SELECT glop.geom, glop.osm_id, glop.name
FROM greater_london_osm_point glop
INNER JOIN greater_london_loac gll
ON st_intersects(
st_transform(glop.geom, 27700),
st_transform(gll.geom, 27700)
)
WHERE glop.other_tags like '%bicycle_parking%' AND
gll.lad11nm = 'Havering'
)
SELECT hm.geom, hm.osm_id, hm.name,  glop2.building, glop2.tourism, glop2.sport, g
lop2.office, glop2.geom
FROM hm
INNER JOIN greater_london_osm_polygon glop2
ON st_intersects(st_transform( hm.geom, 27700)
, st_transform(glop2.geom, 27700))
WHERE glop2.building IS NOT NULL"
)
```

```
## Warning in postgresqlExecStatement(conn, statement, ...): RS-DBI driver warnin
g:
## ((null))

## Warning in postgresqlExecStatement(conn, statement, ...): RS-DBI driver warnin
g:
## ((null))
```

```
Building_int_Haerving_Cycle_P
```

```
## [1] geom     osm_id   name     building tourism  sport    office   geom
## <0 rows> (or 0-length row.names)
```

```
#no building intersect bicycle parkings
```

```
#Buildings within 50m distance from the bicycle parkings
Building_50m_away_Haerving_Cycle_P <- dbGetQuery(
  conn = pgsql_conn,
  statement = "WITH  hm
as(
SELECT glop.geom, glop.osm_id, glop.name, glop.other_tags
FROM greater_london_osm_point glop
INNER JOIN greater_london_loac gll
ON st_intersects(
st_transform(glop.geom, 27700),
st_transform(gll.geom, 27700)
)
WHERE glop.other_tags like '%bicycle_parking%' AND
gll.lad11nm = 'Havering'
)
SELECT hm.geom, hm.osm_id, hm.name,  glop2.building, glop2.tourism, glop2.sport, g
lop2.office, glop2.geom, hm.other_tags
```

```
FROM hm
INNER JOIN greater_london_osm_polygon glop2
ON st_dwithin(st_transform( hm.geom, 27700)
, st_transform(glop2.geom, 27700), 50)
WHERE glop2.building IS NOT NULL"
)
```

```
## Warning in postgresqlExecStatement(conn, statement, ...): RS-DBI driver warnin
g:
## ((null))

## Warning in postgresqlExecStatement(conn, statement, ...): RS-DBI driver warnin
g:
## ((null))
```

```
class(Building_50m_away_Haerving_Cycle_P)
```

```
## [1] "data.frame"
```

```
All_road <- st_read(
  pgsql_conn,
  query = "SELECT * FROM greater_london_osm_line;"
)
```

```
## Warning: RS-DBI driver warning: ((null))
```

```
Hovering <- st_read(
  pgsql_conn,
  query = "  SELECT * FROM greater_london_loac gll
WHERE lad11nm = 'Havering';"
)
```

```
## Warning: RS-DBI driver warning: ((null))
```

```
all_building <- st_read(
  pgsql_conn,
  query = "SELECT * FROM greater_london_osm_polygon as glop2
  WHERE glop2.building IS NOT NULL;"
)
```

```
## Warning: RS-DBI driver warning: ((null))
```

```
cycle <- st_read(
  pgsql_conn,
  query = "SELECT *
      FROM greater_london_osm_point as glop
      WHERE glop.other_tags like '%bicycle_parking%'"
)
```

```
## Warning: RS-DBI driver warning: ((null))
```

```
Cycle_Pk_in_Haerving <- st_read(
  pgsql_conn,
  query = "
    SELECT glop.geom, gll.supgrp_cd, gll.grp_cd
      FROM greater_london_osm_point as glop
INNER JOIN greater_london_loac as gll
        ON ST_within(
                ST_Transform(glop.geom, 27700),
                ST_Transform(gll.geom, 27700)
             )
             WHERE glop.other_tags like '%bicycle_parking%' AND
gll.lad11nm = 'Havering';"
)
```

```
## Warning: RS-DBI driver warning: ((null))
```

```
buildingwithin50m <- st_read(
  pgsql_conn,
  query = "WITH  hm
as(
  SELECT glop.geom, glop.osm_id, glop.name, glop.other_tags
  FROM greater_london_osm_point glop
  INNER JOIN greater_london_loac gll
  ON st_intersects(
    st_transform(glop.geom, 27700),
    st_transform(gll.geom, 27700)
  )
  WHERE glop.other_tags like '%bicycle_parking%' AND
  gll.lad11nm = 'Havering'
)
SELECT hm.geom, hm.osm_id, hm.name,  glop2.building, glop2.tourism, glop2.sport, g
lop2.office, glop2.geom, hm.other_tags
FROM hm
INNER JOIN greater_london_osm_polygon glop2
ON st_dwithin(st_transform( hm.geom, 27700)
              , st_transform(glop2.geom, 27700), 50)
WHERE glop2.building IS NOT NULL;"
)
```

```
## Warning: RS-DBI driver warning: ((null))

## Warning: RS-DBI driver warning: ((null))
```

```
roads19maway <- st_read(
  pgsql_conn,
  query = "
  WITH  hr
as(
SELECT glop.geom, glop.osm_id, glop.name
```

```
FROM greater_london_osm_point glop
INNER JOIN greater_london_loac gll
ON st_intersects(
st_transform(glop.geom, 27700),
st_transform(gll.geom, 27700)
)
WHERE glop.other_tags like '%bicycle_parking%' AND
gll.lad11nm = 'Havering'
)
SELECT hr.*, glol.geom , glol.name, glol.highway
FROM hr
INNER JOIN greater_london_osm_line glol
ON st_intersects(st_buffer(st_transform(hr.geom, 27700),19
), st_transform(glol.geom, 27700))
"
)
```

```
## Warning: RS-DBI driver warning: ((null))

## Warning: RS-DBI driver warning: ((null))
```

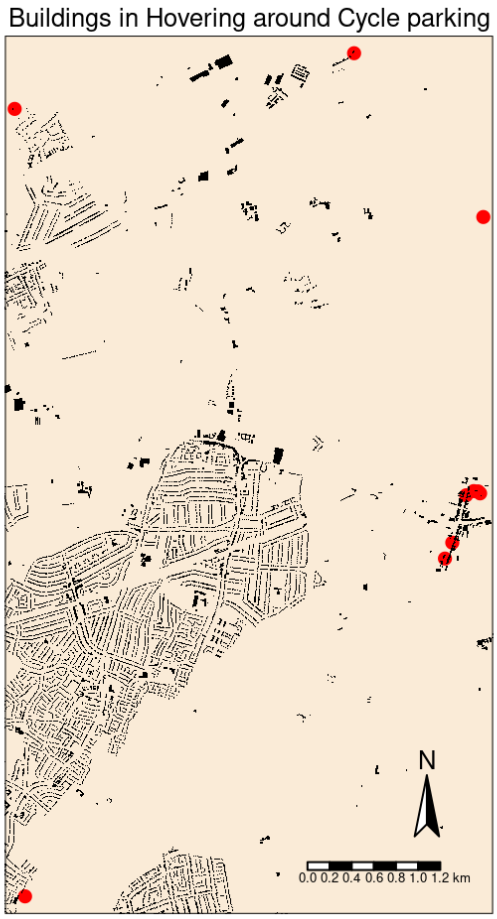
```
pall <- c("#E41A1C", "#377EB8", "#4DAF4A", "#984EA3", "#FF7F00", "#FFFF33", "#A656
28", "#F781BF",'#01BEFE')
```

```
#Bicycle Parkings in Havering
tm_shape(Cycle_Pk_in_Haerving) +
  # Represent them as filled polygons
  tm_dots(col = 'red', size = 0.18) +
  # Add the line shapes
  tm_shape(All_road) +
  # Represent them as lines
  tm_lines(col = "#333333", alpha = 0.7) +
  tm_layout(
    main.title = 'Roads in Hovering around Cycle parking', bg.color="lightblue",
    main.title.size = 0.8, main.title.position="center") +
  tm_compass() + tm_scale_bar() +
  tm_basemap(server="OpenStreetMap",alpha=0.5)
```

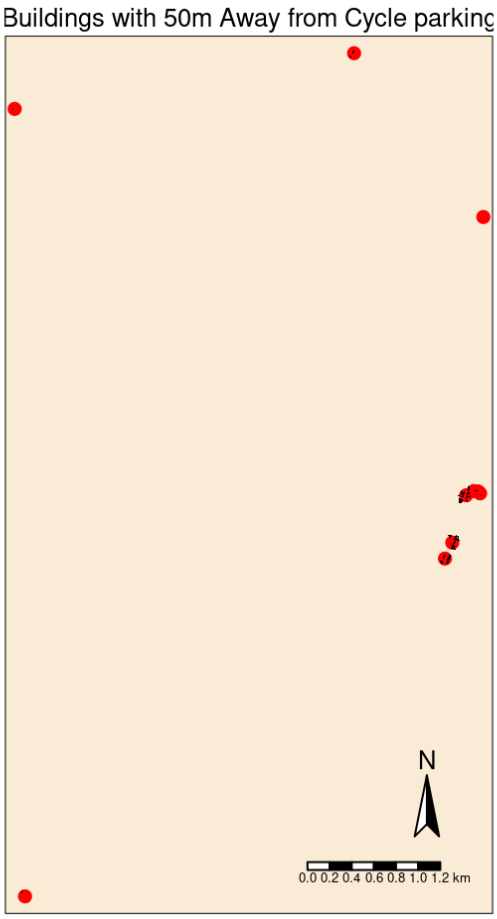

```
tm_shape(Cycle_Pk_in_Haerving) +
  # Represent them as filled polygons
  tm_dots(col = 'red', size = 0.18) +
  # Add the line shapes
  tm_shape(all_building,) +
  # Represent them as lines
  tm_fill( col = 'black', lwd = 3)  +
  tmap_options(check.and.fix = TRUE) +
  tm_layout(
    main.title = 'Buildings in Hovering around Cycle parking', bg.color="antiquewh
ite",
    main.title.size = 0.8,main.title.position="center") +
  tm_compass() + tm_scale_bar() +
  tm_basemap(server="OpenStreetMap",alpha=0.5)
```

```
## Warning: The shape all_building is invalid. See sf::st_is_valid
```

```
## Warning: The shape all_building contains empty units.
```

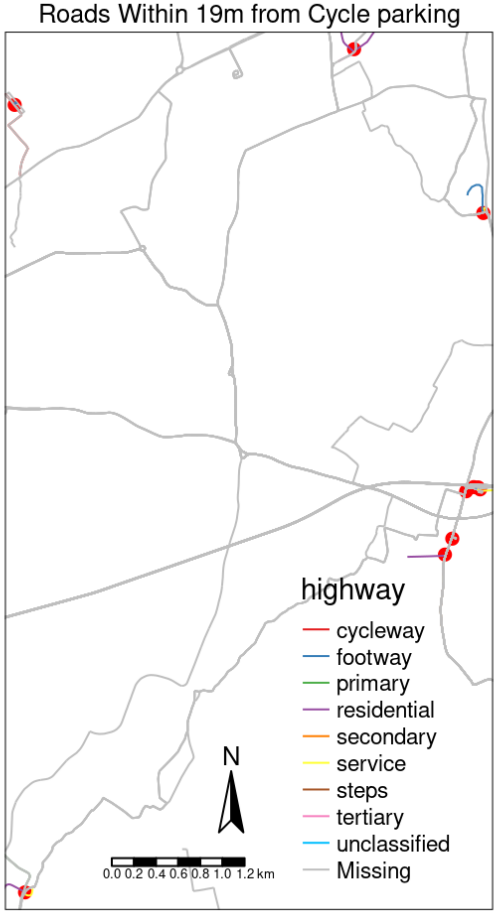



```
tm_shape(Cycle_Pk_in_Haerving) +
  # Represent them as filled polygons
  tm_dots(col = 'red', size = 0.18) +
  # Add the line shapes
  tm_shape(buildingwithin50m) +
  # Represent them as lines
  tm_fill(col = 'black', lwd = 3)  +
  tmap_options(check.and.fix = TRUE) +
  tm_layout(
    main.title = 'Buildings with 50m Away from Cycle parking', bg.color="antiquewh
ite",
    main.title.size = 0.8,main.title.position="center") +
  tm_compass() + tm_scale_bar() +
  tm_basemap(server="OpenStreetMap",alpha=0.5)
```




```
tm_shape(Cycle_Pk_in_Haerving) +
  # Represent them as filled polygons
  tm_dots(col = 'red', size = 0.18) +
  # Add the line shapes
  tm_shape(roads19maway) +
  # Represent them as lines
  tm_lines('highway', palette = pall)  +
  tmap_options(check.and.fix = TRUE) +
  tm_layout(
            main.title = 'Roads Within 19m from Cycle parking',
            main.title.size = 0.8,main.title.position="center") +
  tm_compass() + tm_scale_bar() +
  tm_basemap(server="OpenStreetMap",alpha=0.5)
```

Roads Within 19m from Cycle parking



# Part 1

Just like many databases around, the dataset “GY7708_2021-22_Assignment_1--Statues-Wikidata.csv” given for this course work contains anomalies. These anomalies affect the structure of the data, making it redundant, contain duplicate and lack integrity among others. The effect of these problems is that it would likely lead to deletion, insertion, and update anomaly, making it difficult to work with our data. Normalization is thus introduced to help solve these problems. As taught in class, the first, second and third normal form are important for database to be in the right structure.

As seen in the GY7708_2021-22_Assignment_1--Statues-Wikidata.csv, it does not comply with the first normal form rule which opines that every column must be in atomic format and should not contain list of duplicated information. This anomaly can be found in the DepictedAltLabel column of the dataset. In order to ensure that the dataset set comply with the first normal form, we need to break the list variables in the rows of DepictedAltLabel column into new ones, allowing the column dataset to be in atomic form. Also, to ensure that the dataset can be identified and traced easily, I created a column “ID” which was used as the primary key for the table.

While I depended on the ID and the Statue column as composite primary keys, I noticed that the table does not conform with the second normal rule as some of the columns have partial dependence. Hence, I partitioned the table into two, keeping the columns inception which completely dependents of ID and statue. The other table contained other columns that are partially dependent. This made the table conform with the second normal form rule, however, it contradicts the third normal form rule.

While the third normal form states that no non-primary variable should depend on another non-primary variable i.e., there should be no transitive dependency for non-prime attributes. To ensure that this principle is met, I divided the table into five, allowing columns that are dependent on other columns stay with their dependee.

For each of the five tables created, a column is selected as the primary key, this allows identification and integrity of the dataset. Also, to link the tables together, I created foreign key.

The foreign key is regarded as a primary key in one table link to another table that contains the similar column but not as its primary key.

To ensure that all the tables are well linked with their dependent and dependee, I confirmed the integrity of the table in r with the function dm_examine_constraints(). This showed that the tables are well linked.

# Part 2

The Points of Interest (POIs) assigned to me is Bicycle_parking in the Havering region of London. Using the ST_within function in PostGIS, I was able to identify the POIs assigned to me in the study area. I found out that there are ten Bicycle_parking points in Havering. The Bicycle_parking are distributed across the area, however, there clusters in some part of the study area. When counted and grouped by the ID of the POI, it was observed that there are 10 unique bicycle ID. However, four bicycle_parkings were in the output area ‘E00011935’ and super group H. While 4 other bicycle_parking units were distributed among four output areas, output area E00011914 also had two bicycle_parking.

To further enhance the spatial analysis functions ST_buffer and ST_intersects were used. With the help of the ST_intersect, I was able to check the number of roads that intersect the bicycle_parking points, and the result indicates that three roads intersected the bicycle_parking, and one of them is the ‘cycleway.

According to the Cambridge Cycle Parking Guide 2008, bicycle parking should be situated atleast 20 meters away from the highway. This made me conduct an analysis on the type of roads that intersects a 19-meter buffer around the bicycle_parking. The spatial analysis revealed that 112 different roads intersected the 19-meter buffer around the points. However, 5 of them were tertiary i.e highway. Other road types include footway, service, secondary, residential and primary roads.

Also, according to the Standard for Public parking, bicycle parking can only service building 50 meters around it adequately

These queries were also executed in R to give images of the spatial analysis performed. After connecting the R studio to the database, I performed the same query done in PostGreSQL. I also used tmap to print out the images of the indicating the spatial relationship between POI and other features in the study are.

# Reference


1. Camcycle.org.uk. 2008. *How to provide Cycle Parking: a step-by-step guide for planners and providers*. [online] Available at: <https://www.camcycle.org.uk/files/resources/cycleparking/guide/cycleparkingguide.pdf> [Accessed 7 March 2022].

2. Bicycleassociation.org.uk. 2021. *STANDARDS FOR PUBLIC CYCLE PARKING*. [online] Available at: <https://www.bicycleassociation.org.uk/wp-content/uploads/2021/06/05132-Cycle-Parking-and-Security-Standards-June-2021-REV-5.pdf> [Accessed 7 March 2022].