

Ara Proje Raporu

by Yağmur Duran

Submission date: 23-Jan-2022 01:51AM (UTC+0300)

Submission ID: 1745996224

File name: UK_Ara_18011071_18011903.pdf (10.02M)

Word count: 4560

Character count: 30422

TÜRKİYE CUMHURİYETİ
YILDIZ TEKNİK ÜNİVERSİTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ



**ZAMAN UZAMSAL SENTETİK VERİ SETİ ÜRETİMİ VE
SORGULANMASI**

18011903 – Ramiz Mammadli

18011071 – Elif Yağmur Duran

BİLGİSAYAR PROJESİ

Danışman
Dr. Ögr. Üyesi Mustafa Utlu Kalay

Ocak, 2022

TEŞEKKÜR

Bu projenin tamamlanmasında bize yardımcı olan rehber öğretmenimiz ve fakültemizin saygın öğretim üyesi Yrd. Doç. Dr. Mustafa Utku Kalay'a teşekkürlerimizi sunarız.

Ayrıca rehberlik konusunda yardımını sunan sınıf arkadaşlarımıza özel şükranlarımızı sunarız.

Ramiz Mammadlı
Elif Yağmur Duran

İÇİNDEKİLER

KISALTMA LİSTESİ	v
ŞEKİL LİSTESİ	vi
ÖZET	vii
ABSTRACT	viii
1 Ön inceleme	1
2 Fizibilite	2
2.1 Teknik Fizibilite	2
2.1.1 Yazılım Fizibilitesi	2
2.1.2 Donanım Fizibilitesi	3
2.1.3 İş ve Zaman Fizibilitesi	3
2.1.4 Yasal Fizibilite	3
2.1.5 Ekonomik Fizibilite	3
3 Sistem Analizi	4
4 Sistem Tasarımı	5
4.1 Ön Hazırlık	5
4.1.1 Haritanın bulunması ve PgAdmin 4 ortamında PostGIS'in bağlanması	5
4.1.2 QGIS Ortamı ile Bağlantı Kurulması ve Diğer Shapefile'ların Eklenmesi	6
5 Uygulama	8
5.1 QGIS'te Sentetik Veri Üretimi	8
5.1.1 CRS formatının ayarlanması	8
5.1.2 Rota Üretimi	9
5.1.3 Filtreleme İşlemleri	9
5.1.4 Nokta üretimi	11
5.1.5 Üretilmiş noktalara zaman etiketi atanması	11

5.2 PostGIS'te 'geom' Sütununa Zaman Boyutunun Eklenmesi	13
6 Deneysel Sonuçlar	18
6.1 İndexleme İşlemının Açıklaması ve Kullanılma Sebepleri	18
6.1.1 GIST	19
6.1.2 SPGIST	19
6.1.3 BRIN	20
7 Performans Analizleri ve Yapılan Çıkarımlar	21
8 Sonuç	24
Referanslar	25
Özgeçmiş	26

KISALTMA LİSTESİ

OSM	Open Street Map
SQL ¹⁰	Structured Query Language
GIS	Geographic Information System
CRS	Coordinate Reference System
.shp	Shapefile
QGIS	Quantum Geographic Information System
API	Application Programming Interface

ŞEKİL LİSTESİ

Şekil 2.1 Gantt Diyagramı	3
Şekil 4.1 PostGIS uzantısının bağlanması	5
Şekil 4.2 Geometri tablosunun oluşturulması	6
Şekil 4.3 QGIS'e, indirilen .shp uzantılı dosyaların aktarılması	7
Şekil 4.4 Ham verinin QGIS ortamında görüntülenmesi	7
Şekil 5.1 Projenin CRS formatı değiştirildikten sonraki görüntüsü	9
Şekil 5.2 Rastgele seçilmiş 2500 tane sokak	10
Şekil 5.3 80 metre ve üstü, rasgele seçilmiş sokakların bir kısmı	10
Şekil 5.4 Önceden rasgele seçilmiş rotalar üzerinde, 40 metreden bir üretilmiş noktalar	11
Şekil 5.5 Noktanın bulunduğu çizgi üzerinde açısal değerinin bulunması .	13
Şekil 5.6 Noktaların özelliklerini gösteren veriseti	13
Şekil 5.7 LAST_POINTS tablosundaki değerlerin yerleşme şekli	14
Şekil 5.8 LAST_POINTS tablosunda ST_X(geom) ve ST_Y(geom) değerlerinin gösterimi	15
Şekil 5.9 SELECT EXTRACT fonksiyonunun test edilmesi	15
Şekil 5.10 ST_MakePointM fonksiyonunun test edilmesi	16
Şekil 5.11 AddGeometryColumn fonksiyonunun çalıştırılması	16
Şekil 5.12 insert into ile yeni tabloya kayıtların eklenmesi	17
Şekil 5.13 Yeni tablonun son hali	17
Şekil 6.1 Bounding Boxların yerlesimi	18
Şekil 6.2 Create Index Using GIST sorgusu	19
Şekil 6.3 Create Index Using SPGIST sorgusu	20
Şekil 6.4 Create Index Using BRIN sorgusu	20
Şekil 7.1 ST_DWithin ile yapılan büyük uzaklık sorgusu	21
Şekil 7.2 ST_DWithin ile yapılan küçük uzaklık sorgusu	22
Şekil 7.3 Büyük uzaklık sorgusunun çalıştırılma süresinin index çeşitlerine göre karşılaştırılması	22
Şekil 7.4 Küçük uzaklık sorgusunun çalıştırılma süresinin index çeşitlerine göre karşılaştırılması	23

ÖZET

ZAMAN UZAMSAL SENTETİK VERİ SETİ ÜRETİMİ VE SORGULANMASI

Ramiz Mammadli

Elif Yağmur Duran

¹ Bilgisayar Mühendisliği Bölümü

Bilgisayar Projesi

Danışman: Dr. Ögr. Üyesi Mustafa Utku Kalay

İstanbul günümüzde dünyanın en ünlü ve kalabalık metropol şehirlerinden biri olarak bilinmektedir. 2500 yıllık tarihi boyunca birçok medeniyete ev sahipliği yapmış olan şehrimiz, yıllar içerisinde farklı geçmişlerden birçok insanı barındırmayı başarmıştır. Şehirden geçen her yeni insan grubu ile yeni yollar, sokaklar ve mahalleler inşa edilmiştir ve imar asla durmamıştır. Dolayısıyla da şehrin haritası 2000 yıldır değişmektedir. Şüphesiz, tarihsel olarak böylesine karmaşık bir haritanın analizi de aynı derecede zor olacaktır. Bu projede de tam olarak bunu yapmayı hedefledik. Mentorümüz Doç. Dr. Utku Kalay'ın yardımıyla QGIS üzerinde bir uzay-zaman veri tabanı oluşturduk ve bunu PostGIS aracı yardımıyla PostgreSQL ortamına yüklemeyi başardık. Ardından, işleri daha verimli hale getirmek için verilerimize belirli optimizasyon prosedürleri uyguladık. Burada yazdığımız scriptler ile sokakların belirli koşullarını göz önünde bulundurarak oluşturulan rotalara bir zaman faktörü eklemeyi başardık. Son olarak sentetik verilerimizi PostgreSQL ortamına geri çekerek 3 boyutlu zaman uzamsal veri haline getirdik, çeşitli indeksleme yöntemleri uyguladık ve performans analizleri yaptık. Yaptığımız analizle, farklı indeksleme tekniklerinin farklılıklarını, artıları ve eksileri, kendilerini belleğe nasıl yerleştirdikleri, PostgreSQL'in indeksleri kullanma konusunda nasıl çalıştığı gibi birçok farklı faktörü gözlemledik, proje sonunda da bulgularımızı görselleştirdik.

Anahtar Kelimeler: İstanbul, yol analizi, veri seti üretimi, QGIS

ABSTRACT

SPATIO - TEMPORAL SYNTHETIC DATASET GENERATION AND QUERYING

Ramiz Mammadli

Elif Yağmur Duran

Department of Computer Engineering
Computer Project

Advisor: Assist. Prof. Dr. Mustafa Utku Kalay

Istanbul is known to be one of the most famous and crowded metropolitan cities in the world. Our city, which has been a home to many civilizations throughout its 2500-year history, has managed to withstand housing many people of different backgrounds throughout the years. With every new group of people passing through the city, new roads, streets and neighborhoods were built, and the reconstruction never stopped. For this reason in particular, the map has been evolving for the past 2000 years. Surely, the analysis of such a historically complex map would be just as difficult. In this project, we aimed do exactly that. With the help of our mentor, Associate Professor Utku Kalay, we generated a spatio-temporal database on QGIS and managed to upload that into PostgreSQL environment, with the aid of PostGIS tool. Then we applied certain optimization procedures to our data, in order to make things more efficient. With the scripts we have written here, we managed to add a time factor to the generated routes, considering the certain conditions of the streets. Lastly, we pulled our synthetic data back to PostgreSQL environment to make it 3-dimensional spatio-temporal data, apply various indexing methods and performance analysis. With the analysis we performed, we observed on many different factors, such as, the differences and the pros and cons of different indexing techniques, how they place themselves into the memory, and how PostgreSQL operates on using the indexes, then we made visual representations for our findings.

Keywords: Istanbul, road analysis, dataset generation, QGIS

1

Ön inceleme

Günümüzde bilgisayar bilimi, kullanılan araçlar ve diller konusundaki popüler konseptlerin çok hızlı değiştiği bir alan olarak bilinmektedir. Özellikle son yıllarda verinin analizi, saklanması, yönetilmesi ile ilgili meslekler yükselişe geçmiş bulunmaktadır. Ancak yine de günümüz yazılımcıları arasında yapılan bir ankette kendilerine tanındık gelen araçları kullanmaya daha eğilimli oldukları saptanmıştır. Yani popüler konseptler çabuk yükseliyor olsa bile, çoğu meslektaşımız, hala istenen sonuçlara ulaşmak için keşfedilen, var olan yöntemlerin daha gelişmiş halini kullanıcıya sunan yeni araçları tercih etmemektedir. Tam olarak da bu sebeple de uzamsal SQL'in gizli potansiyeli henüz çoğu yazılımcı tarafından keşfedilmemiş durumdadır. Geçtiğimiz birkaç yıl içerisinde uzamsal SQL'in sunduğu imkanlar sayesinde, geleneksel veri tabanı sorgulama yöntemleri ile çözemediğimiz problemleri çözebilen birçok proje ortaya konulmuştur. Bunlara örnek vermek gerekirse, Stockon Üniversitesi öğrencileri arasında yapılan bir yarışmada, potansiyel yaban mersini çiftlik konumları analizinden[1] çevresel ırkçılığın etkilerinin coğrafi dağılımına[2] kadar pek çok değişik konuya ilgili projeler çıkarılmıştır. Bu yarışma sayesinde uzamsal SQL alanının önemi bir kez daha göz önüne konulmuş oldu. Uzamsal SQL'in önemi her ne kadar göz ardı ediliyor olsa da son yıllarda bu konuda yardımcı olacak pek çok araç geliştirildi ve dağıtıma çıkarıldı. Biz de bu projemizde, araştırmamızın ihtiyaçları ve mentorümüzün yönlendirmeleri doğrultusunda PostgreSQL, PostGIS ve QGIS araçlarından faydalandık.

2 Fizibilite

Bu bölümde projenin teknik, iş ve zaman, yasal ve ekonomik açıdan fizibilite çalışmaları gerçekleştirilmiştir.

2.1 Teknik Fizibilite

Bu bölümde yazılım ve donanım fizibilitesi ayrıntılı olarak gerçekleştirilmiştir.

2.1.1 Yazılım Fizibilitesi

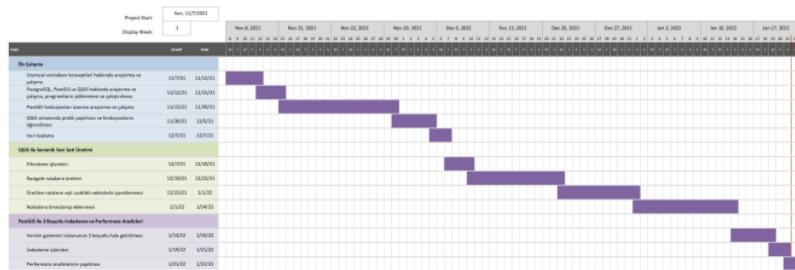
- **QGIS:** QGIS, coğrafi verilerin görüntülenmesini, düzenlenmesini ve analizini destekleyen ücretsiz ve açık kaynaklı 'cross platform' coğrafi bilgi sistemi (GIS) uygulamasıdır. Bu uygulama sayesinde OSM üzerinden indirilen veri setlerini inceleme, katmanlara göre analiz yapma, ham veriye göre sentetik veriseti üretme işlemleri rahatlıkla yapılır.
- **PostgreSQL:** Postgres olarak da bilinen PostgreSQL, genişletilebilirliği ve SQL uyumluluğunu vurgulayan ücretsiz ve açık kaynaklı bir ilişkisel veritabanı yönetim sistemidir.
- **POSTGIS:** POSTGIS, coğrafi nesneler için destek ekleyen açık kaynaklı bir PostgreSQL veritabanı uzantısıdır. Birçok geometri türlerini, mekansal ve uzaysal operatörleri, uzamsal ve jeo-uzamsal küme işlemlerini içerisinde barındırır ve zaman-uzamsal veriseti üretiminde en çok tercih edilen teknolojilerden bir tanesidir.
- **Python:** Python, yorumlanmış üst düzey genel amaçlı bir programlama dilidir.
Tasarım felsefesi, önemli girinti kullanımıyla kod okunabilirliğini vurgular. Dil yapıları ve nesne yönelimli yaklaşımı, programcılar küçük ve büyük ölçekli projeler için açık, mantıklı kodlar yazmasına yardımcı olmayı amaçlar.

- **Matplotlib:** Matplotlib, Python için bir cross platform, veri görselleştirme ve grafik çizim kütüphanesidir. Bu nedenle, birçok kullanılan açık kaynak olmayan uygulamalara uygun bir alternatif sunar. Projede Matplotlib, performans analizi sırasında kaydedilen sonuçları görselleştirmek için kullanıldı.

2.1.2 Donanım Fizibilitesi

Projenin geliştirilmesi için 2 bilgisayar kullanıldı. Birinci bilgisayar, Intel Core i7-1165G7 CPU, 16 GB RAM, Nvidia MX450 GPU ve 1 TB SSD, ikinci bilgisayar ise Intel Core i7-6700HQ CPU, 16 GB RAM, Nvidia GTX960M GPU ve 512 GB SSD'ye sahiptir. Bu özellikler geliştirme yapabilmek için yeterli bir donanım sağlamaktadır.

2.1.3 İş ve Zaman Fizibilitesi



Şekil 2.1 Gantt Diyagramı

2.1.4 Yasal Fizibilite

Kullandığımız yazılımlar ve araçlar açık kaynak kodludur. Bu yazılımlar ve araçların ticari kullanım dışında herhangi bir kısıtlaması yoktur. Projemiz ticari amaçlı olmadığından kullandığımız tüm yazılım ve araçlar yasal olarak sorun teşkil etmemektedir.

2.1.5 Ekonomik Fizibilite

Kullandığımız yazılımlar ve araçlar ücretsizdir. Geliştirme yaptığımız cihazlar kişisel cihazlarımız olduğundan altyapı konusunda da mali bir yükümlülüğümüz bulunmamaktadır.

3 **Sistem Analizi**

Projenin ilk kısmında QGIS üzerinden İstanbul haritasının karayolları verisi üzerinden rastgele yerleştirilmiş noktalar arasında rotalar üretilmiştir. İleride performans analizleri yapılrken fark görülebilmesi için verinin büyük olması amaçlanmıştır. Bu rotalar üzerinde eşit uzaklıkta noktalar yerleştirilip, elde edilen noktalar tablosuna zaman boyutu eklenmiştir. Eklenen bu ‘timestamp’ sütununda ilk adımda her noktanın arasında eşit zaman uzaklığı vardır. Projedeği gerçekçiliği artırabilmek için noktaların bulunduğu yol üzerinde dönüş açıları hesaplanarak, belli bir açıdan büyük değişimlere rastlandığında zaman bilgisine gecikme eklenir. Rotalar ve noktalar, İstanbul yolları ile beraber görsel bir sunum yapılabilmesi için İstanbul'un binalar, su yolları, ray hatları gibi diğer uzamsal veri tabloları ile birlikte gösterilir.

İkinci kısmında ise üzerinde işlemler yapılmış veri üzerinden yeni üretilen rotalar ve noktalar tabloları PostGIS'e geri gönderilecek ve üzerinde 3 farklı index tipinden farklı veri boyutları üzerinde performans analizleri yapılacaktır. Bu indeks tiplerinin aralarındaki farklar faydalari ve zararları üzerinden açıklanacak, PostgreSQL'in bunları hangi şartlar altında nasıl kullanmayı tercih ettiğinden bahsedilecektir. Indexlerin veriye yerleşme ve sorgulara cevap getirme süreleri arasında yapılan karşılaştırmalardan alınan sonuçlar da bu kısımda görselleştirilmiştir.

4 Sistem Tasarımı

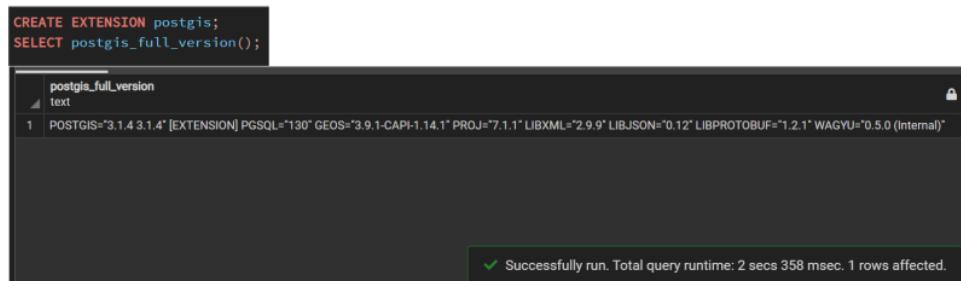
4.1 Ön Hazırlık

Bu başlık altında projenin başında gerekli olan hazırlıklar kısaca anlatılıp incelenecektir.

4.1.1 Haritanın bulunması ve PgAdmin 4 ortamında PostGIS'in bağlanması

Projede kullanılan haritanın ilk başta 'OpenStreetMap' üzerinden çekilmesi planlandı. Ancak, çekeceğimiz harita verisi boyut olarak 'OpenStreetMap'in, kendi websitesinin belirlediği üst sınırdan daha büyük olduğu için bu yöntemden vazgeçildi ve bu tip senaryolar için alternatif olarak kullanılan OSM tabanlı BBBike API'sı kullanıldı. Yukarıda belirtilen her iki kaynağın ücretsiz hizmet verdiğiin de belirtilmesi gerekir. Çekilen veri içerisinde .shp formatında 'buildings', 'landuse', 'natural', 'places', 'points', 'railways', 'waterways' ve 'roads' olmak üzere 8 farklı tablo vardır. İşlemler 'roads' tablosu üzerinde yapıldı ve diğer shp dosyaları QGIS ortamında görselliği artırmak için kullanıldı.

İlk adımda, PgAdmin4 ortamında, PostgreSQL kullanılarak boş bir veritabanı açıldı ve PostGIS uzantısı bu veritabanına bağlandı. PostGIS veritabanına bağlandıktan sonra otomatik olarak bir "spatial_ref_sys" tablosu oluşturur (Şekil 4.1).



The screenshot shows a PostgreSQL query window in PgAdmin 4. The SQL command is:

```
CREATE EXTENSION postgis;
SELECT postgis_full_version();
```

The results pane displays the output of the `postgis_full_version()` function:

postgis_full_version
text
1 POSTGIS="3.1.4 3.1.4" [EXTENSION] PGSQL="130" GEOS="3.9.1-CAPI-1.14.1" PROJ="7.1.1" LIBXML="2.9.9" LIBJSON="0.12" LIBPROTOBUF="1.2.1" WAGYU="0.5.0 (internal)"

A green success message at the bottom right of the interface reads: "Successfully run. Total query runtime: 2 secs 358 msec. 1 rows affected."

Şekil 4.1 PostGIS uzantısının bağlanması

Ardından, geometri çeşitlerini temsil etmesi için bir "geometries" tablosu oluşturuldu. Bu tablo sayesinde, projenin ilerleyen aşamalarında kullanılabilecek fonksiyonlara ortam hazırlanmış oldu. (Şekil 4.2).

```

Query Editor  Query History
1 CREATE TABLE geometries (name varchar, geom geometry);
2
3 INSERT INTO geometries VALUES
4 ('Point', 'POINT(0 0)'),
5 ('Linestring', 'LINESTRING(0 0, 1 1, 2 1, 2 2)'),
6 ('Polygon', 'POLYGON((0 0, 1 0, 1 1, 0 1, 0 0))'),
7 ('PolygonWithHole', 'POLYGON((0 0, 10 0, 10 10, 0 10, 0 0),(1 1, 1 2, 2 2, 2 1, 1 1))'),
8 ('Collection', 'GEOMETRYCOLLECTION(POINT(2 0),POLYGON((0 0, 1 0, 1 1, 0 1, 0 0)))');
9
10 SELECT name, ST_AsText(geom) FROM geometries;

```

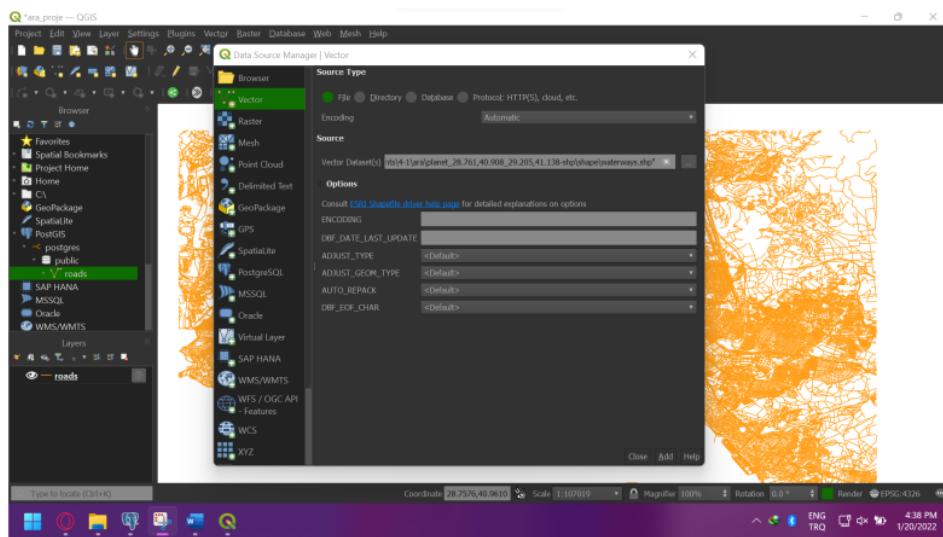
Data Output		Explain	Messages	Notifications
name	st_astext			
character varying	text			
1 Point	POINT(0 0)			
2 Linestring	LINESTRING(0 0,1 1,2 1,2 2)			
3 Polygon	POLYGON((0 0,1 0,1 1,0 1,0 0))			
4 PolygonWithHole	POLYGON((0 0,10 0,10 10,0 10,0 0),(1 1,1 2,2 2,2 1,1 1))			
5 Collection	GEOMETRYCOLLECTION(POINT(2 0),POLYGON((0 0,1 0,1 1,0 1,0 0)))			

Şekil 4.2 Geometri tablosunun oluşturulması

4.1.2 QGIS Ortamı ile Bağlantı Kurulması ve Diğer Shapefile'ların Eklenmesi

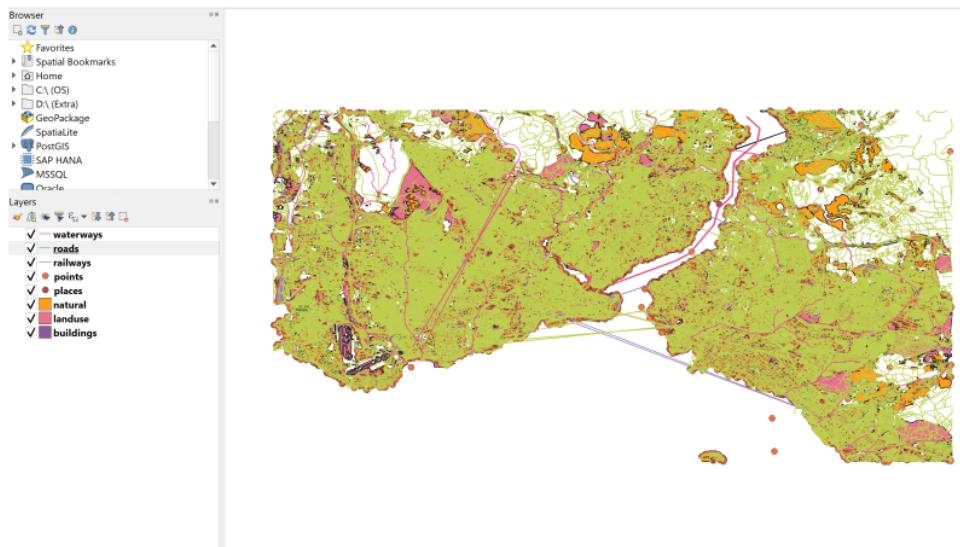
Verinin işlenebilmesi için PostGIS ile QGIS arasında bağlantı kuruldu. Bunun için QGIS'te yeni bir proje açılıp PostGIS'e kendi kullanıcı ismimiz üzerinden bağlantı sağlandı. İlk başta, önceden bulunan ham veri QGIS ortamına eklenecek ve sentetik veri üretildikten sonra PostGIS ortamına bağlanacak.

Çekilen ham İstanbul verisinin içerisinde, öncesinde de belirtildiği gibi, veriler 'vector layer' olarak tutuluyor. BU yüzden, QGIS ortamına bu Shapefile'ların eklenmesi için 'Vector Bundle' kullanılması gereklidir.(Şekil 4.3).



Şekil 4.3 QGIS'e, indirilen .shp uzantılı dosyaların aktarılması

Görsellik sağlama için OSM'den çekilen verinin diğer tabloları da buraya eklendi. Haritanın QGIS'te, henüz işlenmemiş hali Şekil 4.4'de gösterildiği gibidir.



Şekil 4.4 Ham verinin QGIS ortamında görüntülenmesi

5

Uygulama

Bu kısımda, sentetik verinin nasıl üretildiği, indekslendiği gibi önemli noktalar belirtilecektir.

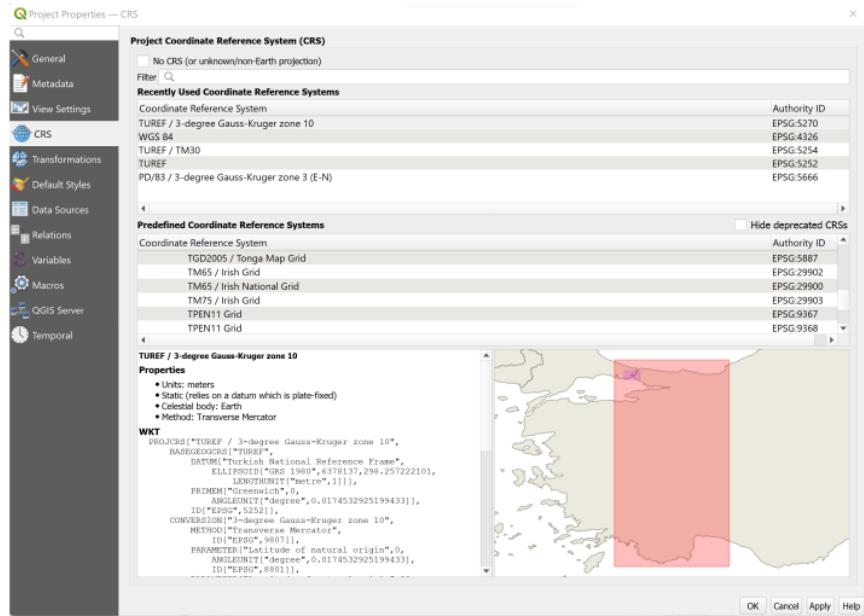
5.1 QGIS'te Sentetik Veri Üretimi

5.1.1 CRS formatının ayarlanması

Şimdiye kadar, İstanbul haritası verisinin internetten nasıl çekilib QGIS ortamına eklendiği konuşuldu. Bu başlık altında, sentetik verinin üretimi adım adım anlatılacaktır.

İlk başta, indirilen Shapefile dosyaları içerisindeki 'roads' katmanını incelendiğinde zaman CRS formatının WGS 84'e sabitlendiğini söylebilir. QGIS yüklenen her projede otomatik olarak 'Global Projection Specification' olan WGS 84 modunda açmak için ayarlanmıştır [3]. Yapılan proje kıtasal ya da daha büyük bir harita üzerinde işleniyor olsaydı, CRS farkı görmezden gelmek mümkün idi.

Fakat, bu projenin kapsamında İstanbul haritası gibi daha küçük bir alanda çalışıldığı için, projeksiyon farkının göz önünde bulundurulması şarttır. Buna ek olarak, WGS 84 CRS formatında projeyi çalıştırılırsa, katmanlar üzerinde ölçü birimi derece olacaktır, bu da projenin ilerleyen adımlarında belirli zorluklar oluşturabilir. Bu kapsamda, projenin CRS formatı 'TUREF / 3-degree Gauss-Kruger zone 10'a değiştirilmiştir. İlaveten, önceden eklenen her katmanın da belirtilen CRS formatına getirilmesi gerekmektedir. Şekil 5.1'de de görüldüğü gibi, hem ölçü birimi metre olarak sabitlenmiştir hem de harita üzerinde İstanbul'un içerisindeki kesitin alınmasına dikkat edilmiştir. Bu sayede, ölçümler daha net yapılacaktır.



Şekil 5.1 Projenin CRS formatı değiştirildikten sonraki görüntüsü

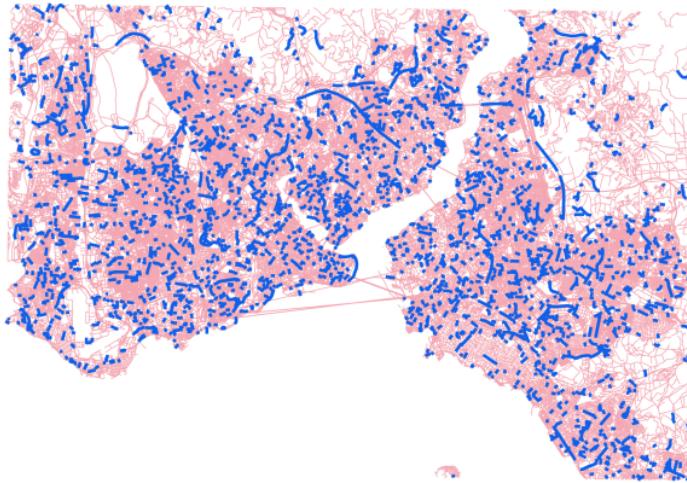
5.1.2 Rota Üretimi

Projemin formatı ayarlandıktan sonra, veri üretme işlemleri başlatıldı. Rota üretimi senaryosu olarak İstanbulda bulunan 88 binden fazla sokaktan ilk olarak 2500 tanesinin seçilmesi ve ayrı bir katman olarak kaydedilmesi kararlaştırıldı. Bunun uygulanması için QGIS'te, 'Processing Toolbox'ta yer alan 'Random Extract' algoritması kullanıldı. Bu algoritma bir vektör katmanı alır ve girdi katmanındaki özelliklerin yalnızca bir alt kümesini içeren yeni bir katman oluşturur. Alt küme, alt kümedeki toplam özellik sayısını tanımlamak için bir yüzde veya sayı değeri kullanılarak rasgele tanımlanır [4]. Bu kurallar esasında, önceden belirtildiği gibi 2500 tane sokak rasgele seçilip yeni bir katman olarak tanımlandı (Şekil 5.2).

5.1.3 Filtreleme İşlemleri

Rasgele üretilen rotaların devamında, sentetik veriseti de dahil olmak üzere, diğer tüm katmanlarda hiç kullanılmayacak, ve/veya değerleri NULL olan kolonlar bulunuyordu. Bu kolonlar, 'Attributes Table' penceresinden uygulanan işlemle silindi.

Ek olarak, tahmin edilebileceği üzere, İstanbulda çok kısa (80 metrenin altında) sokaklar da bulunuyor. Bunlar üzerinde işlem yapmak projenin amacına hizmet etmeyecektir. O yüzden, küçük bir filtreleme ile uzunluğu 80 metrenin altında olan tüm sokaklar üretilen verisetinden temizlendi.



Şekil 5.2 Rastgele seçilmiş 2500 tane sokak.

Sonuç olarak, Şekil 5.3'de gösterilen sentetik bir veriseti elde edildi. Belirtilmesi gereklidir ki, Bazı seçilmiş sokak isimlerinin NULL değerde olması projenin sonuçlarına etki etmemektedir.

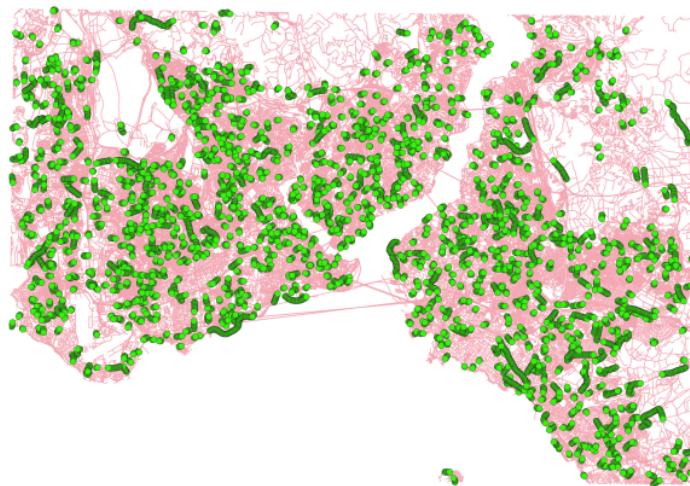
— Features Total: 1432, Filtered: 1432, Selected: 0

	osm_id	name	type
289	88498282	Yan Sokağı	residential
290	89169548	Erman Sokağı	residential
291	89375430	NULL	tertiary
292	89375456	85/7. Sokak	residential
293	90929400	87. Sokak	residential
294	90929435	Beşkardeşler 3. Sokak	residential
295	91140255	8. Bağ Sokağı	residential
296	91227535	Haseki Caddesi	tertiary
297	91233353	Vezir Odaları Sokağı	residential
298	91310354	NULL	footway
299	91353644	Karabulut Sokağı	residential
300	91353655	Türkmen Sokağı	residential
301	91353668	Kuru Çınar Sokağı	residential
302	91387265	Başhoca Sokağı	residential
303	91387287	Hattat Nafiz Caddesi	residential
304	91387316	Kıztaşı Caddesi	residential
305	91397283	Cibali Şerefiye Sokağı	residential
306	91397292	Şair Bakı Sokağı	residential

Şekil 5.3 80 metre ve üstü, rastgele seçilmiş sokakların bir kısmı

5.1.4 Nokta üretimi

Her üretilmiş rotanın üzerinde bir aracın harekete hazır olduğunu düşünelim. Eğer, rotaların hepsinin başından sonuna, her 40 metreden bir nokta üretilirse, bu noktalar, o sokak üzerinde hareket eden aracın gittiği yolu temsil eder. Bu noktaları üretmek için Processing Toolbox'ta bulunan 'Points along geometry' algoritmasını kullandık. Bu algoritma, bir girdi vektör katmanının çizgileri boyunca dağıtılan noktalarla bir nokta katmanı oluşturur. Çizgi boyunca olan noktalar arasındaki mesafe parametre olarak tanımlanır. Tahmin edileceği üzere, noktalar önceden üretilen rotalar üzerinde, 40 metreden bir oluşturulmuştur. Sonuç olarak, Şekil 5.4'teki gibi bir görüntü elde edilmiştir.



Şekil 5.4 Önceden rasgele seçilmiş rotalar üzerinde, 40 metreden bir üretilmiş noktalar

5.1.5 Üretilmiş noktalara zaman etiketi atanması

Tüm bu işlemlere ek olarak, bütün araçların belirli bir hızla gittiğini varsayıyalım. Hızlarını 20 m/s, yani 72 km/saat olduğunu düşünürsek eğer, araçlar bir noktadan sonrakine iki saniyeye varmış olurlar. Bu mantıkla yola çıktıığında, aracın o noktaya kaç saniyede geldiğini gösteren bir zaman etiketi her noktaya atanabilir.

Fakat, zaman etiketi üretmeden önce, proje katmanlarının shapefile formatından çıkarılması gereklidir. Çünkü, Shapefile formatında yeni bir kolon eklemeye çalışılırsa, veritiplerinin kısıtlı sayıda olduğu farkedilir. Örnek olarak, zaman etiketi eklemek için veri tipinin 'Time' veya 'DateTime' olması gereklidir. Lakin, Shapefile formatında olan katmanların zamanla ilgili sadece 'Date' veri tipi vardır, ve zaman etiketinin atanmasında doğru sonuçlar döndürülmesine engel olur. Çünkü, üretilcek olan

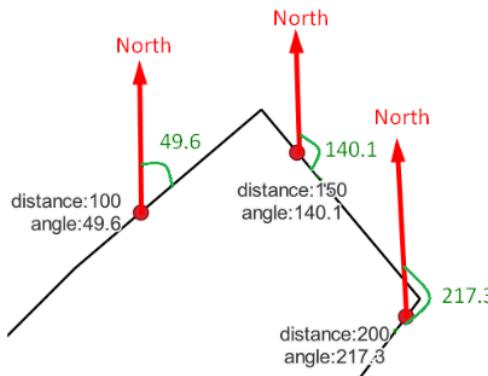
zaman etiketi saniye hassaslığında olmalıdır, 'Date' veri tipinde ise bu mümkün değildir. Bu yüzden katmanların 'Geopackage' formatına ayarlanması zorunludur. Şöyle ki, Geopackage formatlı katmanlarda 'DateTime' veri tipi vardır ve zaman etiketi atanmasında elverişli ortam hazırlar.

Belirtilen şartlarda bir zaman etiketi (timestamp) eklemek için önceden üretilmiş noktaların 'Attributes Table'ında bulunan 'Field Calculator' sayesinde oluşturulabilir. 'Field Calculator' her katmanın verisini gösteren tabloda bulunan, sentetik veri üretmeye en çok yaranan, içerisine kod yazılabilen QGIS özelliklerinden birtanesidir. Expression alanına, SQL'e benzeyen, ama kendine has fonksiyonlarını ve syntaxını kullanarak algoritma geliştirilebilir. Bu özellikten faydalananarak, kod geliştirilmiş ve zaman etiketi noktalara atanmıştır.

5.1.5.1 Yoldaki dönüşlerde aracın hızının azalması

Şimdiye kadar rasgele seçilmiş rotalar üzerinde belirli aralıklarla noktalar üretildi ve o noktalara zaman etiketi atandı. Lakin, atanan zaman etiketleri, yolu koşullarına bakılmaksızın, her belirli aralığı belirli hızla gidiyor. Başka bir deyişle, araçlar her zaman yollarda sabit hızla ilerlemiyorlar. Mesela, eğer yolda belirli bir düzeye dönüş varsa, aracın hızının azalması gereklidir.

Bu kapsamda, biz de projemizde dönüslü olan yollarda aracın hızını belirli bir düzeye azalttık. Yolu dönüslü olup olmadığını anlamak için üretilen noktalar, üzerinde bulundukları 'linestring'in o andaki azimut (çizginin Kuzey istikametine göre) açısını 'Attributes Table'da tutuyorlar (Şekil 5.5). Bu bilgiden yola çıkararak, her nokta bu açı değerini tuttuğu için, bir rota üzerinde iki ardışık noktaların açıları arasındaki farkı bulursak yolu ne kadar dönüslü olduğunu tahmin etmiş oluruz. Şöyle ki, eğer iki nokta arasındaki açı farkı 40 ve üzeri ise, aracın hızının ciddi bir şekilde (yarı yarıya) azalması gerekdir. Eğer sabit bir mesafede hız azalıyorsa, o yolu katetmeye sarfedilen zaman artar. Dolayısıyla, eğer belirlenen sınırın üzerinde açı farkı var ise, aracın o anda bulunduğu nokta üzerinde önceden tanımlanan zaman etiketi güncellenecektir. Bu durumun algoritması öncesinde olduğu gibi, 'Attributes Table' içinde bulunan 'Field Calculator' kullanılarak hesaplanmıştır. Sonuç olarak, geliştirdiğimiz kod sayesinde dönüş durumu kontrol edilip zaman etiketi güncellenmiştir. Şekil 5.6'da, kırmızı karelerle gösterilmiş iki nokta arasında açı farkının 40'in üzerinde olduğu, ve dolayısıyla da o iki nokta arasındaki mesafeyi daha uzun sürede gittiği gözlemlenebilir.



Şekil 5.5 Noktanın bulunduğu çizgi üzerinde açısal değerinin bulunması

LAST_POINTS — Features Total: 10100, Filtered: 10100, Selected: 0						
	fid	osm_id	name	type	distance	angle
141	141	25302628 Yusuf Aşkın Sokagi		residential	120	40.50520337299...
142	142	25428552 Kutlugin Sokağı		residential	0	44.99005214101...
143	143	25428552 Kutlugin Sokağı		residential	40	44.99005214101...
144	144	25428552 Kutlugin Sokağı		residential	80	43.67973869239...
145	145	25428552 Kutlugin Sokağı		residential	120	42.40309815715...
146	146	25428552 Kutlugin Sokağı		residential	160	42.40309815715...
147	147	25512790 Sultanahmet Meydanı		pedestrian	0	210.7327168770...
148	148	25512790 Sultanahmet Meydanı		pedestrian	40	93.65109303673...
149	149	25512790 Sultanahmet Meydanı		pedestrian	80	38.26180266148...
150	150	25512790 Sultanahmet Meydanı		pedestrian	120	38.28352270393...
151	151	25512790 Sultanahmet Meydanı		pedestrian	160	38.26790628887...
152	152	25512790 Sultanahmet Meydanı		pedestrian	200	38.26800677600...
153	153	25512790 Sultanahmet Meydanı		pedestrian	240	38.27046049670...
154	154	25512790 Sultanahmet Meydanı		pedestrian	280	38.27571488113...
155	155	25512790 Sultanahmet Meydanı		pedestrian	320	38.26786881498...
156	156	25512790 Sultanahmet Meydanı		pedestrian	360	38.27109459554...
157	157	25512790 Sultanahmet Meydanı		pedestrian	400	38.27109459554...
158	158	25512790 Sultanahmet Meydanı		pedestrian	440	281.2740163432...

Şekil 5.6 Noktaların özelliklerini gösteren veriseti

5.2 PostGIS'te 'geom' Sütununa Zaman Boyutunun Eklenmesi

Bir önceki kısımda yapılan işlemlerin sonucunda QGIS'te verinin işlenmesi tamamlanmıştır. Sıradaki adımda veri üzerine yapılacak ayarlamalar için PostGIS fonksiyonlarından faydalandırıldı.

Öncelikle QGIS tarafında 'roads' tablosundan türetilerek elde edilen ve orada işlemlerden geçirilmiş yeni iki veri seti PostGIS'e eklendi. Bu durumda PostGIS tarafından 2 tablo bulunur, bunlar yeni elde ettiğimiz LAST_POINTS ile LAST_ROUTES'tır.

Öncelikle LAST_POINTS tablosunun (Şekil 5.7) yerleşme şekli incelendiğinde şu çıkarımlar yapıldı:

Data Output							Explain			Messages			Notifications		
	fid	geom	osm_id	name	type	distance	angle	timestamp	rowid	lock	table	rowcount	estimated cost	estimated rows	
1	1	01010000209614000058ABC0620ADD634153D66AD56E535141	4885644	Bab-I Hümayun C...	residential	0	42.78758496576681	2022-01-25 20:00...							
2	2	01010000209614000001E43BC8000D6341AB83242C76535141	4885644	Bab-I Hümayun C...	residential	40	42.78758496576681	2022-01-25 20:00...							
3	3	010100002096140000866ED83A11D06341F017FA697D535141	4885644	Bab-I Hümayun C...	residential	80	45.43740050975379	2022-01-25 20:00...							
4	4	010100002096140000E4388CA14D06341746BAC6E84535141	4885644	Bab-I Hümayun C...	residential	120	45.43740050975379	2022-01-25 20:00...							
5	5	01010000209614000037AB854018D063413CD7F9A388E535141	4885644	Bab-I Hümayun C...	residential	160	42.00650627419994	2022-01-25 20:00...							
6	6	0101000020961400006C484DEA9FE2634185FC061AA05551...	22875933	[null]	motorway_link	0	128.1578340971945	2022-01-25 20:00...							
7	7	010100002096140000F9FC708C3E26341ACFD21ED6A555141	22875933	[null]	motorway_link	40	128.1578340971945	2022-01-25 20:00...							
8	8	010100002096140000F5CDC06C8E263413BFF9C78A1555141	22875933	[null]	motorway_link	80	117.1692654027964	2022-01-25 20:00...							
9	9	01010000209614000086339A1CE26341CAEB0FA790555141	22875933	[null]	motorway_link	120	109.31827847170466	2022-01-25 20:00...							
10	10	010100002096140000131267D1E2634190F7124098555141	22875933	[null]	motorway_link	160	99.20571961417623	2022-01-25 20:00...							
11	11	0101000020961400005C9650551E3634151BD15E92555141	22875933	[null]	motorway_link	0	288.8957699029531	2022-01-25 20:00...							
12	12	01010000209614000062824B9A11E363416CAB22686555141	22875933	[null]	motorway_link	40	288.8957699029531	2022-01-25 20:00...							
13	13	010100002096140000676C46DF0CE3634187092F6389555141	22875933	[null]	motorway_link	80	288.8957699029531	2022-01-25 20:00...							
14	14	0101000020961400004EBABF4008E363415998ED2FD8555141	22875933	[null]	motorway_link	120	294.37163063748744	2022-01-25 20:00...							

Şekil 5.7 LAST_POINTS tablosundaki değerlerin yerleşme şekli

1. Her osm_id değeri bir rotayı temsil eder. Aynı rota üzerindeki noktaların osm_id değerleri aynıdır.
2. Her noktanın kendine özgü bir fid değeri vardır.
3. name, type, distance ve angle bu adımda önemsizdir, daha önce gerekli işlemler için kullanıldılar.
4. timestamp değeri ile geom değeri ayrı sütunlarda bulunmaktadır. Yani geom değerine 3. bir boyut eklenmesine ihtiyaç vardır.

Sonraki adımda veriye 3 boyutlu indeksleme yapılacağı için bu adımda geom ve timestamp sütunlarını birleştirecek işlemleri yapmayı amaçladık. Bu işlem için PostGIS fonksiyonlarından faydalandık. X, Y ve M koordinatlarını argüman olarak Kabul ederek yeni nokta yaratan ST_MakePointM fonksiyonunu kullanmayı seçtik [5].

Bahsedilen fonksiyon argümanlarının hepsini numeric cinsinden kabul ettiği için öncelikle gerekli değerleri doğru formatlara getirdik. X ve Y değerleri ST_X(geom) ve ST_Y(geom) sayesinde çevrilebilir(Şekil 5.8).

Data Output Explain Messages Notifications						
	fid [PK] bigint	st_x double precision	st_y double precision	osm_id bigint	name character varying (48)	
1	1	10414163.086019203	4541883.334645825	4885644	Bab-I Hümayun Caddesi	
2	2	10414190.257311257	4541912.689728658	4885644	Bab-I Hümayun Caddesi	
3	3	10414217.829764616	4541941.655889496	4885644	Bab-I Hümayun Caddesi	
4	4	10414246.329133939	4541969.723414291	4885644	Bab-I Hümayun Caddesi	
5	5	10414274.022174938	4541998.562074479	4885644	Bab-I Hümayun Caddesi	
6	6	10425855.321935378	4544180.418395211	22875933	[null]	
7	7	10425886.774406897	4544155.70519964	22875933	[null]	

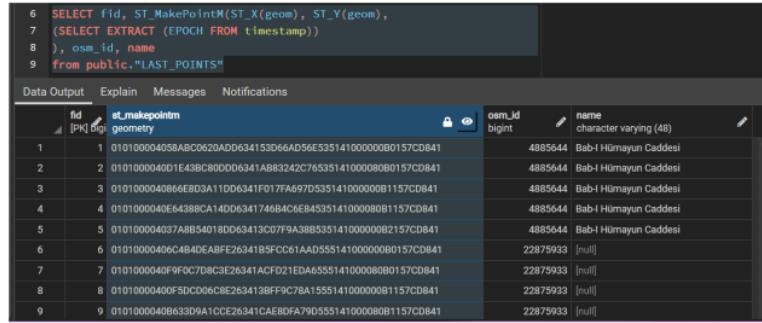
Şekil 5.8 LAST_POINTS tablosunda ST_X(geom) ve ST_Y(geom) değerlerinin gösterimi

Kullanılan diğer SQL fonksiyonu SELECT EXTRACT'tır(Şekil 5.9).

Data Output Explain Messages Notifications	
	date_part double precision
1	1643140800
2	1643140802
3	1643140804
4	1643140806
5	1643140808
6	1643140800
7	1643140802

Şekil 5.9 SELECT EXTRACT fonksiyonunun test edilmesi

Bu fonksiyon sayesinde timestamp değerleri double precision tipine çevrildi. Daha sonra elde edilen tüm değerler ST_MakePointM fonksiyonunda bir araya getirildi (Şekil 5.10).



```

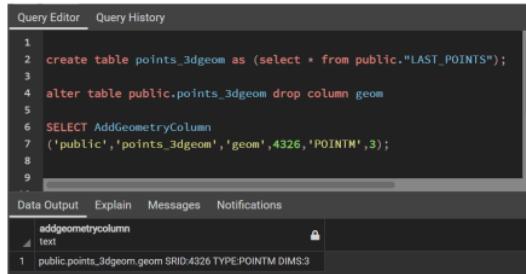
6   SELECT fid, ST_MakePointM(ST_X(geom), ST_Y(geom),
7   (SELECT EXTRACT (EPOCH FROM timestamp)
8   ), osm_id, name
9   from public."LAST_POINTS"

```

	osm_id	name
1	4085644	Bab-I Hümayun Caddesi
2	4085644	Bab-I Hümayun Caddesi
3	4085644	Bab-I Hümayun Caddesi
4	4085644	Bab-I Hümayun Caddesi
5	4085644	Bab-I Hümayun Caddesi
6	22875933	[null]
7	22875933	[null]
8	22875933	[null]
9	22875933	[null]

Şekil 5.10 ST_MakePointM fonksiyonunun test edilmesi

Yeni elde edilen geom değerleri LAST_POINTS tablosunun gerçek geom değerleriyle karşılaştırıldığında işlemde başarılı olunduğu görüldü. Sonraki adımda bu sorgunun sonucu olarak gelen tablo indexleme ve analizler için yeni bir tabloya insert edildi. Bu işlemin sebebi daha önce de belirtildiği gibi indexlemeyi 3 boyutlu geom değeri üzerinden yapmaktadır. Burada da eski tabloyu bozmamak için 3 boyutlu geomları tutacak yeni bir tablo oluşturuldu. Sorgunun sonucu yeni tabloya eklenip orada indexlenmesi amaçlandı. Bu sebeple tablonun bütün sütunları aynen yeni tabloya kopyalandı. Ancak eski geom sütunu POINT türünden 2 boyutlu olduğu için, ve ihtiyaç duyulan sütun POINTM tipinden 3 boyutlu olduğu için resimde görülen soru ile önce var olan geom sütunu silindi, daha sonra ihtiyaca uygun yeni bir sütun tanımlandı. Bu işlem için PostGIS'in AddGeometryColumn fonksiyonu kullanıldı (Şekil 5.11).



```

1
2 create table points_3dgeom as (select * from public."LAST_POINTS");
3
4 alter table public.points_3dgeom drop column geom
5
6 SELECT AddGeometryColumn
7 ('public','points_3dgeom','geom',4326,'POINTM',3);
8
9

```

	text
1	public.points_3dgeom.geom SRID:4326 TYPE:POINTM DIMS:3

Şekil 5.11 AddGeometryColumn fonksiyonunun çalıştırılması

Artık tüm kayıtlar yeni geomlar ile birlikte points_3dgeom tablosuna aktarılabilir durumdadır (Şekil 5.12).

```
10 insert into points_3dgeom (
11 select fid, osm_id, name, type, distance, angle, timestamp,
12     ST_MakePointM(ST_X(geom), ST_Y(geom),
13     (SELECT EXTRACT (EPOCH FROM timestamp))
14 )
15 from public."LAST_POINTS")
```

Data Output Explain Messages Notifications

INSERT 0 10100

Query returned successfully in 58 msec.

Şekil 5.12 insert into ile yeni tabloya kayıtların eklenmesi

İki tablo karşılaştırıldığında geom değerleri arasındaki farklar görülebilir (Şekil 5.13).

17 select fid, name, ST_AsText(geom) from points_3dgeom			
Data Output		Explain	Messages
fid	name	st_astext	
1	Bab-I Hümeyun Caddesi	POINT M (10414163.086019203 4541883.334645825 1643140800)	
2	Bab-I Hümeyun Caddesi	POINT M (10414190.257311257 4541912.689728658 1643140802)	
3	Bab-I Hümeyun Caddesi	POINT M (10414217.829764616 4541941.655889496 1643140804)	
4	Bab-I Hümeyun Caddesi	POINT M (10414246.329133939 4541969.723414291 1643140806)	
5	Bab-I Hümeyun Caddesi	POINT M (10414274.022174938 4541998.562074479 1643140808)	
6	[null]	POINT M (10425855.321935378 4544180.418395211 1643140800)	
7	[null]	POINT M (10425886.774406897 4544155.70519964 1643140802)	
8	[null]	POINT M (10425920.212568788 4544133.884582336 1643140804)	
9	[null]	POINT M (10425957.057763916 4544118.623041341 1643140806)	

Şekil 5.13 Yeni tablonun son hali

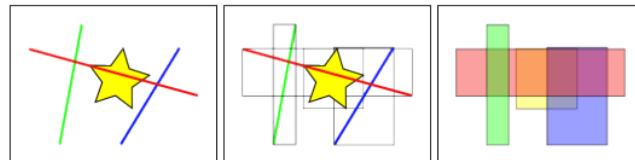
Projede veriye uygulamamız gereken tüm işlemler tamamlandı. Bu adımdan sonra indexlemeler ve performans analizleri yapıldı.

6

Deneysel Sonuçlar

6.1 Indexleme İşleminin Açıklaması ve Kullanılma Sebepleri

Verilerin indexlenmesi işlemi, özellikle büyük yer kaplayan veri tabanlarında sorguların yapılabilmesi için büyük fayda sağlar. Indexleme yapılmaması, her sorguda verilerin lineer aramadan geçirilmesi gerekiirdi. Normal veri tabanlarında indeksleme işlemleri, bu iş için seçilen sütunun değerleri üzerinden bir ağaç veri yapısına yerleştirilerek yapılır. Ancak uzamsal veri tabanlarında böyle bir uygulama yapmak mümkün değildir. Geometrik özellikler indekslenebilir değerler olmadığı için onların yerine genellikle şekilde belirtildiği gibi sınırlayıcı kutular oluşturulur ve daha sonra elde edilen bu sınırlayıcı kutular uzamsal veri yapılarına uygun veri yapılarına yerleştirilir(Şekil 6.1)[6].



Şekil 6.1 Bounding Boxların yerleşimi

Değişik ihtiyaçlara uyum sağlayabilmek için farklı veri yapıları ile verilerini yerleştiren index tipleri üretilmiştir. PostGIS'in geometri türüne uyum sağlayan ve günümüzde hala uygulamalar tarafından desteklenmekte olan 10 farklı tür index vardır.

Bunlardan ilk ikisi olan 'btree' ve 'hash' indexleri geometrik sütunlarda özel durumları olan indexlemeler için hazırlanmıştır. 'btree' indexi sıralanabilir olan geometri türlerine, 'hash' indexi her satırın unique olduğu bilinen geometri türlerine özeldir. Ayrıca kullanıcının bu iki index tipini ekleyip kaldırmak PostgreSQL'in son kullanıcıya verdiği izinler arasında değildir. Bu sebeple projede bu indexlerin vereceği sonuçlar analizlere katılmamıştır. Diğer indexler temelde 3 grupta toplanabilir: bunlar GIST, BRIN ve SPGIST'tir. Indexledikleri verilerin boyutlarına göre 2 boyutlu GIST indexleri, 3 boyutlu GIST indexleri gibi alt gruplara da ayrılırlar. Projemizde kullandığımız veri

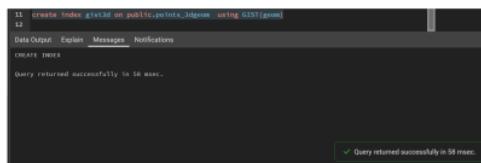
3 boyutlu ve POINTM tipinde olduğu için sadece 3 boyutlu tipler kullanıldı [7].

Analizler buraya kadar yapılan işlemlerin sonucunda en son adımda erişilmiş olan points_3dgeom tablosu üzerinden yapılacaktır. Tabloda 10bin civarında satır vardır. Verinin bu büyülükte seçilmesinin sebebi projede kullanılan bilgisayarların CPU güçleri sebebiyle olan bir kısıtlamadır. Bu nedenle alınan sonuçlardaki ufak değişikliklerin büyük farklar anlamına geldiğinin belirtilmesi gereklidir. Ayrıca bazı indexleme tiplerinde algoritmalar oluşturulan sınırlayıcı kutular arasında kesişmeyi minimuma indirmek üzere tasarlanmıştır. Bazı program türlerinde belli bir index çeşidinin tercih edilme sebebi bu olabilir. Ancak projemizde indexleme yaptığımız tablo sadece noktalardanoluştuğu ve bu noktalar oluşturulurken kesişme olmayacak şekilde ayarlandıkları bilindiği için bu çalışmada bu faktörler göz önüne alınmayacağı.

Daha önce belirtildiği gibi analizlerde üç tip index kullanılmıştır. Indexler hakkında zaten bilinen belli özelliklerden kısaca bahsedilecektir, daha sonra performans analizlerinde bu bekleneler incelenecaktır. Indexlerin tabloya yerleşme sürelerinden de bahsedilecektir

6.1.1 GIST

GIST metodu r-tree veri yapısını implement eden index tipidir. G harfi ‘generalized’ anlamına gelir, GIST metodu da en genel ihtiyaçlara cevap vermek için tasarlanmıştır, bu nedenle özelleştirilmiş tablolarda yeterince verimli olmadığı bilinmektedir. Bu yüzden en çok uniform dağılmış veri tiplerine uygun olduğu söylenebilir. Bizim analizlerimizde kullanacağımız veri uniform dağılımdadır. Indexin build süresi şekil6.2deki gibidir.



The screenshot shows a PostgreSQL terminal window. The command entered is:

```
11 Create index gixtd on public.points_3dgeom using GIST (geom)
```

Below the command, the status bar displays:

```
Data Output Explain Messages Notifications
```

Underneath the status bar, there is a message:

```
CREATE INDEX
```

At the bottom of the terminal, there is a message indicating the query was successful:

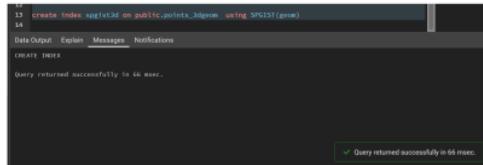
```
Query returned successfully in 58 msec.
```

Şekil 6.2 Create Index Using GIST sorgusu

6.1.2 SPGIST

SPGIST metodu, GIST metodundan yola çıkılarak dizayn edilmiştir ve onun kadar genelleyici özelliklere sahip bir index tipidir. Ancak GIST metodundan farklı olarak üst üste gelen bounding boxlar arası karışıklıkları çözmeye odaklanmıştır. Quad-tree

veri yapısı kullanır. Projemizde kullandığımız veri uniform dağılmış ve çakışmadığı bilinen noktalardan oluşmaktadır. İndexin build süresi şekil 6.3deki gibidir.



A screenshot of a PostgreSQL terminal window. The command entered is:

```
13 create index spgist3d on public.points_3dgeom using SPGIST (geom)
```

The terminal shows the following status bar:

Data Output Explain Messages Notifications

CREATE INDEX

Query returned successfully in 66 msec.

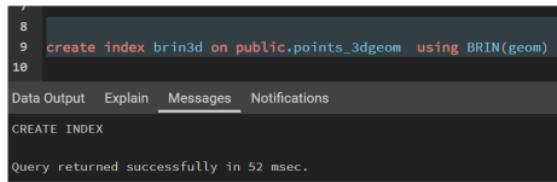
In the bottom right corner, there is a small message box with the text "Query returned successfully in 66 msec."

Şekil 6.3 Create İndex Using SPGIST sorgusu

6.1.3 BRIN

BRIN metodu ise diğer ikisinden tamamen farklıdır. Tablolar büyükçe verimliliği artıran bir metottur. Özellikle düzenli artan ve sıralı yerleşen sütunlarda oldukça efektiftir. timestamp sütunu sayesinde verinin düzenli arttığını ve sıralı yerleştiğini bilmektedir.

İndexin build süresi şekil 6.4deki gibidir.



A screenshot of a PostgreSQL terminal window. The command entered is:

```
7
8
9 create index brin3d on public.points_3dgeom using BRIN(geom)
10
```

The terminal shows the following status bar:

Data Output Explain Messages Notifications

CREATE INDEX

Query returned successfully in 52 msec.

Şekil 6.4 Create İndex Using BRIN sorgusu

7

Performans Analizleri ve Yapılan Çıkarımlar

Testler için uzamsal indexlerin çalışmasına imkan veren sorgular yapılması gerekliydi. Tablo noktalardan olduğu için ST_DWithin fonksiyonunun kullanılmasına karar verildi. Bu fonksiyon verilen bir noktaya istenen uzaklıktaki bütün noktaları getirebilen bir sorgu yazılmasını sağlar. Uzaklık değerinin büyük ve küçük verildiği iki farklı sorgu hazırlandı. Sorguların döndürdüğü değerler şekil 7.1 ve şekil 7.2 deki gibidir.

```

Query Editor  Query History
1 select name, ST_AsText(geom)
2 from public.points_3dgeom
3 where ST_DWithin(
4     points_3dgeom.geom,
5     'SRID=4326;POINT M (10429439.688216459 4540017.565623412 1643140802)',
6     280
7 )
Data Output Explain Messages Notifications
name          st_astext
character varying (48)  text
1 Balaban Caddesi POINT M (10429469.05819318 4540042.823314596 1643140800)
2 Balaban Caddesi POINT M (10429439.688216459 4540017.565623412 1643140802)
3 Balaban Caddesi POINT M (10429413.643237237 4539987.206785204 1643140804)
4 Balaban Caddesi POINT M (10429386.778061789 4539957.63633476 1643140806)
5 Balaban Caddesi POINT M (10429352.759188263 4539937.060383019 1643140808)
6 Balaban Caddesi POINT M (10429469.05819318 4540042.823314596 1643140800)
7 Balaban Caddesi POINT M (10429494.848473452 4540073.398818118 1643140802)
8 Balaban Caddesi POINT M (10429520.636310967 4540103.976382288 1643140804)
9 Balaban Caddesi POINT M (10429546.42679001 4540134.551718601 1643140806)
10 Balaban Caddesi POINT M (10429572.217269056 4540165.127054914 1643140808)

```

Şekil 7.1 ST_DWithin ile yapılan büyük uzaklık sorgusu

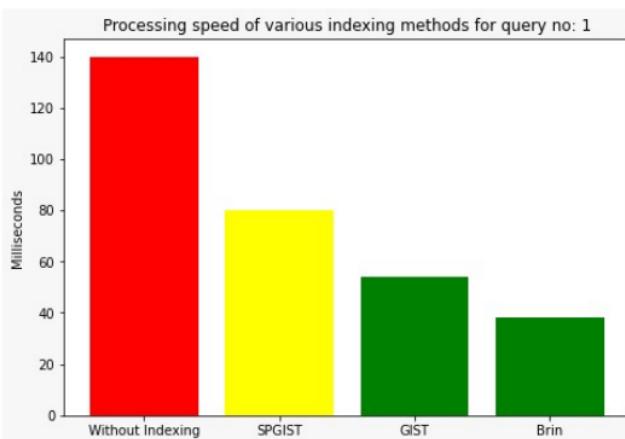
```

Query Editor  Query History
1 select name, ST_AsText(geom)
2 from public.points_3dgeom
3 where ST_DWithin(
4     points_3dgeom.geom,
5     'SRID=4326;POINT M (10429439.688216459 4540017.565623412 1643140802)' ,
6     15
7 )
Data Output Explain Messages Notifications
name          st_asText
character varying (48)  text
Balaban Caddesi POINT M (10429439.688216459 4540017.565623412 1643140802)

```

Şekil 7.2 ST_DWithin ile yapılan küçük uzaklık sorusu

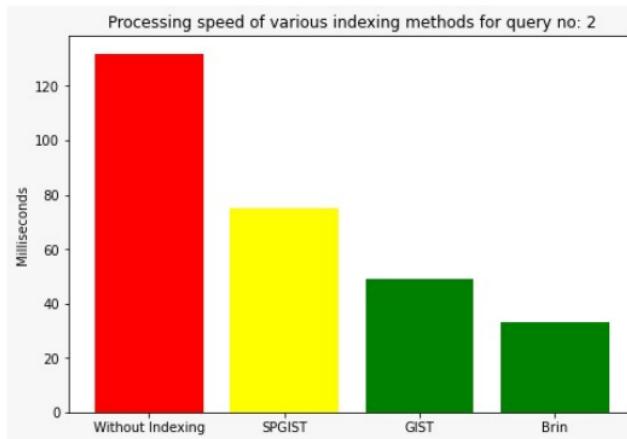
Çalışmada öncelikle referans için bu tablo üzerinden indeksleme işlemi yapılmamışken sorgular çalıştırıldı. Daha sonra diğer indexler tek tek denendi ve şekil 7.3 ve şekil 7.4de görülen sonuçlara ulaşıldı. Küçük uzaklık sorusunda daha çabuk sonuç alınmış olsa da sonuçlar oldukça benzerdir



Şekil 7.3 Büyük uzaklık sorusunun çalıştırılma süresinin index çeşitlerine göre karşılaştırılması

Bu sonuçların belli şartlar altında elde edildiğinin tekrar belirtilmesi gereklidir. Öncelikle tablo büyütüğü kullanılan bilgisayarların CPU gücünün kısıtları içerisinde küçük tutulmaya çalışılmıştır. Ayrıca PostgreSQL belli durumlarda daha hızlı işlem yapabileceğini hesaplayıp, kullanıcının kontrolü dışında bir şekilde indexlemeyi kullanmamayı tercih etmektedir. Bu seçimi bazen sorgu sonucunun cache’te bulunması ya da kullanıcının erişemediği daha kullanışlı indexlerin bulunması gibi faktörler etkileyebilir. Çalışmamız bizi ilgilendiren index tiplerinin özellikleri ve hangi durumlarda faydalı olabileceği üzerine olduğunu sonuclar alınırken olabildiğince bu faktörlerden ayrı tutulmaya çalışılmıştır.

Bahsedilen bu diğer faktörler görmezden gelindiğinde performans analizlerinin



Şekil 7.4 Küçük uzaklık sorgusunun çalıştırılma süresinin index çeşitlerine göre karşılaştırılması

sonuçları bekleniği gibi çıkmaktadır.

- Indexlenmemiş veri sorgulanırken zaman kaybı olması doğaldır.
- SPGIST metodu tablomuz için en az verimli olandır, bunun sebebi de metodun özelleştirildiği özelliklerin tablomuzda bulunmamasıdır. Bütün noktaların farklı geom değerlerini sahip olduğu bilinmektedir.
- GIST metodu, SPGIST'e göre daha başarılı olmuştur, çünkü verinin uniform dağıldığı bilinmektedir.
- BRIN metodu projemiz için en verimli indexleme metodur, sıralı ve düzenli artan verimize iyi uyum sağlamaktadır.

8 Sonuç

Projemizde uzamsal veritabanlarına giriş yaptıktı ve yeni konseptler öğrenmiş olduk. Bu alanda yaygın olarak kullanılan toolları tanıdık ve veritabanı kullanabilme becerilerimizi geliştirdik.

Projemizin ilk yarısında bulunan açık kaynak üzerinden İstanbul haritasının çeşitli katmanları çekildi ve QGIS ortamına eklendi. Bu sayede, sentetik veriseti üretimi için gerekli ortam hazırlanmış oldu. Yol haritasının CRS formatı ve ölçü birimi değiştirildi, rasgele 2500 tane sokak seçildi ve bunlardan belirli bir uzunluğun altında olanlar filtreldi. Devamında, Bu rotalar üzerinde her 40 metreden bir, rota üzerinde hareket eden araçların yönünesini temsil eden noktalar üretildi ve sokakların keskin dönüşlü yerleri de göz önünde bulunarak belirli bir hızda göre zaman etiketleri atandı.

İkinci yarında QGIS'te işlenen verileri PostGIS'e taşıdı ve önceden 2 boyutlu olan geometri sütununa zaman boyutunu ekleme işlemi tamamlandı. Elde edilen son tablo üzerinde öğrendiğimiz 'spatial indexing' yöntemleri sayesinde analizler yapıldı. Analizler sonucunda PostGIS fonksiyonlarıyla oluşturulmuş sorgular kullanıldı. Sonuçta, büyük ve 3 boyutlu verilerde indekslemenin faydalı olduğu yeniden kanıtlanmış oldu ve projede verinin en uygun indeks tipinin BRIN metodu olduğunu gözlemlendi. Alınan sonuçlar da Python'un Matplotlib kütüphanesi yardımıyla görselleştirildi.

Ara Proje Raporu

ORIGINALITY REPORT



PRIMARY SOURCES

1	v1.overleaf.com Internet Source	1 %
2	tr.theindipedia.com Internet Source	1 %
3	repositorio.una.edu.ni Internet Source	<1 %
4	batteriepc.exblog.jp Internet Source	<1 %
5	www.ludoetsophie.com Internet Source	<1 %
6	www.ulubey.gov.tr Internet Source	<1 %
7	9lib.net Internet Source	<1 %
8	HARUNOĞULLARI, Muazzz. "KİLİŞ'İN İNANÇ TURİZMİ POTANSİYELİ VE KUTSAL ", Erzincan Üniv. Fen Edebiyat Fak. Türk Dili ve Edebiyatı Bl., 2016. Publication	<1 %

9	docplayer.biz.tr Internet Source	<1 %
10	www.owasa.org Internet Source	<1 %
11	futur.upc.edu Internet Source	<1 %
12	www.makinaegitim.com Internet Source	<1 %

Exclude quotes On

Exclude bibliography Off

Exclude matches Off