

Önsöz

Coğrafi Bilgi Sistemleri II dersi, birinci sınıf bahar döneminde görülen Algoritma ve Programlama dersi ve ikinci sınıf güz döneminde görülen Coğrafi Bilgi Sistemleri I derslerinin devamı şeklinde planlandı. Ders içeriği, günümüz Coğrafi Bilgi Sistemleri (CBS) teknolojilerinin gelişmesine paralel olacak şekilde ve açık kaynaklı CBS programlarının kullanım artışını dikkate alarak, hazırlanmıştır. Açık kaynaklı CBS yazılımlarına haiz olmanın getirdiği önem ile orta ve büyük ölçekli projelerin ihtiyacını karşılayacak açık kaynaklı veri tabanı yönetim sistemleri yazılımlarına haiz olmanın getirdiği önem düşünerek Kaman Meslek Yüksekokulu öğrencilerimize bu dersi hazırlamış bulunmaktayım.

Harita ve Kadastro Programı Öğretim Elemanı

Öğr. Gör. EMRE İNCE

2023

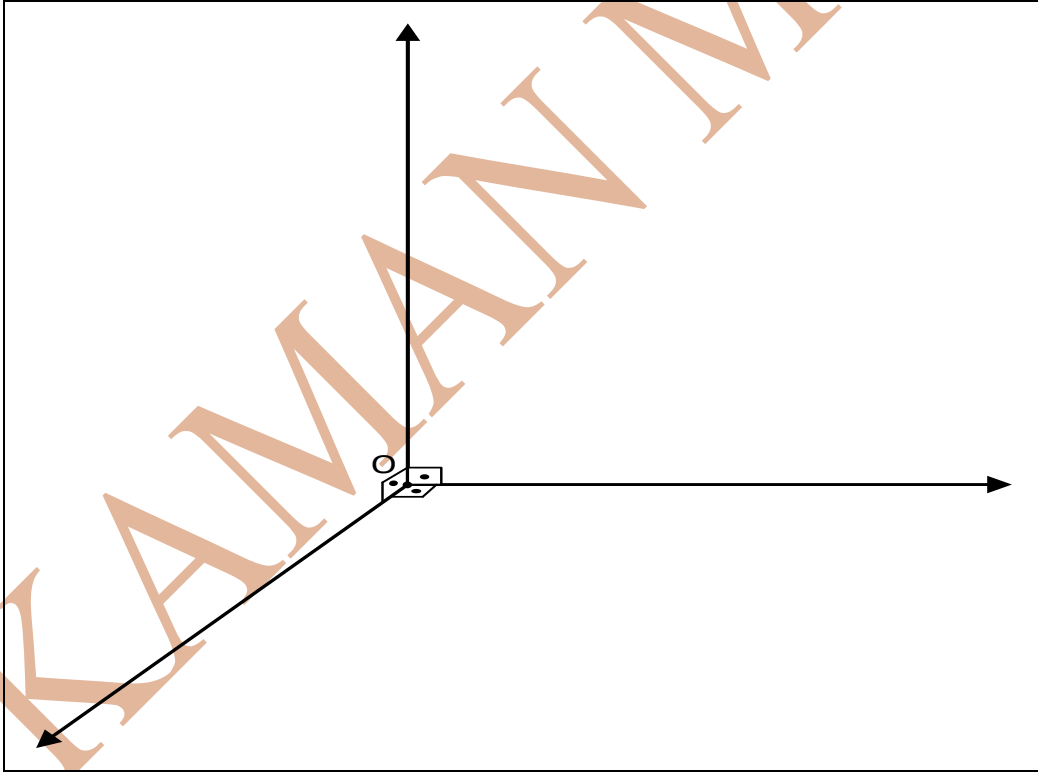
Koordinat Sistemi

Reference Frame (Referans Çerçevesi) ve 3 Boyutlu Kartezyen Koordinat Sistemi

Referans çerçevesi ifadesindeki çerçeve kelimesi *Şekil 1*'de görülmektedir. Çerçeve, koordinat sisteminin eksenlerinin belirtilmemiş, eksen yönleri veya dönme yönleri belirtilmemiş halidir. 3 boyutlu koordinat sistemi referans çerçevesinin

- Orijin noktasının yerinin belirli olduğu,
- Eksenlerin hangi yönde arttığı,
- Eksenlerin dönme yönlerinin belirli olduğu,
- Eksen üzerinde orijin noktasına uzaklığın veya düzlemlerdeki açı birimlerinin ifadelerinin belirli olduğu durumdur.

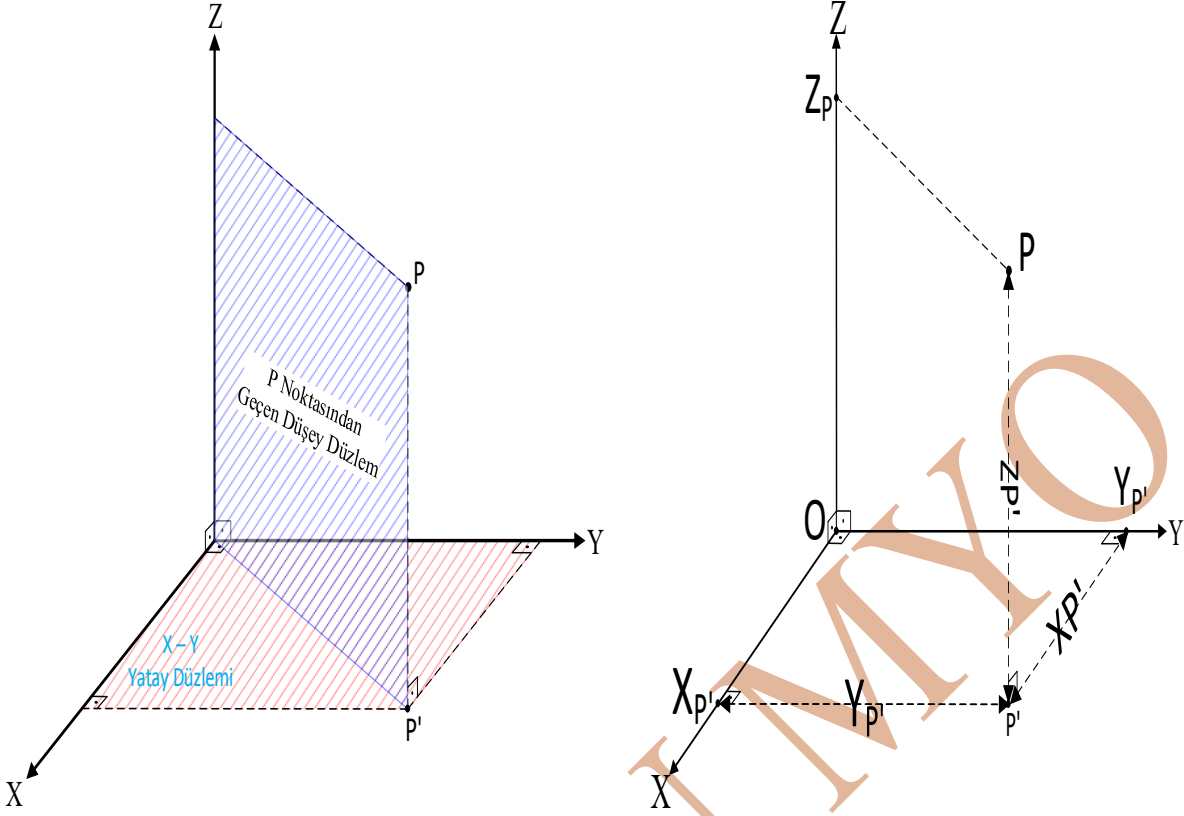
Kısa bir tanımla referans çerçevesi, 3 boyutlu kartezyen (eksenlerin birbirine dik olması) koordinat sisteminin tanımında ve orijin noktasının yerleşiminde kullanılır.



Şekil 1

Coğrafi objeleri sadece 2 boyutlu bir koordinat sistemi (X – Y yatay düzlemi) kullanarak gerçekteki konum değerlerini belirleyemeyiz. Nokta konum değerlerini ifade etmek için 3 boyutlu Kartezyen koordinat sisteminin kullanılması gerekir. **Referans çerçevesi 3 boyutlu Kartezyen koordinat sisteminin oluşması için gerekli tanımları yapar.**

Şekil 2 3 boyutlu referans çerçevesinin (3 boyutlu kartezyen koordinat sistemi) kullanımını temsili olarak gösterilmiştir.



Şekil 2

Şekil 2 sol resim, incelendiğinde koordinat sisteminde P noktasının eksenler üzerindeki koordinatlarını bulabilmek için iki farklı düzlem kullanılmıştır. Z eksenindeki koordinatı bulabilmek için P noktasından geçen düşey düzlem kullanılmıştır. P noktasından geçen düşey düzlemde P noktasından Z eksenine dik inilmiştir.

X ve Y eksenindeki koordinatları bulabilmek için P noktasından, X – Y eksenlerinin oluşturduğu yatay düzleme dik inilmiş ve yatay düzlemi kestiği nokta olan P' elde edilmiştir. P' noktasından X ve Y eksenlerine dik inilerek P noktasının yatay düzlemdeki koordinatları elde edilmiştir.

Yukarıda bir referans çerçevesi (3 boyutlu kartezyen koordinat sistemi) üzerinde bir noktanın X – Y – Z eksenleri üzerindeki koordinat temsilleri anlatılmıştır. Gerçekte bir referans çerçevesi oluşturabilmek için (daha önce de yazıldı): (Krakiwsky & Wells, 1971)

- Çerçevenin başlangıç noktasının konumu,
- Referans çatısının üç eksenin dönüklükleri ve artış yönleri,
- Bir noktanın konumunu tanımlayan parametreler ve birim bilgisi gereklidir.

Yukarıdaki üç kriteri dikkate alarak referans çerçevesine dair tanımlar yapılır ve 3 boyutlu koordinat sistemi oluşturulur. Uygulamada bu tanımlar dikkate alınarak hem harita yapımı için ölçümler hem de yapılan ölçümlere istinaden harita çizimleri yapılır.

Harita çizimi ve harita yapımı için referans çerçevesinin birden fazla kullanımı bulunmaktadır.

- 1) Toposentrik (Topocentric) sistem: Referans çerçevesinin orijin noktası yeryüzünde bir noktadır,
- 2) Jeosantrik (Geocentric) sistem: Referans çerçevesinin orijin noktası yeryuvarı (dünyanın) ağırlık merkezidir.

Toposentrik Sistem:

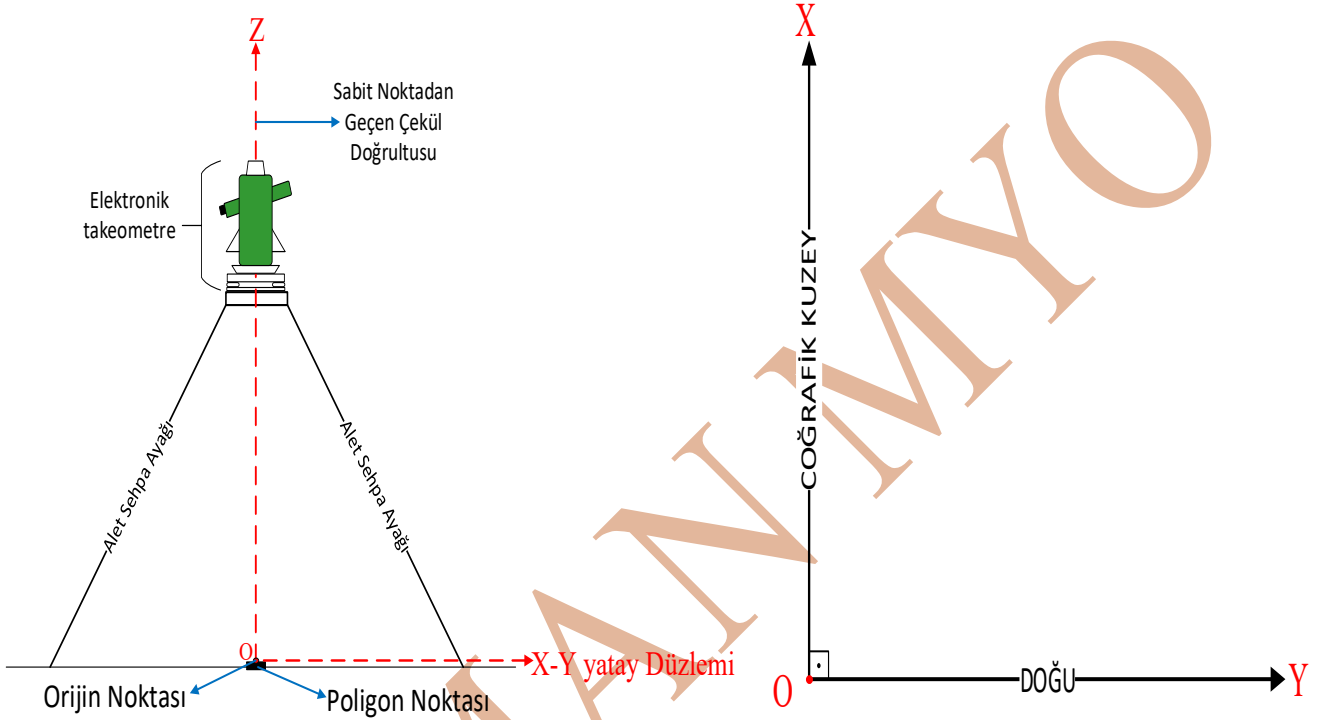
Toposentrik sistem, orijin noktasının yeryuvarı üzerinde olduğu referans sistemidir. Toposentrik sistem, harita yapımı için gerekli yersel ölçümlerde kullanılmaktadır. Elektronik takeometre kullanılarak, coğrafi objelerin detay noktalarının koordinatlarının hesaplanması için, ölçüm yapılması gerektiğinde iki adet sabit noktaya (koordinatı bilinen nokta - poligon noktası,...) ihtiyaç vardır. Noktalardan bir tanesi elektronik takeometrenin kurulu olduğu noktadır, diğer nokta ise başlangıç doğrultusunun oluşturulması için gerekli noktadır. Elektronik takeometrenin kurulu olduğu sabit noktada toposentrik referans çerçevesi oluşur. Bu referans çerçevesinin oluşması için gereken kriterler:

- a) Orijin noktası elektronik takeometrenin kurulu olduğu sabit noktadır,
- b) Z eksenini elektronik takeometrenin kurulu olduğu sabit noktadan geçen düşey doğrultusudur (çekül doğrultusu) ve Z ekseninin pozitif artışı sabit noktadan çekül doğrultusu boyunca yukarı doğrudur. Elektronik takeometre Z eksenini etrafında döner,
- c) X eksenini, elektronik takeometrenin kurulu olduğu sabit noktada oluşan coğrafi kuzey doğrultusuyla çakışıktır ve X koordinatının pozitif artışı kuzey yönündedir,
- d) Y eksenini, elektronik takeometrenin kurulu olduğu sabit noktada oluşan X eksenine dik açıyla oluşacak şekilde, doğu yönüyle çakışıktır ve Y koordinatının pozitif artışı doğu yönündedir,
- e) X – Y – Z eksenleri metre uzunluk birimiyle ifade edilirler.

Yukarıdaki toposentrik sistemde referans çerçevesi üzerinde eksenler için tanımlamalar yapılmıştır. Bu sayede Elektronik takeometrenin üzerine kurulu olduğu her sabit noktada yeniden toposentrik koordinat sistemi oluşur.

Şekil 3 sol ve sağ resimlerde 3 boyutlu toposentrik koordinat sistemi (toposentrik referans çerçevesi) temsili vardır. Sol resimde, elektronik takeometrenin üzerine kurulu olduğu poligon noktasında toposentrik referans çerçevesinin orijin noktası oluşmuştur. Poligon noktasından geçen çekül doğrultusu toposentrik referans çerçevesinin Z eksenini geçmektedir. X-

Y yatay düzlemi poligon noktasından geçmektedir (Şekil 3 sol resim). Şekil 3 sağ resim orijin noktasına (takeometrenin üstünden) kuş bakışı görünüşüdür. Şekil 3 sağ resim, poligon noktasında oluşan toposentrik referans çerçevesinin yatay düzlemi gözükmemektedir. X eksenin coğrafi kuzey ile çakışık olduğu ve artış yönü gözükmemektedir. Şekil 3 sağ resimde Y ekseninin X eksenine dik bir şekilde doğu yönünde olduğu ve artış yönünün doğu yönü olduğu gözükmemektedir. Bu tanımlamalar yapılarak toposentrik referans çerçevesini oluşturur.



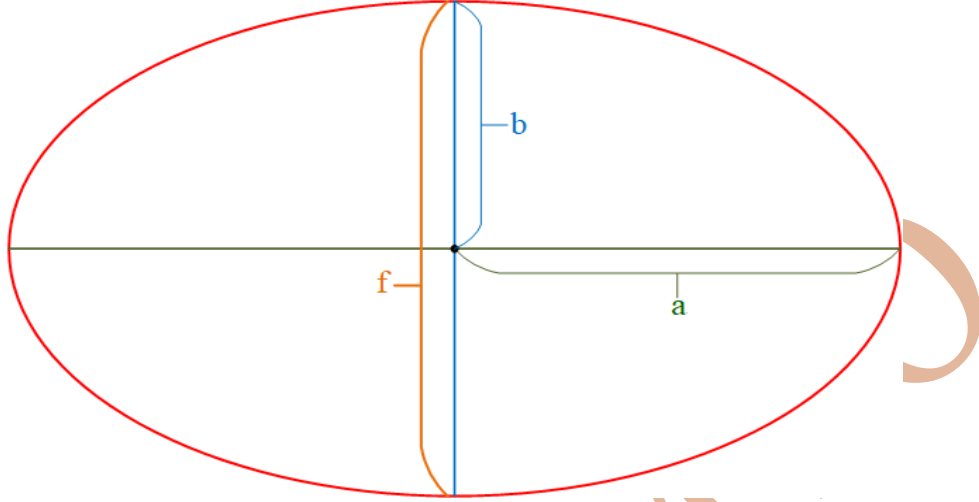
Şekil 3

Jeosantrik Sistem:

Jeosantrik Sistemde, referans sisteminin orijin noktası dünyanın ağırlık merkezidir. Bu sayede referans çerçevesi dünyanın içinde oluşmaktadır. Jeosantrik sistem, harita yapımı için gerekli ölçümlerden uydu bazlı konum belirleme yöntemlerinde kullanılmaktadır. Jeosantrik sistem dünya merkezli olduğu için, oluşan çerçeve tüm dünyada geçerli referans çerçevesine dönüşür.

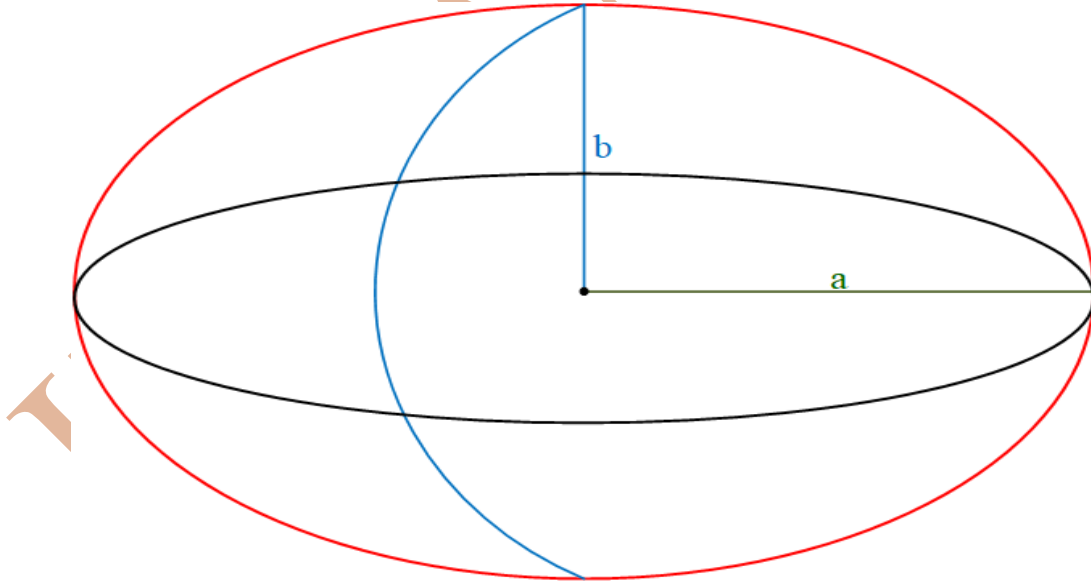
Yeryüvarı merkezli bir referans çerçevesini dünya yüzeyi ile ilişkili hale getirmek için dünya yerine referans yüzey kullanılmalıdır. Dünya yerine seçilecek referans yüzey üç boyutlu düzgün bir geometrik şekil olmalıdır. Dünya yerine seçilecek ve dünyayı en iyi temsil edecek olan düzgün geometrik şekil dönelel elipsoittir. Dönelel elipsoit elips düzgün geometrik şeklidir. Şekil 4 elips düzgün geometrik şeklinin temsilidir. Elips, daireden farklı olarak, iki ayrı eksenden oluşur (Eksenler Şekil 4'de mavi ve yeşil çizgilerle temsil edilmekte). Bu eksenlerin yarıçapları Şekil 4'de a ve b olarak gösterilmiştir. a yarıçap değeri b yarıçap

değerinden büyük olacak ($a > b$) şekilde elips oluşur. Şekil 4 üzerindeki f değeri, elipsin basıklık değeridir. Elipsin f basıklık değeri sayesinde dünyanın kutuplardan basıklığı ifade edilebilecektir.



Şekil 4

Dönel elipsoit elipsin küçük yarı eksenini olan b eksenini etrafında dönmesi sonucu oluşan üç boyutlu yüzeydir. Şekil 5 dönel elipsoidin tasviridir. Şekil 5'deki elipsoit, Şekil 4'deki elipsin küçük eksenini (b yarıçaplı eksen) etrafında dönmesiyle oluşur. Elipsoit oluştuğunda a yarıçaplı eksen bir dairenin yarıçapı olacak şekilde daire oluşur.



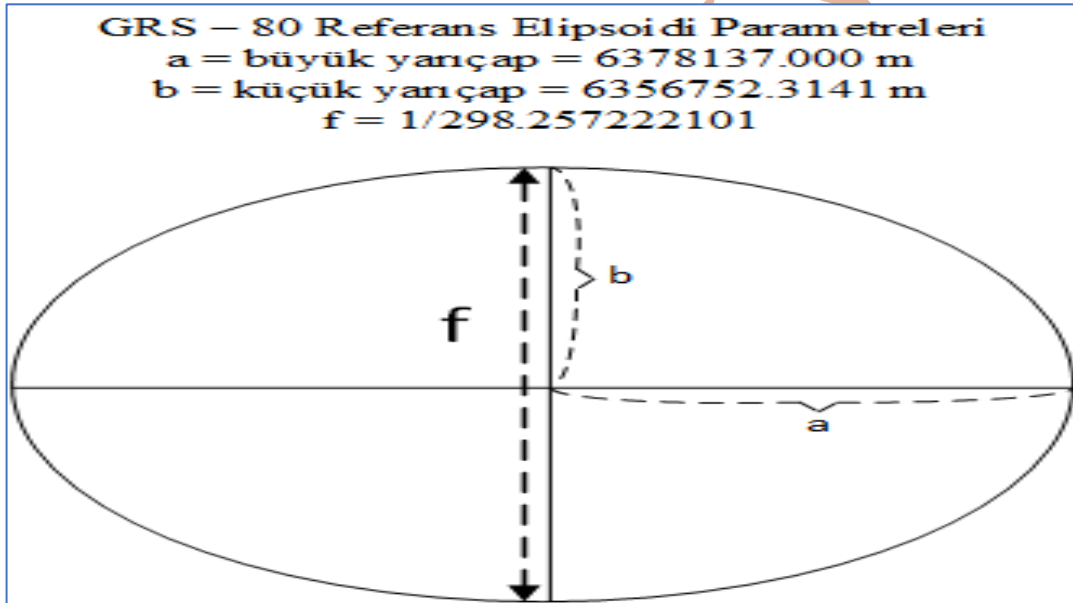
Şekil 5

Oluşan elipsoit dünya yerine kullanılacak referans yüzeyi olarak kullanılacaktır. Dünya yerine referans yüzeyi kullanabilmek için f basıklık değeri ve a büyük eksen yarıçap değerine ihtiyaç vardır. Bu değerler gerçekteki dünya üzerinde ölçülerek, dünya yerine bir model oluşturulacaktır. Kutuplar arası f basıklık değeri ve a büyük eksen yarıçap değeri, uydu bazı

yapılan ölçümlerle bulunur. Bulunan f basıklık değeri ve a büyük eksen yarıçap değeri yardımıyla elipsoidin diğer parametreleri de hesaplanarak referans yüzey modellenir. Ülkemizde büyük ölçekli haritaların yapımında kullanılan referans yüzey, Büyük Ölçekli Harita ve Harita Bilgileri Üretim Yönetmeliği'ne göre Geodetic Reference System 1980 (GRS-80) isimli elipsoiddir (Şekil 6). GRS-80 elipsoid parametreleri:

GRS – 80 Referans Elipsoidi Parametreleri	
a = büyük yarıçap	6378137.000 m
b = küçük yarıçap	6356752.3141 m
f = kutuplar arası basıklık değeri	1/298.257222101

$$f = \frac{a - b}{a}$$

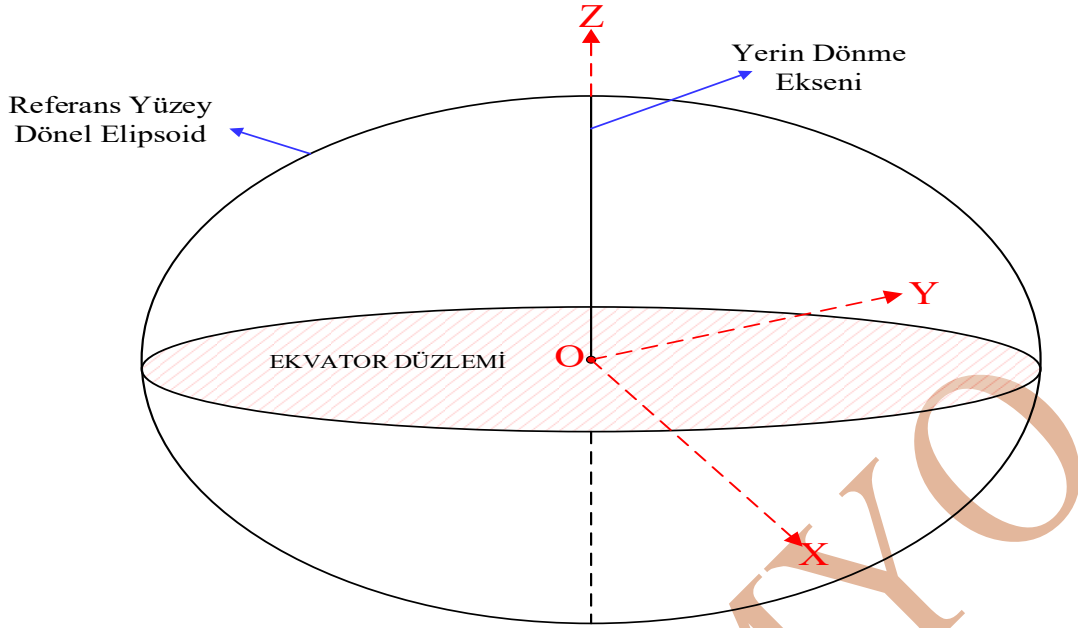


Şekil 6

Jeosantrik referans çerçevesinde koordinat sisteminin değerleri (X – Y – Z koordinatları) kullanılacak *referans yüzeyi ile ilişkili olduğu sürece* dünyanın içinde kalacak olan kartezyen koordinatlar *dünya yüzeyi koordinatlarına* dönüştürülebilir.

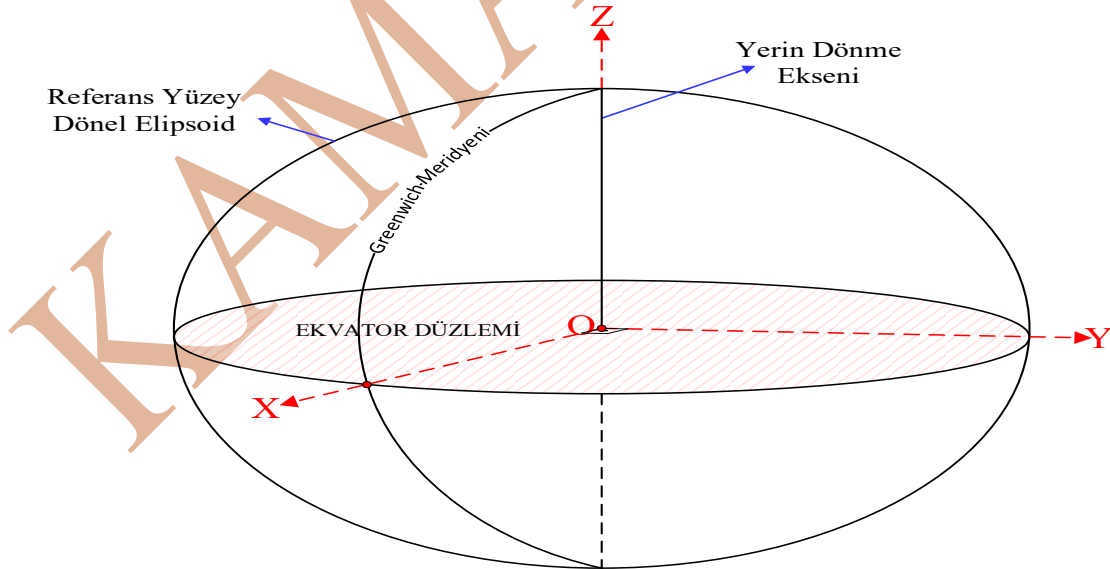
Jeosentrik referans çerçevesini oluşturmak için:

- referans elipsoidinin merkezi ile referans çerçevesinin orijin noktası çakıştırıldığında,
- Z eksenini elipsoidin dönme eksenine ile çakıştırıldığında ortaya çıkan sonuç Şekil 7'de tasvir edilmiştir.



Şekil 7

Eğer jeosantrik referans çerçevesi Şekil 7’de olduğu gibi bırakılırsa, dünya döndükçe X ve Y eksenleri sabit olmayacak, dünya üzerindeki noktaların koordinatları da her daim sabit kalmayacak değişecektir. X -Y yatay düzleminin sabit kalması için eksenlerden biri dünya üzerinde sabit bir noktadan geçmelidir. Şekil 8 X ekseninin sabitletlenmesinin tasviri vardır. X eksenini Greenwich meridyeninin ekvator düzlemini kestiği noktadan geçer. Bu sayede X – Y yatay düzlemi de sabitletlenmiş olur.



Şekil 8

Dünya dönmesi sonucunda, X – Y yatay düzlemi sabit kalacak, dünya üzerindeki noktaların jeosantrik referans çerçevesindeki X – Y koordinatları değişmeyecektir. Jeosentrik referans çerçevesinin oluşması için:

- Referans çerçevesinin orijin noktası (Şekil 8 O noktası) yeryuvarının (dünyanın) ağırlık merkezinde olmalıdır,
- Referans çerçevesinin Z eksenini (Şekil 8), yerin dönme eksenini ile çakıştıktır. Z eksenini dünyayla beraber dönmelidir,
- Referans çerçevesinin X – Y yatay düzlemi, dünyanın ekvator düzleminde oluşmalıdır,
- Referans çerçevesinin X eksenini, Greenwich başlangıç meridyeninin ekvator düzlemini kestiği noktadan geçecektir (Şekil 8). Bu sayede X ve X – Y düzlemi dünya üzerinde sabitlenmiş olacaktır. Dünya döndüğünde X – Y düzlemi de dönecektir (Şekil 9), dünya üzerindeki noktaları Jeosentrik referans çerçevesindeki koordinatları değişmeyecektir,
- Referans çerçevesinin Y eksenini, X eksenine dik olacak ve 90° boylam değerine sahip meridyeninin ekvator düzlemini kestiği noktadan geçmelidir.



Yukarıda yapılan tanımlara karşılık gelen, haritacılıkta kullanılan Jeosentrik referans çerçevesi ismi vardır. Bu tanım Yer Merkez Yer Sabit Koordinat Sistemi (Earth Centered Earth Fixed Coordinate System - ECEF) olarak isimlendirilir. Yer

merkez ifadesi, referans çerçevesinin dünyanın ağırlık merkezi ile çakışmasından dolayı kullanılmaktadır. Yer Sabit ifadesi, X ekseninin Greenwich Meridyeninin ekvator düzlemini kestiği yerden geçmesiyle X – Y – Z kartezyen koordinat sisteminin dünya ile aynı anda dönmesinden ötürü kullanılmaktadır.

Şekil 9 (Department of Aerospace Engineering at The University of Bristol, 2023)

Datum Kavramı:

Datum, yatay datum ve düşey datum olarak ikiye ayrılır. Yatay datum, üretilecek harita yapımı için yapılacak ölçümlerde temel alınacak referans çerçevesinin koordinat sistemini ifade etmesi için gerekli 4 parametre bilgileri, referans çerçevesine göre elde edilen koordinatların harita düzlemine aktarılmasında kullanılacak referans yüzey bilgilerini içeren standart tanımlardır. Yatay datum oluşturulurken, uydu bazlı ölçümlerle dünya yüzeyinde birçok noktanın koordinatları hesaplanarak gerçek bir modelin parametreleri hesaplanır ve hesaplamalara göre dünya yerine hesaplamalarda kullanılacak model oluşturulur.

Büyük Ölçekli Harita ve Harita Bilgileri Üretim Yönetmeliği (BÖHHBÜY) Madde 7 L bendinde Türkiye’de kullanılan referans çerçevesinin tanımı yapılmıştır. Bu tanıma göre adı **Türkiye Ulusal Referans Çerçevesi (TUREF)** olarak belirlenmiştir (Büyük Ölçekli Harita ve Harita Bilgileri Üretim Yönetmeliği, 2018). Tanımda Türkiye Ulusal Referans Çerçevesinin uluslar arası bir datum olan International Terrestrial Reference Frame 1996 (ITRF96 - Uluslararası Yersel Referans Çerçevesi 1996) ile çakışık olduğu belirtilmiştir. . ITRF 96 datumu jeosantrik referans çerçevesidir ve referans yüzeyi olarak GRS – 80 elipsoidini kullanır.



Türkiye Ulusal Temel GPS Ağı (TUTGA) koordinatları TUREF koordinat sisteminde sunulmaktadır (BAYKAL, 2009)



*TUREF referans çerçevesine göre elde edilen koordinatlar farklı isimler ile ifade edilebilir. Bu isimler **Jeosentrik koordinatlar**, **Kartezyen koordinatlar** olarak ifade edilebilir.*

TUREF referans çerçevesinde bulunan koordinatlar dünyanın ağırlık merkezinin orijin olduğu bir koordinat sistemine göre elde edilmiştir. Bu koordinatlar dünya yüzeyindeki bir noktaya aittir ama koordinatlar dünyanın yüzeyinde değildir. Bu sebeple jeosantrik koordinatlarla noktalar harita düzlemi üzerine aktarılamaz, noktalar kullanılarak coğrafik objeler harita düzlemi üzerinde temsil edilemezler.

Jeosantrik koordinatlara sahip noktaların harita düzleminde ifade edilebilmesi için nokta koordinatlarının dünya yüzeyi üzerindeki koordinatlarının bulunması gerekmektedir. Dünya yüzeyindeki koordinatları elde edebilmek için yapılacak hesaplamalarda dünya yerine kullanılacak bir hesap yüzeyine (referans yüzeyine) ihtiyaç vardır. Türkiye’de TUREF

koordinatların (jeosantrik koordinatlar) dönüştürüleceği hesaplama yüzeyi Geodetic Reference System-1980 (GRS-80) elipsoididir.

Jeosantrik yersel dik koordinatların (Yer Merkez Yer Sabit koordinat sistemindeki koordinatlar) referans olarak kullanılacak yüzeyde karşılık gelen dik koordinatlarının bulunması gerekir. Bu işlemi yapmak için coğrafik koordinat sisteminin bir türevi olan jeodezik koordinatlara ihtiyaç vardır.

Coğrafi Koordinat Sistemi ve Jeodezik Koordinat Sistemi

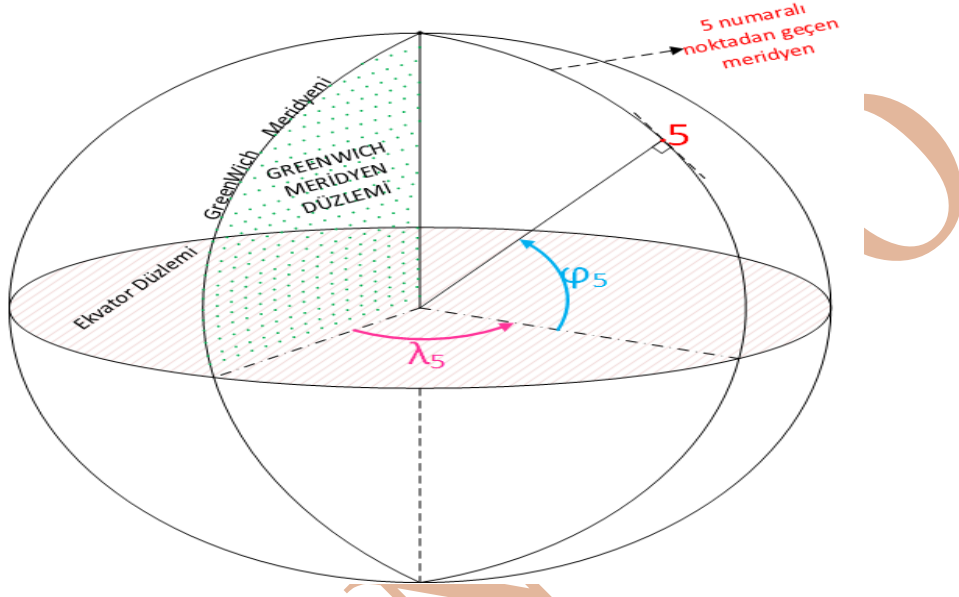
Jeosantrik Koordinatların (TUREF sistemine göre koordinatlar– Kartezyen Koordinatlar) harita düzlemine aktarılabilmesi için ilk önce jeosantrik koordinatların referans yüzeyindeki koordinatlarının bulunması gerekir. Fiziksel yeryüzü geometrisinin belirlenmesi (dünya şeklinin belirlenmesi) açısından bir hesap yüzeyine (dünya yerine kullanılacak projeksiyon işlemlerindeki hesaplama yüzeyi – dönel elipsoit) gerek olmadığı düşünülebilir, ancak (BAYKAL, 2009):

- Jeosantrik referans çerçevesindeki koordinat sisteminde elde edilmiş koordinatlar, doğrudan doğruya harita düzlem koordinatlara dönüştürülemediğinden (dönüştürmek için bir ara yüzey – dönel elipsoit kullanılmaktadır) yani düzlem haritaya aktarılamadığından,
- Çekül doğrultusunu temel alarak çalışan elektronik takeometre (total station), nivo gibi donanımlarla üretilen ölçüm değerlerini ve üretilen dik koordinatları, jeosantrik yersel koordinat sisteminde kullanarak yeni büyüklükler hesaplamak mümkün olmadığından referans yüzey kullanılır.

Ülkemizde, TUREF sisteminde elde edilmiş koordinatların GRS-80 dönel elipsoit yüzeyinde üzerindeki ifade edilecek koordinatlara dönüştürülmesi gerekir. Hesaplamalarda referans yüzey olacak elipsoidin ana (büyük) yarıçap (a eksen) değeri ve elipsoidin basıklık değeri (f) gibi değerleri, dünyanın geometrik bilgilerine göre hesaplanır.

Dünya üzerindeki noktanın Jeosantrik koordinatlarından, referans yüzeyindeki koordinatlara dönüştürmek için ilk önce noktanın **Coğrafi Koordinatları** belirlenmelidir. *Şekil 10*, dünya yerine düzgün geometrik şekil olarak kürenin referans yüzey belirlenmesiyle oluşan noktanın coğrafik koordinatlarının temsildir. *Şekil 10*'de 5 numaralı noktanın coğrafik koordinatları olan φ_5 enlemi ile λ_5 boylam koordinatları temsili vardır. Noktanın Enlem koordinatı (φ_5), 5 numaralı noktadan geçen yüzey normalinin (çekül doğrultusu) kürenin merkezi ile kesiştiği doğrunun ekvator düzlemi ile yaptığı açıdır. Noktanın Boylam koordinatı

(λ_5), başlangıç meridyeni olan Greenwich meridyenin oluşturduğu düzlem ile noktadan geçen (Şekil 10'de 5 numaralı noktadan geçen) meridyen düzlemi arasında kalan açıdır. Jeosantrik koordinat değerleri metrik birimde ifade edilirken, coğrafik koordinat değerleri açı değerindedir ve derece açı biriminde ifade edilir. Alman literatüründe enlem “Breite” (enlem) ve “Länge” (Boylam) kelimelerinin baş harfleri olan *B* ve *L* harfleriyle de ifade edilir (TORGE, 1991).

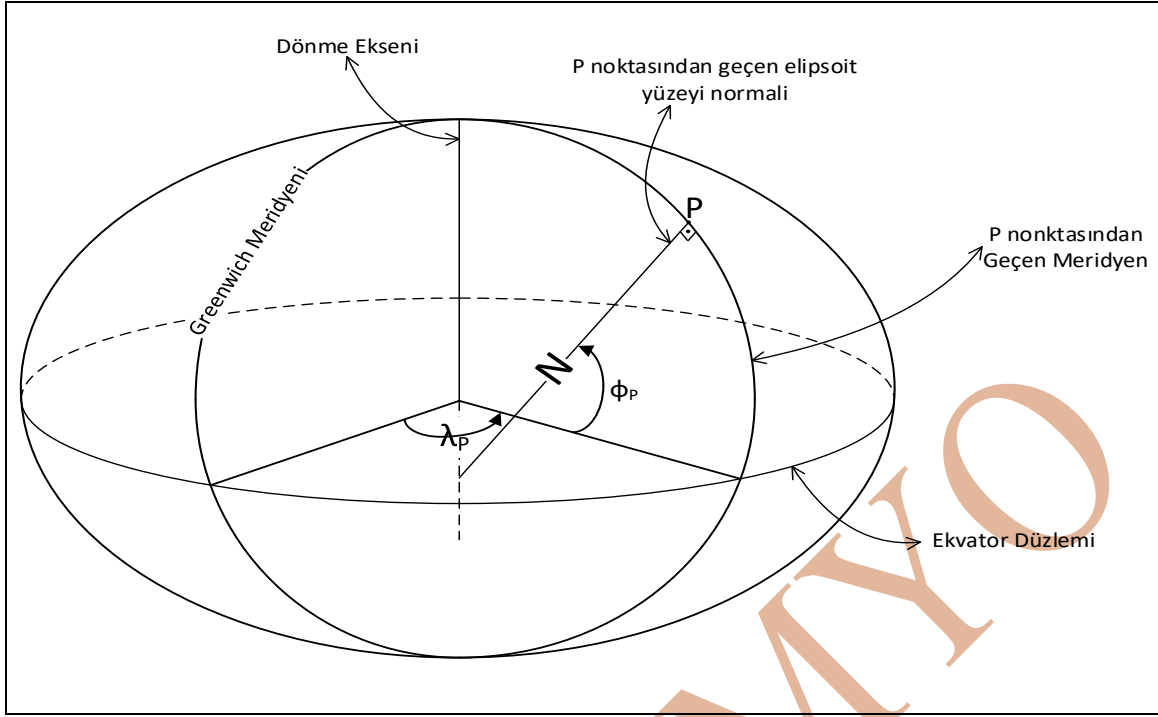


Şekil 10

Enlem koordinatları ekvator düzleminde 0° (sıfır derece) değeri olacak şekilde bu düzlemden itibaren kuzeye doğru pozitif (+) olarak artarken, güneye doğru negatif (-) değerlidir, fakat mutlak artış gösterir. Enlem değerleri, 0° ile $+90^\circ$ aralığında kuzey enlem; 0° ile -90° aralığında güney enlem değerleri arasındadır. Boylam koordinatları Greenwich başlangıç meridyeni 0° (sıfır derece) değeri olacak şekilde bu düzlemden doğuya doğru pozitif (+) olarak artarken, batıya doğru negatif (-) değer fakat mutlak artış gösterir. Boylam değerleri 0° ile $+180^\circ$ aralığında doğu boylam; 0° ile -180° aralığında batı boylam değerleri arasındadır.

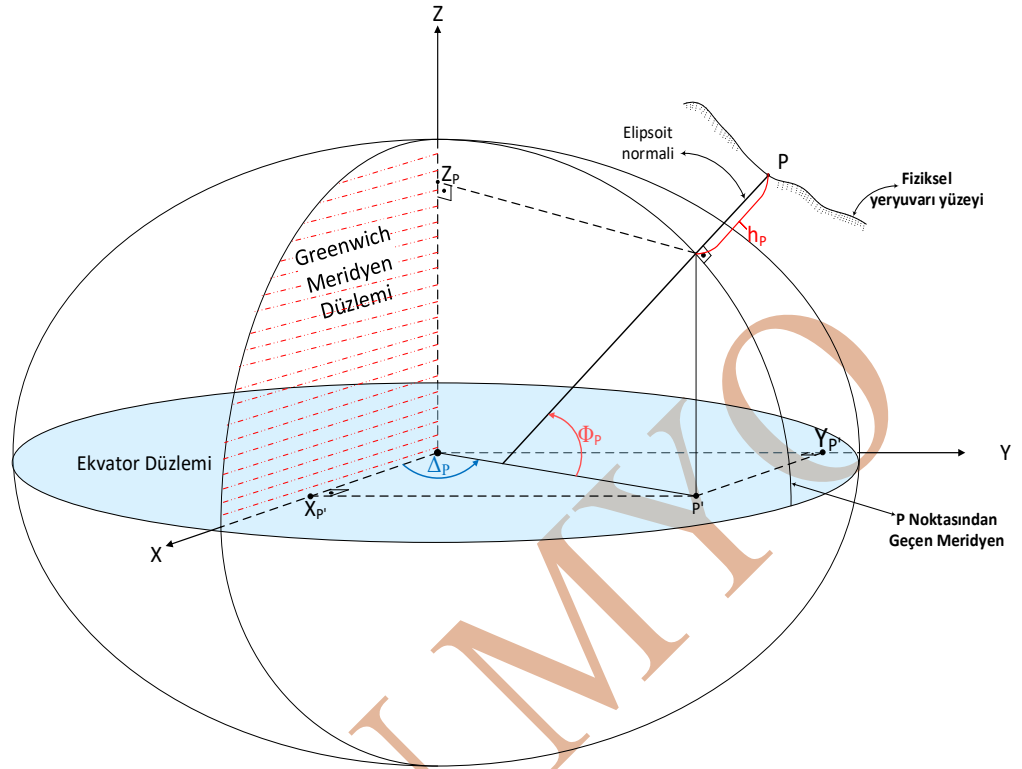
Coğrafi koordinat sistemi ilk tanımında referans yüzeyi olarak küre kullanılmıştı. Noktadan geçen çekül doğrultusu, kürenin merkezi ile birleşiyor. Çünkü kürenin yarıçapı sabittir, küre yüzeyi üzerindeki her noktanın çekül doğrultusu (noktadan geçen düşey ve yüzeye dik doğrultu) küre merkezi ile birleşir.

Referans yüzey olarak elipsoit kullanıldığında noktadan geçen çekül doğrultusu (elipsoit normali) elipsoit merkezinden geçmez (Şekil 11).



Şekil 11

Elipsoit tam küre değildir, kutup noktalarından basıklığı vardır. Elipsoit yüzeyi üzerindeki noktaların enlem ve boylam açı artış yönleri ve koordinat oluşum mantıkları aynıdır. Fakat noktaların koordinatları artık coğrafi koordinatlar yerine **JEODEZİK KOORDİNATLAR** olarak ifade edilecektir. Şekil 12 elipsoit yüzeyi üzerindeki P noktasından geçen çekül doğrultusunun (elipsoit normalini) ekvator düzlemini ile yaptığı açıya **jeodezik enlem** denir (Şekil 12, φ_p). Şekil 12 elipsoit yüzeyi üzerindeki P noktasından geçen meridyeni ile başlangıç meridyeni olan Greenwich Meridyeni arasında kalan açıya **jeodezik boylam** denir. Teknik kitaplarda coğrafi koordinat değerlerinden enlem küçük Phi φ (veya B), boylam küçük lamda λ (veya L) ile gösterilirken, jeodezik koordinat değerlerinden enlem büyük Phi Φ (Şekil 12), boylam büyük lamda Δ (Şekil 12) ile gösterilir. Jeodezik koordinatlar da 3 boyutludur. Yükseklik değeri **h** ile gösterilir ve elipsoit yüksekliği olarak ifade edilir (Şekil 12, **h**). Şekil 12 hem jeosantrik koordinatlar (X_p, Y_p, Z_p) hem de jeodezik koordinatlar (Φ_p, Δ_p, h_p) gösterilmiştir.



Şekil 12

Jeodezik Koordinat Sistemi ile Jeosantrik Yersel Koordinat Sistemi Arasındaki Geometrik İlişki

Jeodezik koordinat sisteminde:

- Dünya yerine dünyayı temsil eden referans yüzeyi dönel elipsoittir,
- Koordinat sisteminde bir noktanın jeodezik koordinatları enlem (B veya Φ) ve boylam (L veya Δ) açı değerleri ile ifade edilir,
- Koordinat sisteminin başlangıcı yeryuvarının (Dünya) ağırlık merkezi ile çakışık olan referans yüzey olan dönel elipsoidin merkezi olarak belirlenmiştir.

Jeosantrik koordinat sisteminde:

- Dünya yerine dünyayı temsil eden referans yüzeyi dönel elipsoittir,
- Koordinat sisteminde bir noktanın koordinat değerleri X , Y ve Z eksenlerindeki koordinat değerleriyle ifade edilir, koordinat değerleri metre uzunluk birimiyle ifade edilir,
- Koordinat sisteminin başlangıcı yeryuvarının (Dünya) ağırlık merkezi ile çakışık olan referans yüzey olan dönel elipsoidin merkezi olarak belirlenmiştir.

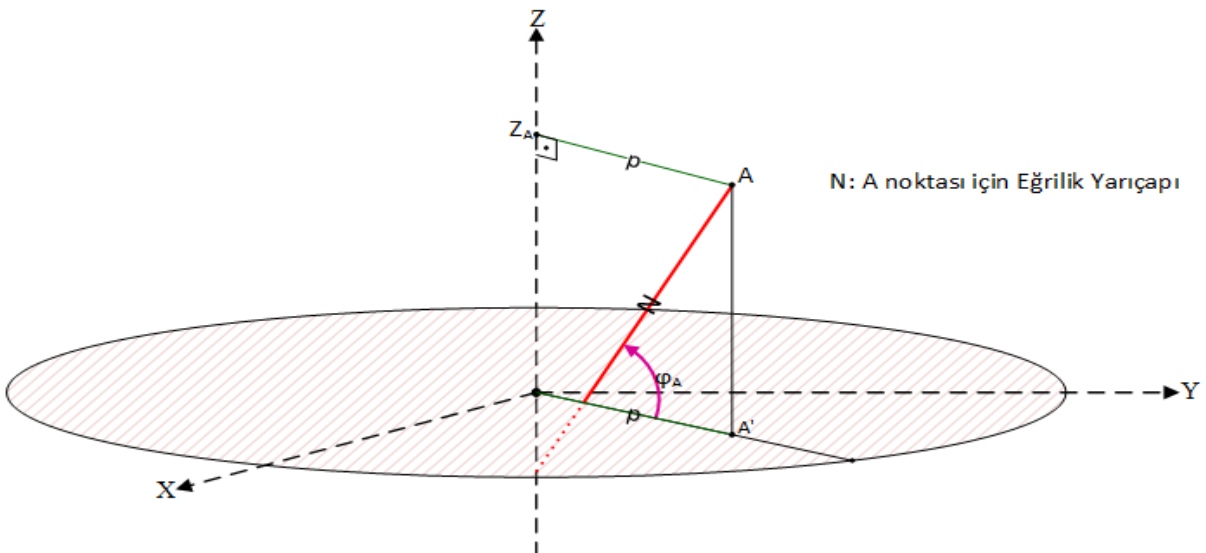
Jeosantrik Referans Çerçevesi Koordinatların (Kartezyen koordinatların ya da TUREF koordinatların) Harita Düzlem Koordinatlarına Aktarım Süreci:

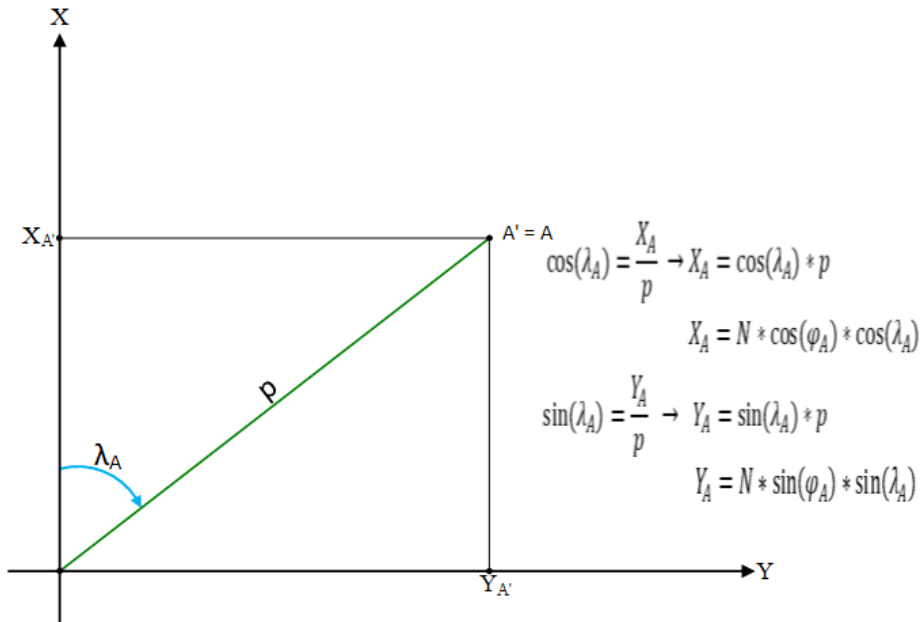
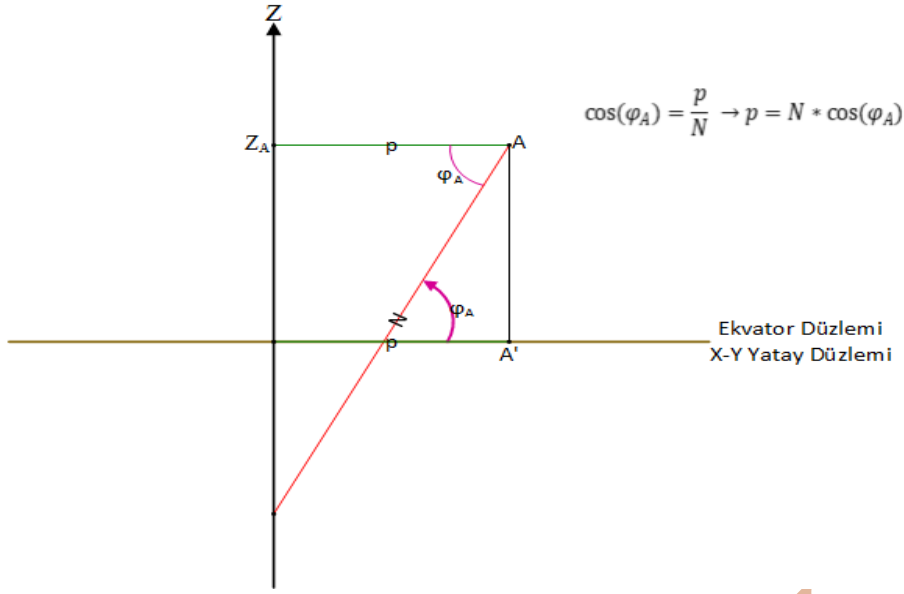
Eğer çalışmamızda Jeosantrik Koordinatlar (Yer Sabit Yer Merkez Koordinatlar – Kartezyen Koordinatlar) mevcutsa ve bu koordinatların harita düzlem koordinatlarına dönüştürülmesi gerekiyorsa:

- 1) Jeosantrik koordinatlar kullanılan referans yüzeyi (elipsoid) yardımıyla geometrik olarak dönüştürülebilecekleri jeodezik koordinatlara (Jeodezik enlem, Jeodezik Boylam, Elipsoit yüksekliği) dönüştürülmelidir.
- 2) Jeodezik koordinatlar, *projede kullanılan projeksiyon* ve *kullanılan referans yüzeyi* (örneğin elipsoit: GRS-80 elipsoidi) dikkate alınarak düzlem koordinatlara (Kuzey yönde artan X koordinatı – Yukarı koordinat ve Doğu yönde artan Y koordinatı – Sağa Koordinat) dönüştürülmelidir.

Jeodezik Koordinatlar (Enlem – Boylam Koordinatları) ile Jeosantrik Koordinatlar (Yer Sabit Yer Merkez Koordinatlar) Arasındaki Matematiksel Formüller:

Yukarıdaki işlem sırasına göre ilk olarak jeosantrik koordinatlardan (TUREF sisteminde elde edilmiş koordinatlardan – Kartezyen koordinatlardan) jeodezik koordinatlara dönüştürülmelidir. Bu sayede nokta koordinatı elipsoit yüzeyi üzerinde jeodezik koordinatlarla (jeodezik enlem, jeodezik boylam ve jeodezik yükseklik) ifade edilecektir. İşlemlerde kolaylık olması karışıklığın giderilmesi için jeodezik koordinat değerlerini ifade ederken büyük latin harfleri (Φ , Δ) yerine (ϕ , λ) ifadeleri kullanılmıştır.





Tablo 1 (Özbenli, 2001)

$B = \varphi = \text{Enlem}$
$e^2 = \frac{a^2 - b^2}{a^2} \rightarrow \text{birinci eksentrisite}$
$\frac{b}{a} = \sqrt{1 - e^2} = \frac{1}{\sqrt{1 + e'^2}} = \frac{e}{e'}$
$e'^2 = \frac{a^2 - b^2}{b^2} \rightarrow \text{ikinci eksentrisite}$
$\frac{a^2}{b} = c = \frac{a}{\sqrt{1 - e^2}}$

$\eta^2 = e'^2 * (\cos(B))^2$
$V = \sqrt{(1 + e'^2 * (\cos(B))^2)} = \sqrt{(1 + \eta^2)} = \text{Enlemin kullanıldığı bir fonksiyonu}$
$W = \sqrt{(1 - e^2 * (\sin(B))^2)} = \text{Enlemin kullanıldığı bir fonksiyonu}$
$N = \frac{a}{w} = \frac{c}{V} = \frac{a}{\sqrt{(1 - e^2 * (\sin(B))^2)}}$

Tablo 2 (Hofmann-Wellenhof, Lichtenegger, & Collins, 1994) jeodezik koordinatlardan jeosentrik koordinatların hesabı

$X = (N + h) * \cos(\varphi) * \cos(\lambda)$
$Y = (N + h) * \cos(\varphi) * \sin(\lambda)$
$Z = ((1 - e^2) * (N + h)) * \sin(\varphi) = ((1 - f)^2 * N + h) * \sin(\varphi)$
$N = \frac{a^2}{\sqrt{a^2 * (\cos(\varphi))^2 + b^2 * (\sin(\varphi))^2}} \text{ eğrilik yarıçapı}$

Tablo 3 (Hofmann-Wellenhof, Lichtenegger, & Collins, 1994) Jeosentrik koordinatlardan jeodezik koordinatların hesabı

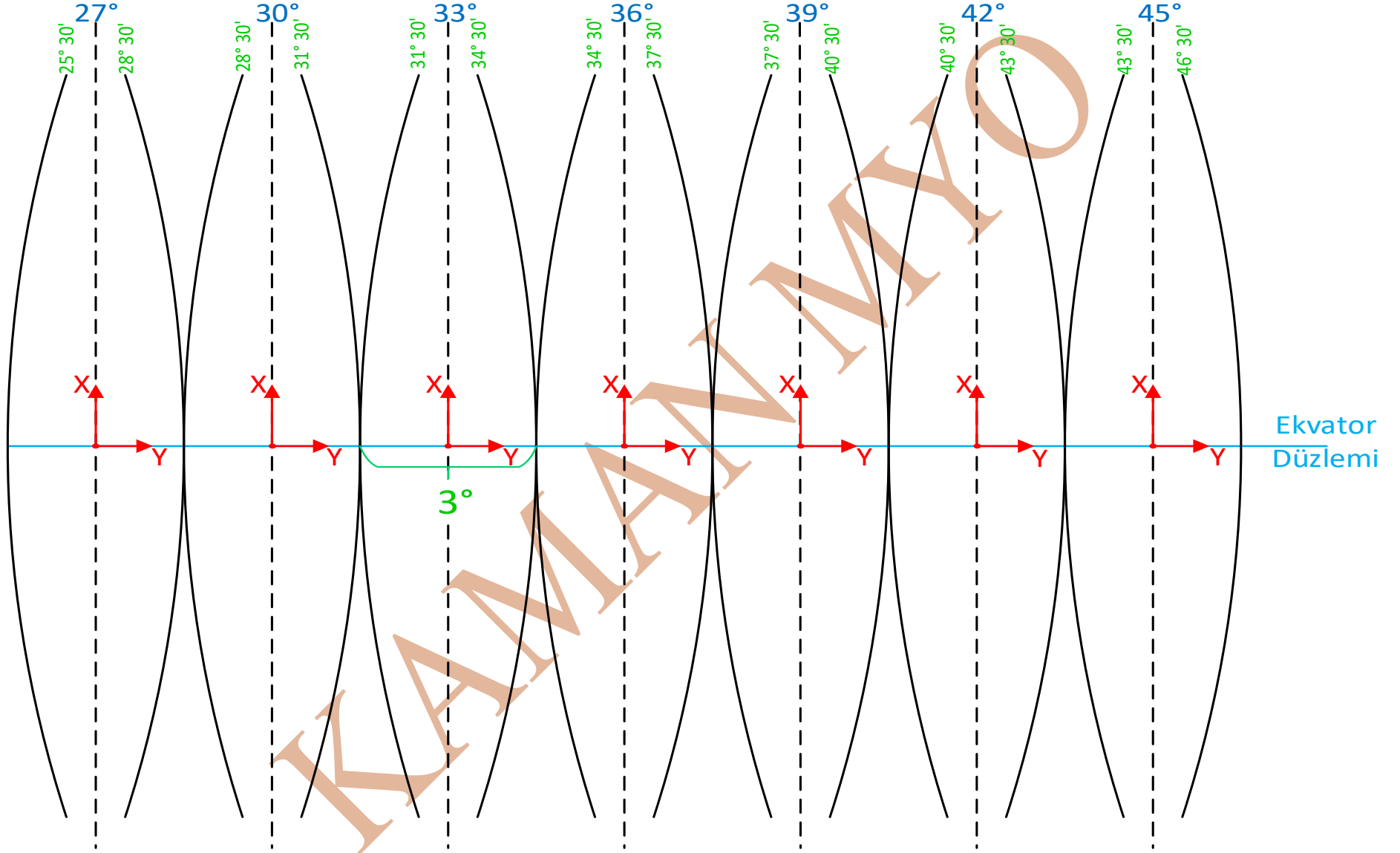
$\tan(\lambda) = \frac{Y}{X} \rightarrow \lambda = \tan^{-1}(Y/X) \rightarrow \text{Boylam}$
$p = \sqrt{(X^2 + Y^2)}$
$\tan(\varphi_{(0)}) = (Z/p) * (1 - e^2)^{-1} \rightarrow \varphi_{(0)} = \tan^{-1}((Z/p) * (1 - e^2)^{-1})$
$N_{(0)} = a^2 / \left(\sqrt{a^2 * (\cos(\varphi_{(0)}))^2 + b^2 * (\sin(\varphi_{(0)}))^2} \right)$
$h = \frac{p}{\cos(\varphi_{(0)})} - N_{(0)} \rightarrow \text{Elipsoid Yüksekliği}$
$\varphi = \tan^{-1} \left((Z/p) * (1 - e^2 * (N_{(0)}/(N_{(0)} + h))) \right)$

İkinci işlem adımında elde edilen jeodezik koordinatlar (φ , λ , h) kullanılacak projeksiyon yardımıyla harita düzlem koordinatlarına dönüştürülür. Büyük Ölçekli Harita ve Harita Bilgileri Üretim Yönetmeliği 10. Madde 1. bendinde “Bu Yönetmelik kapsamında hesaplanacak koordinatlar, **TUREF’e dayalı olarak**, GRS80 elipsoidi ve Transvers Mercator (TM) izdüşümünde üç derecelik dilim esasına göre belirlenir.” belirtmiştir. Elimizdeki jeodezik koordinatlar 3 derecelik Universal Transverse Mercator projeksiyonu (ya da yönetmelikte

olduğu gibi Transverse Mercator projeksiyonu) kullanılarak yardımcı yüzey olan dönele silindire izdüşüm koordinatları elde edilecek. Transverse Mercator (3 derecelik UTM) projeksiyonu sisteminde tüm dünyayı saran 3° aralıklı dilimler mevcuttur. Türkiye bu dilimlerden 7 tanesi arasında kalmaktadır (*Şekil 13*). *Şekil 13* hem dilimlerin dilim orta meridyenlerinin (27°, 30°, 33°, 36°, 39°, 42°, 45°) hem de dilimlerin sınır meridyenlerin boylam değerleri gösterilmiştir.

Dikkat edilirse her dilimin dilim orta meridyeninin ekvator düzlemini tasvir eden çizgiyi kestiği noktada (dilim içinde) harita düzlem koordinatlarını ifade eden kırmızı renkteki X – Y eksenleri yeniden çizilmiştir. Bunun anlamı, her dilim içinde harita düzlem koordinat sistemi yeniden başlıyor. TM projeksiyonu kullanılarak, Jeodezik koordinatlardan (jeodezik enlem ve jeodezik boylam) harita düzlemindeki izdüşüm koordinatları hesaplanırken, noktanın bulunduğu dilime göre hesaplama yapmak gerekir. Çünkü her dilim içinde harita düzlem koordinat sistemi yeniden sıfırdan başlıyor.

KAMAN MYO



Şekil 13

Jeodezik Enlem Boylam Değerlerinden, Tek Değişkenli Seriler Yöntemi ile Gauss – Kruger Koordinatlarının ve Harita Düzlem Koordinatlarının Hesaplanması

Bilinenler:

- B: Noktanın Enlem Değeri,
- L: Noktanın Boylam Değeri
- L_0 : Noktanın Bulunduğu UTM diliminin dilim orta meridyeni

İstenenler:

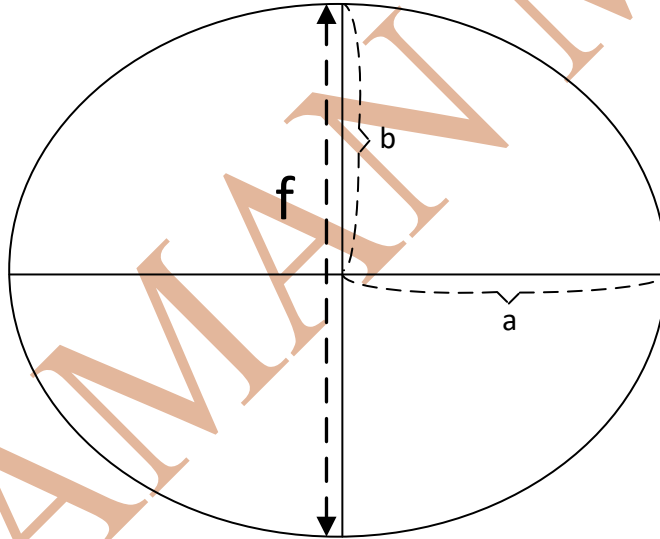
- Noktanın GRS – 80 elipsoidinde Kartezyen Gauss – Kruger Projeksiyon koordinatları olan X_g ve Y_g koordinatları.
- Noktanın TM projeksiyon sistemindeki YUKARI ve SAĞA koordinatları

GRS – 80 Referans Elipsoidi Parametreleri

a = büyük yarıçap = 6378137.000 m

b = küçük yarıçap = 6356752.3141 m

f = 1/298.257222101



$$f = \frac{a-b}{a}, \quad c = \frac{a^2}{b}, \quad e'^2 = \frac{a^2 - b^2}{b^2}, \quad e^2 = \frac{a^2 - b^2}{a^2}, \quad \rho = \frac{180^\circ}{\pi}$$

$$t = \tan(B) \quad \eta^2 = e'^2 * \cos^2(B) \quad V = \sqrt{(1 + \eta^2)} \quad N = \frac{c}{V}$$

$$A' = c * \left(1 - \frac{3}{4} * e'^2 + \frac{45}{64} * (e'^2)^2 - \frac{175}{256} * (e'^2)^3 + \frac{11025}{16384} * (e'^2)^4 - \frac{43659}{65536} * (e'^2)^5 \right) * \frac{1}{\rho}$$

$$B' = c * \left(-\frac{3}{8} * e'^2 + \frac{15}{32} * (e'^2)^2 - \frac{525}{1024} * (e'^2)^3 + \frac{2205}{4096} * (e'^2)^4 - \frac{72765}{131072} * (e'^2)^5 \right)$$

$$C' = c * \left(\frac{15}{256} * (e'^2)^2 - \frac{105}{1024} * (e'^2)^3 + \frac{2205}{16384} * (e'^2)^4 - \frac{10395}{65536} * (e'^2)^5 \right)$$

$$D' = c * \left(-\frac{35}{3072} * (e'^2)^3 + \frac{315}{12288} * (e'^2)^4 - \frac{31185}{786432} * (e'^2)^5 \right)$$

$$A_1 = \frac{N * \cos(B)}{\rho}$$

$$A_2 = \frac{N * \cos^2(B) * t}{2 * \rho^2}$$

$$A_3 = \frac{N * \cos^3(B) * (1 - t^2 + \eta^2)}{6 * \rho^3}$$

$$A_4 = \frac{N * \cos^4(B) * t * (5 - t^2 + 9\eta^2 + 4\eta^4)}{24 * \rho^4}$$

$$A_5 = \frac{N * \cos^5(B) * (5 - 18t^2 + t^4 + 14\eta^2 - 58\eta^2 t)}{120 * \rho^5}$$

$$G = A'B + B' \sin(2B) + C' \sin(4B) + D' \sin(6B)$$

Gauss – Kruger Koordinatlarının Hesabı:

$$X_g = G + A_2(\Delta L)^2 + A_4(\Delta L)^4$$

$$Y_g = A_1(\Delta L) + A_3(\Delta L)^3 + A_5(\Delta L)^5$$

UTM Projeksiyonuna Göre YUKARI Ve SAĞA Koordinatlarının Hesabı:

a) 3 derecelik UTM projeksiyonu için ölçek katsayısı $m_0 = 1$

$$YUKARI = X_g * m_0$$

$$SAĞA = Y_g * m_0 + 500000 \text{ m}$$

b) 6 derecelik UTM projeksiyonu için ölçek katsayısı $m_0 = 0.9996$

$$YUKARI = X_g * m_0$$

$$SAĞA = Y_g * m_0 + 500000 \text{ m}$$



SAĞA (harita düzlem koordinatı olarak Y koordinatı) koordinatı formülünde 500000 m eklenmesinin temel nedeni: her TM dilimi içinde dilim orta meridyeninin solunda kalan noktaların SAĞA (harita düzlem koordinatı olarak Y koordinatı) değerleri negatif olacaktır. Noktanın dilim içinde kaldığı yer dilim orta meridyeninin solunda veya sağında olmasına bakılmaksızın 500000 m eklenir.

Ülkemizde Kullanılan Koordinat Sistemleri İçin Ülkemizde ve Dünyada Belirlenmiş Standartlar

Ülkemizde Büyük Ölçekli Harita ve Harita Bilgileri Üretim Yönetmeliği, Maden Petrol İşleri Genel Müdürlüğü Mapeg Harita Standartları, GNSS sinyal alıcıları için oluşan temel standartlar, yapılacak çalışmalarda kullanılan (veya kullanılacak) koordinat sistemlerinin belirleyicisidir.

Dünya genelinde kullanılan datum değerleri, referans yüzeyleri, koordinat sistemleri, projeksiyonlar belirli standart tanımlamalarla kayıt altına alınmaktadır. Bilinen iki standart vardır. Birisi European Petroleum Survey Group (EPSG) standartları bir diğeri de kişilerin belirlediği standartlar (spatialreference.org sitesinde SR-ORG başlıklı standartlar). EPSG standartları uygulamada küresel, bölgesel, ulusal veya yerel olabilen datum, elipsoit, koordinat referans sistemlerinin ve koordinat dönüşümlerinin tanımlarının bir koleksiyonudur. EPSG Jeodezik Parametre Veri Kümesi, International Association of Oil&Gas Procedurs (IOGP - Uluslararası Petrol ve Gaz Üreticileri Birliği) Geomatik Komitesi'nin Jeodezi Alt Komitesi tarafından korunur (International Association of Oil&Gas Producers, 2023). Herhangi bir EPSG tanımı bir sayısal değer ile tekil hale getirilmiştir. Bu sayede verilen EPSG sayısal kod değeriyle tanım kolayca anılmaktadır.

Türkiye’de kullanılan datum EPSG kodları:

Datum Adı	EPSG kodu	Jeosantrik/Jeodezik/Local	Referans Elipsoidi
ITRF96	6654	Jeosantrik (dinamik) (<i>*Pratikte kullanılmıyor</i>)	GRS 1980
ED50	6230	Potsdam (Helmert Kulesi) Enlem: 52°22'51.4456" K, Boylam: 13°03'58.9283" E	International 1924
WGS 84	6326	Tüm 2 boyutlu veya 3 boyutlu jeosantrik, jeodezik WGS 84 koordinat sistemlerinin datum değeri.	WGS 84
TUREF (Türkiye Ulusal Referans Çerçevesi)	1057	Jeosantrik (ITRF96 datum 2005.0 epok)	GRS 80

Türkiye’de Kullanılan Datuların Referans Yüzeylerinin EPSG kodları:

Elipsoit Adı	EPSG kodu	a (büyük eksen yarıçapı)	f (elipsoit basıklık değeri)
GRS 1980	7019	6378137.000 m	298.257222101
International 1924	7022	6378388.000 m	297
WGS 84	7030	6378137	298.257223563

Türkiye’de kullanılan Koordinat Sistemleri EPSG kodları:

Coğrafi objelerin haritaya aktarımı için yapılan detay ölçümleri sonucu detay noktalarının elde edilen koordinatları yatayda iki boyutlu (jeosantrik iki boyutlu X – Y veya jeodezik iki boyutlu ϕ , λ) ve/veya yükseklik verisi de (elipsoit yüksekliği h veya jeoid yüksekliği H) olacak şekilde elde edilebilir. Noktaların harita düzlemine aktarılması için iki boyutlu YUKARI yönde artan ve SAĞA yönde artan koordinatlara (X – Y harita düzlem koordinatlarına) dönüştürülmesi gerekir. Bunun için projeksiyon kullanılır. Türkiye’de büyük ölçekli haritaların oluşturulmasında 3 derecelik Universal Transverse Mercator (Transverse Mercator) Projeksiyonu kullanılır.

Transverse Mercator (TM) projeksiyonu açılı koruyan, yardımcı yüzey olarak silindir kullanan bir projeksiyondur. TM projeksiyonunda tüm dünya 3 derecelik dilimlere ayrılır. *Şekil 13* TM Projeksiyonun Türkiye sınırları içinde kaldığı dilimleri temsil etmektedir. TM projeksiyonunda her dilimin dilim orta meridyeninin ekvator düzlemini kestiğini noktada 2 boyutlu koordinat sistemi, Yukarı yönde artacak şekilde (Coğrafi Kuzeye doğru artan düzlem X koordinatı) ve Sağa yönde artacak şekilde (Doğuya doğru artan düzlem Y koordinatı), yeniden oluşur. Türkiye sınırları, TM projeksiyonunun üç derecelik dilimlerinden, 27°, 30°, 33°, 36°, 39°, 42° ve 45° dilim orta meridyenine sahip dilimler içinde kalmaktadır. Bu bağlamda, TM projeksiyonuna göre, Türkiye sınırları içinde 7 ayrı koordinat sistemi oluşmaktadır.

Türkiye sınırları içindeki TM projeksiyonun göre koordinat sistemlerinin ad tanımı yapılırken, TM projeksiyonu dilimlerinin dilim orta meridyeni boylam değeri veya dilimin numara bilgisi de koordinat sistemi ad bilgisine eklenir. Bu sayede koordinat sistemlerinin adları tekil hale gelir. Dilim numarası 3° boylama sahip dilim orta meridyeni 1 olacak şekilde başlar. 27° dilim orta meridyeni olduğu dilim 9 numaralı dilim olarak adlandırılır. Dilim numarası yerine *Zone* ifadesi de kullanılır.

Tablo 5 içerisinde bazı koordinat sistemleri aynı özelliklere sahiptir, fakat farklı isimlendirmeye EPSG kaydı yapılmıştır ve farklı EPSG kodları mevcuttur. Örneğin ED50 / 3-

Coğrafi Bilgi Sistemleri II

degree Gauss-Kruger zone 9 ile ED50 / TM27 isimli iki koordinat sisteminde datum, elipsoit, dilim orta meridyenleri aynıdır. Farklı isimlendirme ile kaydedildiği için ED50 / 3-degree Gauss-Kruger zone 9 koordinat sisteminin EPSG kodu 2206, ED50 / TM27 koordinat sisteminin EPSG kodu 2319 olarak kayıt altına alınmıştır.

Tablo 4

Koordinat Sisteminin Adı	EPSG	Datum	Boyut/Eksenler	Elipsoit
TUREF	5250	1057	(3 boyutlu jeosantik X-Y-Z)	GRS80
TUREF	5251	1057	(3 boyutlu joedezik ϕ - joedezik λ Elipsoid yük. h)	GRS80
TUREF	5252	1057	(2 boyutlu joedezik ϕ - joedezik λ)	GRS80
WGS 84	4326	6326	(2 boyutlu joedezik ϕ - joedezik λ)	WGS 84
WGS 84	4978	6326	(3 boyutlu jeosantik X-Y-Z)	WGS 84
WGS 84	3857		(2 boyutlu X (easting), Y (northing))	WGS 84

Tablo 5

Koordinat Sisteminin Adı	EPSG	Datum	Elipsoit	Dilim Orta Meridyeni
ED50 / 3 - degree Gauss-Kruger zone 9	2206	ED50	International 1924	27°

Coğrafi Bilgi Sistemleri II

ED50 / 3 - degree Gauss-Kruger zone 10	2207	ED50	International 1924	30°
ED50 / 3 - degree Gauss-Kruger zone 11	2208	ED50	International 1924	33°
ED50 / 3 - degree Gauss-Kruger zone 12	2209	ED50	International 1924	36°
ED50 / 3 - degree Gauss-Kruger zone 13	2210	ED50	International 1924	39°
ED50 / 3 - degree Gauss-Kruger zone 14	2211	ED50	International 1924	42°
ED50 / 3 - degree Gauss-Kruger zone 15	2212	ED50	International 1924	45°
ED50 / TM27	2319	ED50	International 1924	27°
ED50 / TM30	2320	ED50	International 1924	30°
ED50 / TM33	2321	ED50	International 1924	33°
ED50 / TM36	2322	ED50	International 1924	36°
ED50 / TM39	2323	ED50	International 1924	39°
ED50 / TM42	2324	ED50	International 1924	42°
ED50 / TM45	2325	ED50	International 1924	45°

Coğrafi Bilgi Sistemleri II

TUREF / 3 - degree Gauss- Kruger zone 9	5269	TUREF (1057)	GRS80	27°
TUREF / 3 - degree Gauss- Kruger zone 10	5270	TUREF (1057)	GRS80	30°
TUREF / 3 - degree Gauss- Kruger zone 11	5271	TUREF (1057)	GRS80	33°
TUREF / 3 - degree Gauss- Kruger zone 12	5272	TUREF (1057)	GRS80	36°
TUREF / 3 - degree Gauss- Kruger zone 13	5273	TUREF (1057)	GRS80	39°
TUREF / 3 - degree Gauss- Kruger zone 14	5274	TUREF (1057)	GRS80	42°
TUREF / 3 - degree Gauss- Kruger zone 15	5275	TUREF (1057)	GRS80	45°
TUREF / TM27	5253	TUREF (1057)	GRS80	27°
TUREF / TM30	5254	TUREF (1057)	GRS80	30°
TUREF / TM33	5255	TUREF (1057)	GRS80	33°
TUREF / TM36	5256	TUREF (1057)	GRS80	36°
TUREF / TM39	5257	TUREF (1057)	GRS80	39°
TUREF / TM42	5258	TUREF (1057)	GRS80	42°
TUREF / TM45	5259	TUREF (1057)	GRS80	45°

EPSG kodları, bizim kullanacağımız Computer Based Design (CAD) ve Geographic Information System GIS yazılımlarında kullanılmaktadır. Ticari ve açık kaynaklı CAD ve GIS yazılımlarında temelinde, tabloları oluşturmak veya sorgulama yapmak için kullanılan Structure Query Language (SQL) dili içinde EPSG kodları kullanılmaktadır. Coğrafi objeye

ait gerek geometrik bilgiye ait veri gerekse de öznitelik bilgiye ait veri kayıtları tutulurken, objenin hangi koordinat sisteminde olduğu ya da dönüşüm yapılacağı koordinat bilgisi kayıtları EPSG kodlarıyla yapılmaktadır.

Yukarıda birden fazla farklı tablo vardır. Bu tablolar arasındaki ilişkileri anlamak için GNSS sinyal alıcısının *Şekil 13*'de $25^{\circ} 30'$ ile $28^{\circ} 30'$ meridyenleri arasında kalan bölgede ölçüm yaptığını varsayalım. Sinyal alıcının ilk bulunduğu koordinat değeri ile son bulunduğu koordinat değerlerini hem datum hem koordinat sistemi hem de EPSG kodlarıyla ifade edilirse (Bektaş):

- 1) GNSS sinyal alıcısının ilk bulunduğu koordinat değeri EPSG 4978 kodlu WGS 84 koordinat sistemindedir. Sinyal alıcının topladığı sinyallerdeki uydu koordinatları bu koordinat sistemindedir. Bu koordinat değerleri $X - Y - Z$ jeosantrik koordinatlardır.
- 2) EPSG 4978 kodlu WGS 84 koordinat sistemindeki $X - Y - Z$ jeosantrik koordinatları, EPSG 5250 kodlu TUREF koordinat sistemindeki $X - Y - Z$ koordinat sistemine dönüştürülmektedir.
- 3) TUREF (5250) datumundaki jeosantrik kartezyen koordinatlar ($X-Y-Z$) harita düzlemine aktarılmayacağı için, geometrik olarak ilk önce ilişkili olduğu TUREF (5251) koordinat sisteminde jeodezik koordinatlara (jeodezik enlem, jeodezik boylam ve elipsoidal yükseklik h) dönüştürülür. Bu işlem için *Tablo 3 Jeosantrik koordinatlardan jeodezik koordinatların hesabı* içerisinde belirtilen formüller kullanılır.
- 4) TUREF (5251) koordinat sisteminde elde edilen koordinat değerleri (Jeodezik Enlem ve Jeodezik Boylam ve elipsoid yüksekliği) Türkiye'de kullanılan projeksiyon olan Transverse Mercator projeksiyonu ile TUREF (5253) koordinat sistemindeki harita düzlem koordinatlarına (YUKARI ve SAĞA koordinatlar veya harita düzlem X ve Y koordinatları) dönüştürülür. Dönüşüm işlemlerinde kullanılan hesaplamalar dokümanın "Jeodezik Enlem Boylam Değerlerinden, Tek Değişkenli Seriler Yöntemi ile Gauss - Kruger Koordinatlarının ve Harita Düzlem Koordinatlarının Hesaplanması" başlığı altında anlatılmaktadır.

EPSG kodları ve EPSG kodlarının içeriklerine ulaşmak için iki önemli internet sitesi var:

- 1) <https://epsg.org/> → site IOGP'nin EPSG kodları için oluşturduğu site
- 2) <https://epsg.io/> → site EPSG kodları farklı veri formatlarında görüp incelememize yarıyor

Şekil 14 Epsg.org sitesinde 5255 Epsg kod numaralı Turef geosantric sistemine bağlı

The screenshot displays the EPSG website interface for the TUREF / TM33 projected CRS. The page is titled "TUREF / TM33" and shows the following details:

- Projected CRS Details [VALID]**
 - NAME: TUREF / TM33
 - CODE: 5255
 - CRS TYPE: Projected
 - USAGE: Usage Details
 - SCOPE: Cadastral, engineering survey, topographic mapping (large scale).
 - EXTENT: Turkey - 31.5°E to 34.5°E onshore
- COORDINATE SYSTEM:** Cartesian 2D CS. Axes: northing, easting (X,Y). Orientations: north, east. UoM: m
- BASE CRS:** TUREF
- Geographic 2D CRS Details [VALID]**
 - NAME: TUREF
 - CODE: 5252
 - CRS TYPE: Geographic 2D
 - USAGE: Usage Details
 - SCOPE: Horizontal component of 3D system.
 - EXTENT: Turkey
 - DATUM: Turkish National Reference Frame
 - COORDINATE SYSTEM: Ellipsoidal 2D CS. Axes: latitude, longitude. Orientations: north, east. UoM: degree
 - BASE CRS: TUREF
- Geographic 3D CRS Details [VALID]**
 - NAME: TUREF
 - CODE: 5251

Şekil 14 (International Association of Oil & Gas Producers, 2024)

The screenshot displays the EPSG website interface for the TUREF geocentric CRS. The page is titled "TUREF" and shows the following details:

- Geographic 3D CRS Details [VALID]**
 - NAME: TUREF
 - CODE: 5251
 - CRS TYPE: Geographic 3D
 - USAGE: Usage Details
 - SCOPE: Geodesy.
 - EXTENT: Turkey
 - DATUM: Turkish National Reference Frame
 - COORDINATE SYSTEM: Ellipsoidal 3D CS. Axes: latitude, longitude, ellipsoidal height. Orientations: north, east, up. UoM: degree, degree, metre
 - BASE CRS: TUREF
- Geocentric CRS Details [VALID]**
 - NAME: TUREF
 - CODE: 5250
 - CRS TYPE: Geocentric
 - USAGE: Usage Details
 - SCOPE: Geodesy.
 - EXTENT: Turkey
 - DATUM: Turkish National Reference Frame
- Datum Details [VALID]**
 - NAME: Turkish National Reference Frame
 - CODE: 1057
 - TYPE: geodetic
 - USAGE: Usage Details

PostGIS ve PostgreSQL Kullanımı

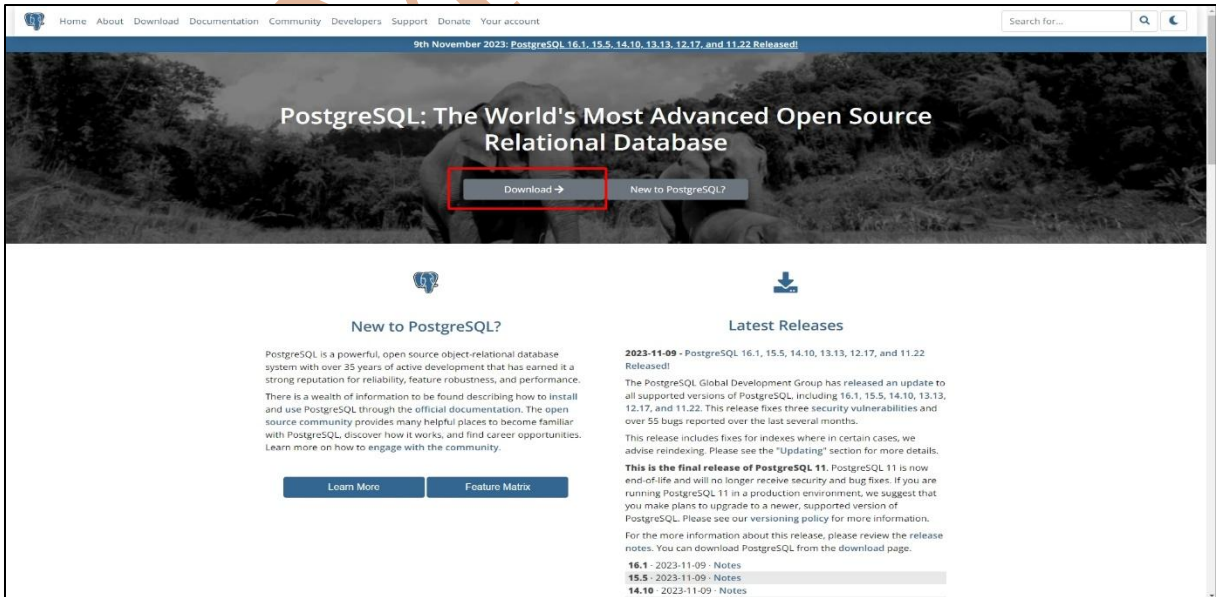
PostgreSQL, en karmaşık veri iş yüklerini güvenli bir şekilde depolayan ve ölçeklendiren birçok özellikle birlikte SQL dilini kullanan ve genişleten güçlü, açık kaynaklı bir nesne ilişkisel veri tabanı sistemidir (PostgreSQL, 2024). PostgreSQL veri tabanı yönetim sisteminde tüm işlemler Structure Query Language (SQL – Yapısal Sorgulama Dili) dili kullanılarak yapılmaktadır. Programın bilgisayara kurulumunda yüklenecek olan ek arayüzler kullanılarak veri tabanları yönetilir, sorgulamalar yapılır, veriler üzerinden analizler yapılabilir. Bu işlemlerin yapılabilmesi için SQL dilinin temel yapısına hakim olmak gerekmektedir.

PostGIS PostgreSQL veri tabanı yönetim sistemi içinde oluşturulan veri tabanları için bir eklentidir. PostgreSQL içinde PostGIS eklentisi sayesinde:

- Oluşturulan veri tabanları içinde coğrafi objelere grafik objeler kayıt altına alınabilir,
- Veri tabanı içinde kayıtlı coğrafi objeye ait öznitelikler grafik objeyle ilişkili olacak şekilde kayıt altına alınabilir,
- Coğrafi objeler arasındaki konuma dayalı ilişkiler sorgulanabilir,
- Coğrafi objeler arasında konuma dayalı analizler (intersect, overlay, proximity,...) yapılabilir.

PostgreSQL Veri Tabanı Yönetim Sisteminin İnternet Üzerinden İndirilmesi:

PostgreSQL veri tabanı yönetim sistemi yazılımını indirmek için <https://www.postgresql.org/> internet sitesi link adresi kullanılmalıdır. Açılan sayfada Download düğmesine tıklanarak yazılımın internet sitesinden indirilmesi için gerekli adrese yönlendirilir (Şekil 15).



Şekil 15

Yeni açılan internet sayfasında Download the installer link yazısı seçilerek yazılımın kurulacağı bilgisayarın işletim sistemine göre uygun PostgreSQL yazılım seçilir (Şekil 16).



Şekil 16

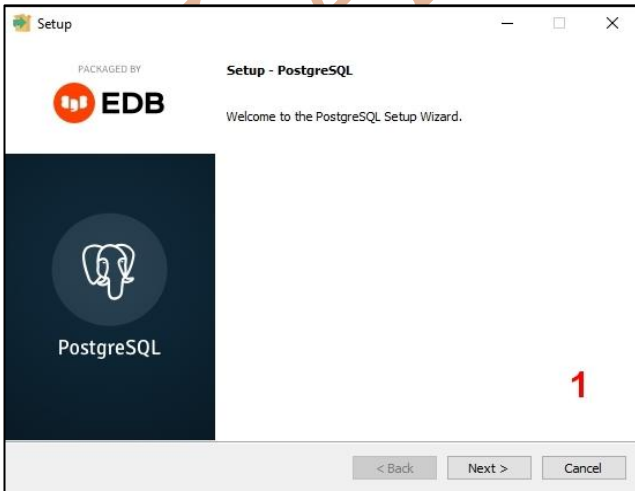
Yazılımın indirileceği sayfada en üstte yer alan versiyon son yayınlanan versiyondur (Şekil 17).



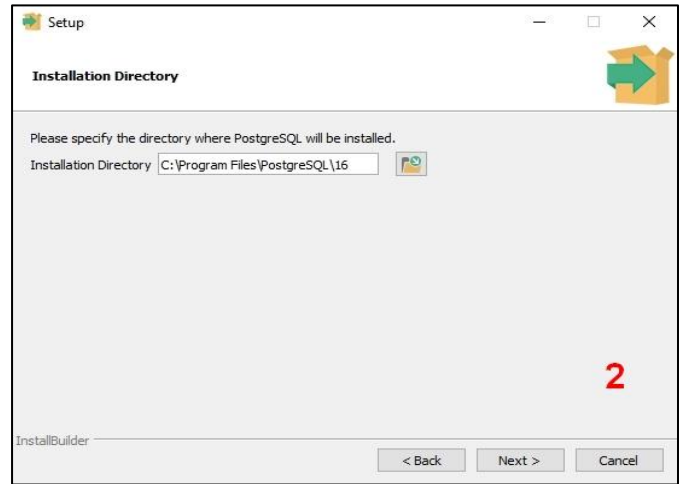
Şekil 17

PostgreSQL Veri Tabanı Yönetim Sisteminin Kurulum

PostgreSQL veri tabanı yönetim sistemi ve PostGIS eklentisinin kurulumu için resimler oluşturulmuş ve bir sonraki sayfadan itibaren numaralandırılarak listelenmiştir.

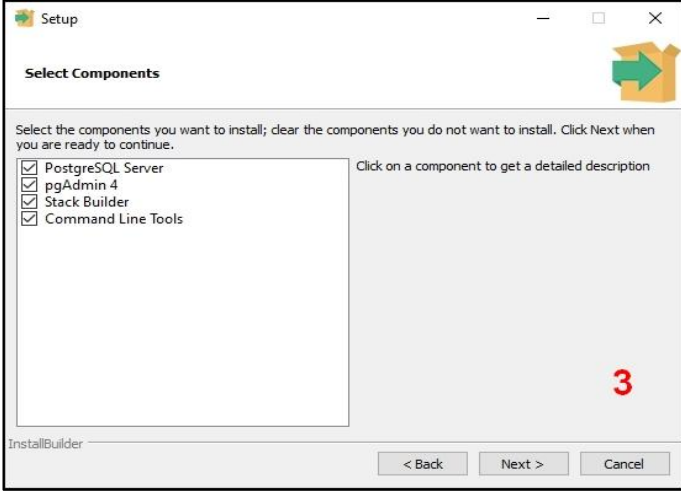


1 nolu pencerede Next tuşuna basılacak.

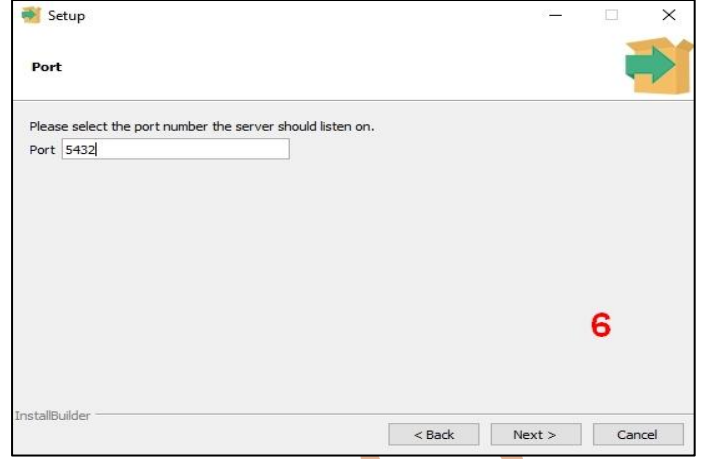


2 numaralı pencerede Next tuşuna basılacak.

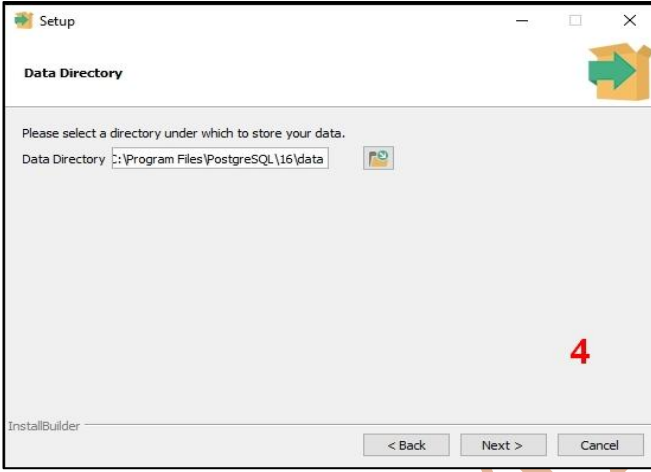
Coğrafi Bilgi Sistemleri II



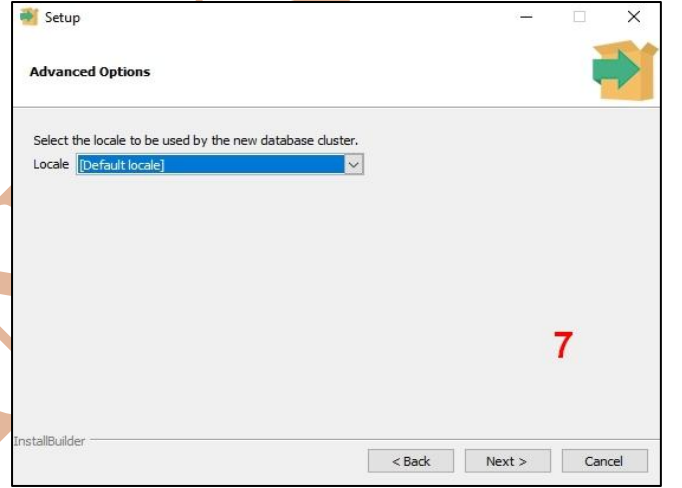
3 numaralı pencerede Next tuşuna basılacak.



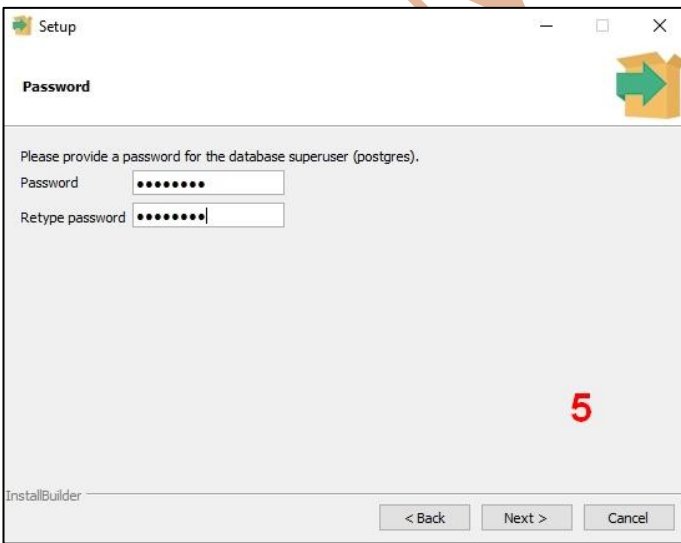
6 numaralı penceredeki port numarası değiştirilmeyecek.



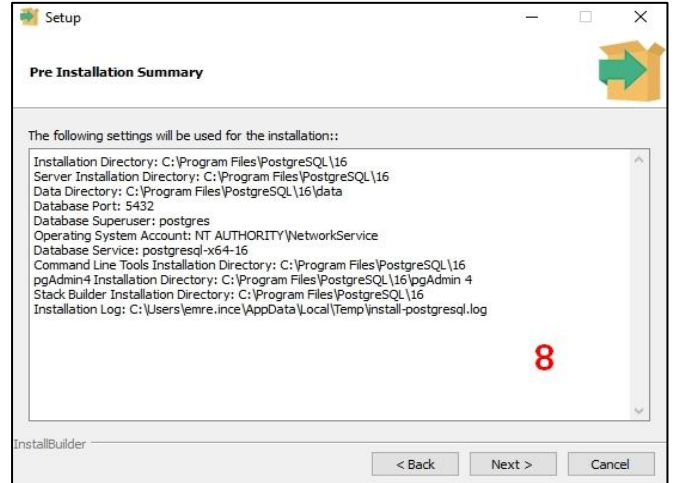
4 numaralı pencerede Next tuşuna basılacak.



7 numaralı penceredeki seçim değiştirilmeyecek.

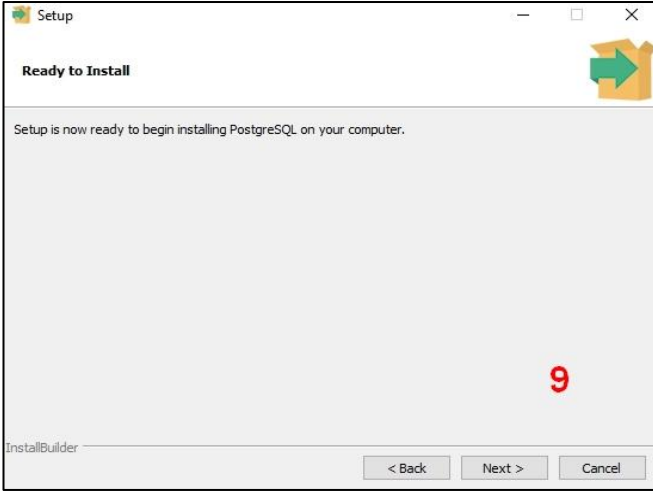


5 numaralı pencerede unutulmayacak bir şifre girilecek.

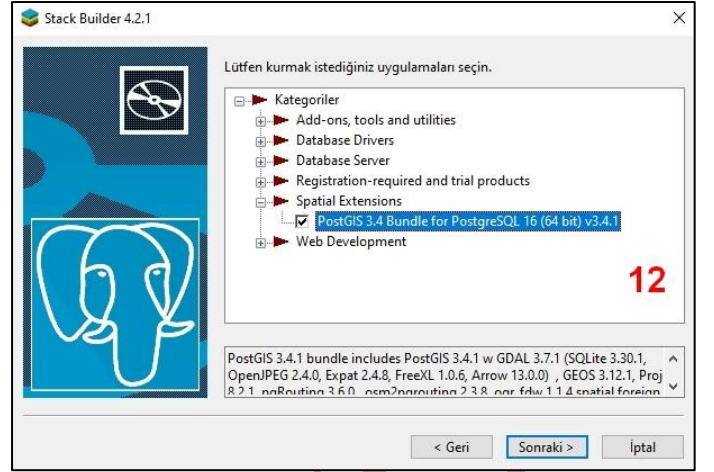


8 numaralı pencerede Next tuşuna basılacak.

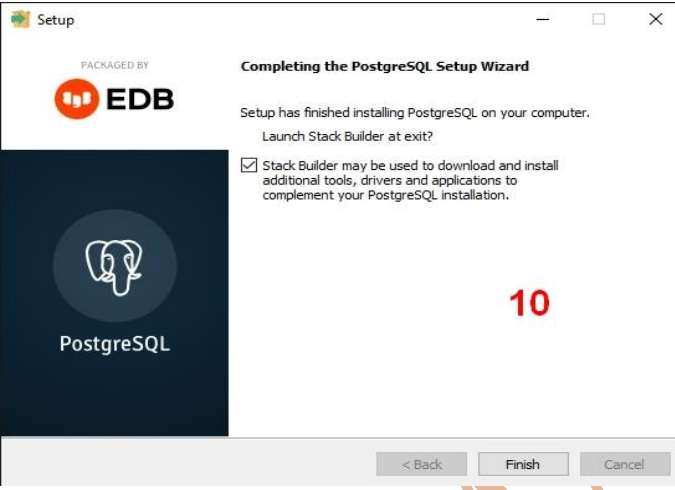
Coğrafi Bilgi Sistemleri II



9 numaralı pencerede Next tuşuna basılacak.



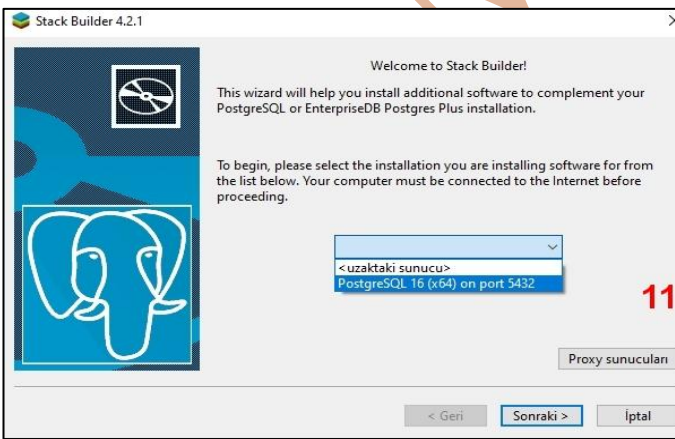
12 numaralı pencerede seçim yapılacak. Başka bir seçim işaretlenmeyecek. Sonraki tuşuna basılacak.



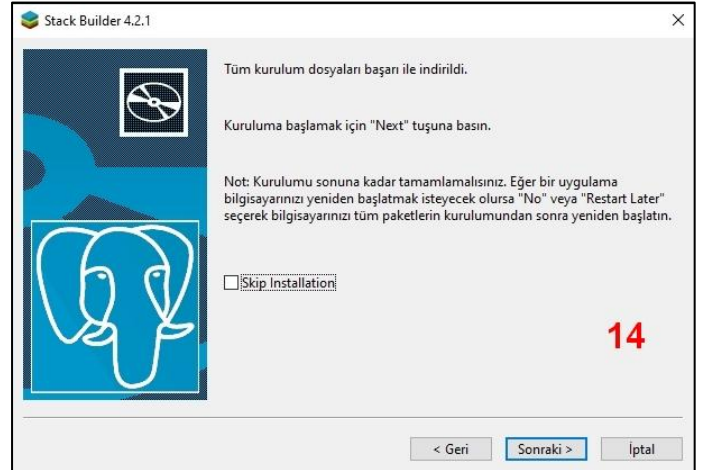
10 numaralı pencerede seçim onaylanacak ve Finish tuşuna basılacak.



13 numaralı pencerede Sonraki tuşuna basılacak.

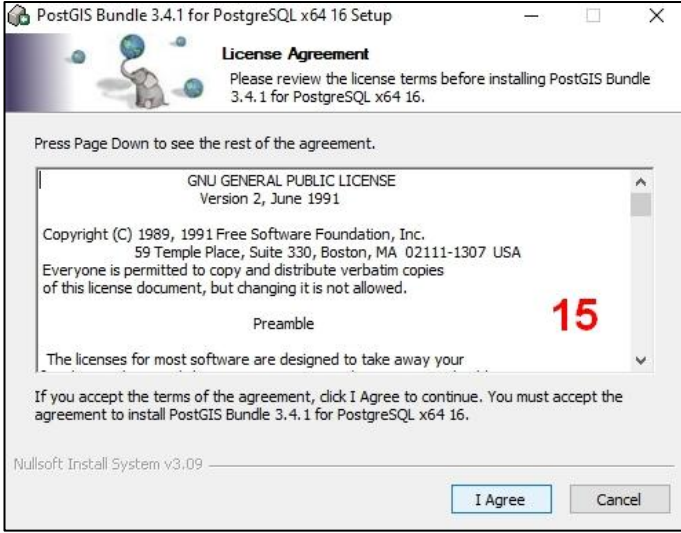


11 numaralı penceredeki seçim yapıp Sonraki tuşuna basılacak.

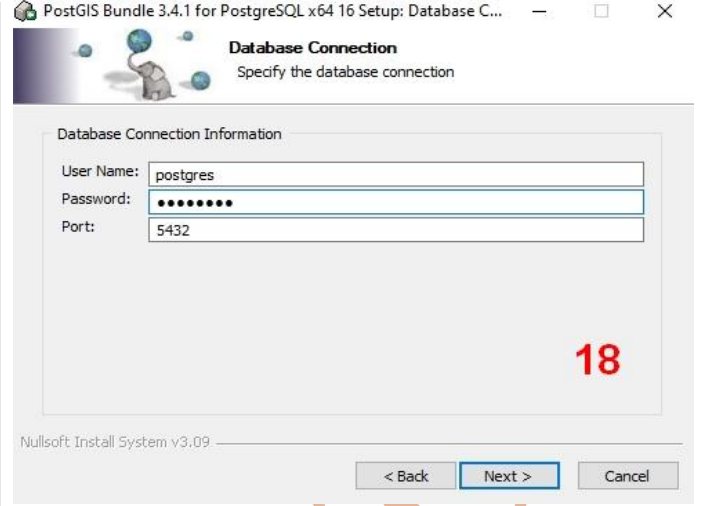


14 numaralı pencerede işaretleme yapılmayacak. Sonraki tuşuna basılacak.

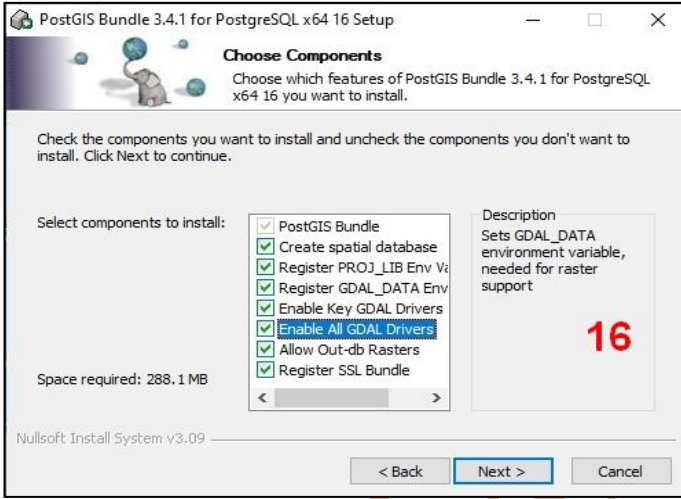
Coğrafi Bilgi Sistemleri II



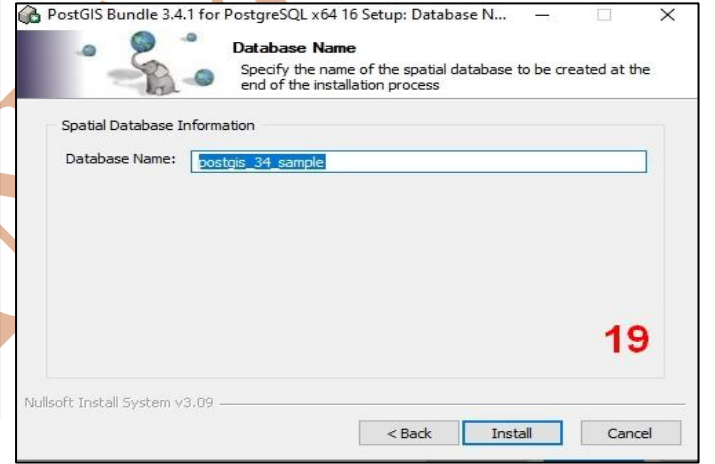
15 numaralı pencerede I agree tuşuna basılacak.



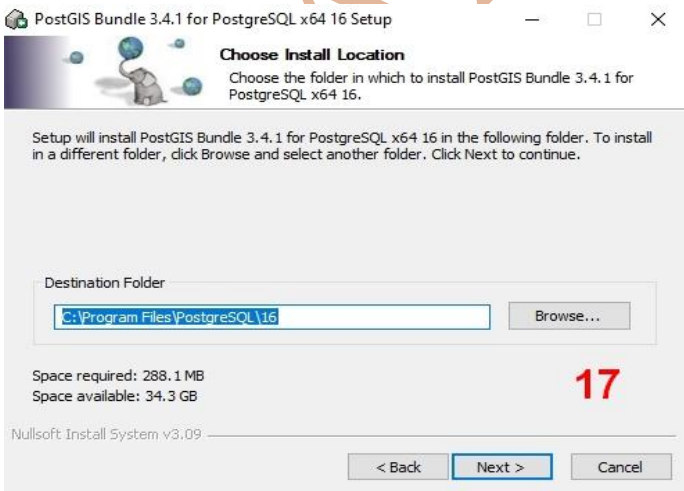
18 numaralı pencerede Password kısmına 5 numaralı pencerede girilen şifre girilecek ve Next tuşuna basılacak.



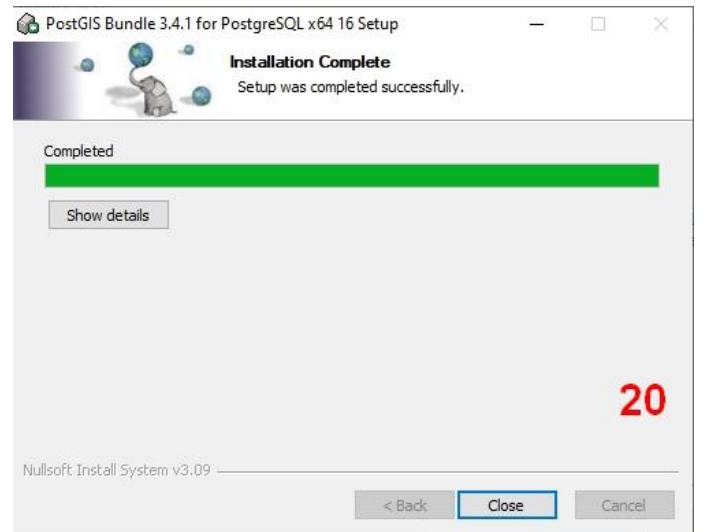
16 numaralı pencerede ki seçimler yapılp Next tuşuna basılacak.



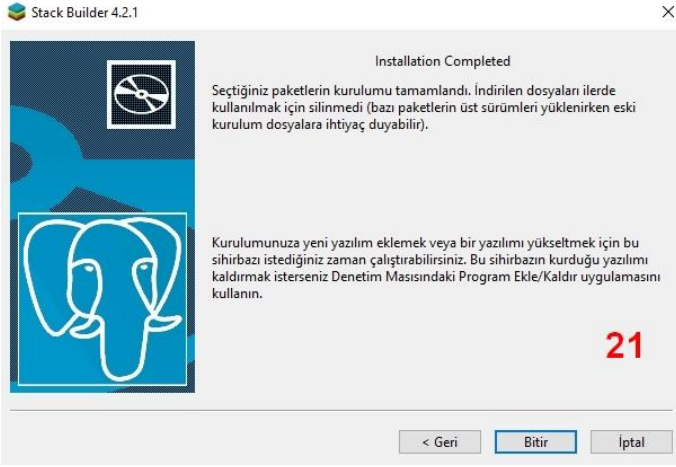
19 numaralı pencerede Install tuşuna basılacak.



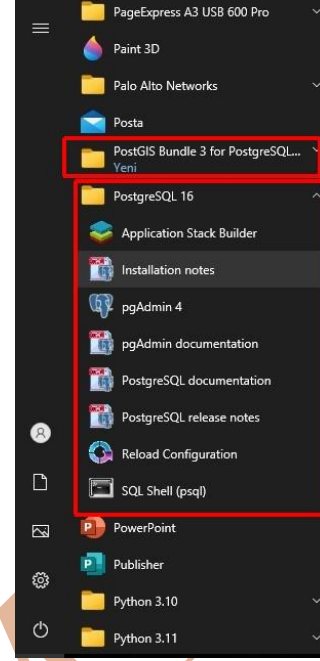
17 numaralı pencerede Next tuşuna basılacak.



20 numaralı pencerede olduğu gibi Completed yazısı gelince Close tuşuna basılacak.



21 numaralı pencerede Bitir tuşuna basılacak.

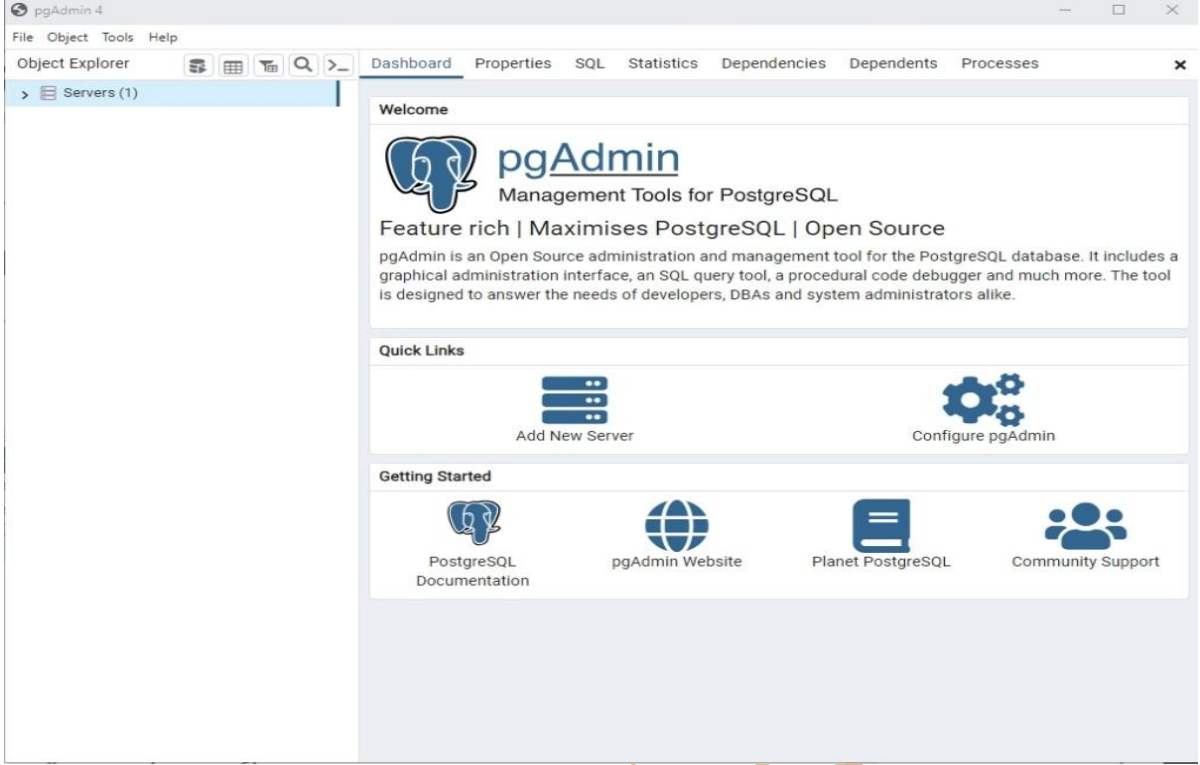


PostgreSQL ve PostGIS eklentisi bilgisayara kuruldu. Başlat menüsünden kontrol edilebilir.

PostgreSQL Veri Tabanı Yönetim Sistemine Ulaşmak İçin Pgadmin 4 Ara Yüzünün Kullanımı

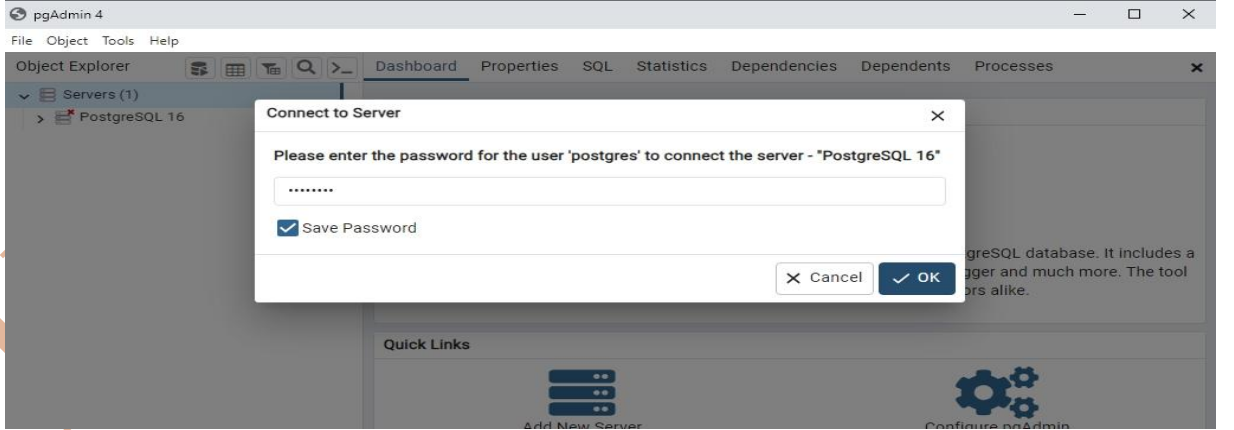
PostgreSQL yazılımının kurulumuna dair sıralamayı uyguladıysanız, kurulum sırasında pgadmin 4 yazılımı da kurulmuş olacaktır. Pgadmin 4 yazılımı PostgreSQL yazılımını yönetmek için kullanacağımız ara yüzdür.

Pgadmin 4 yazılımı ilk açıldığında çıkacak pencerede postgresql yazılımı için kurulu gelen Servers (uzaktaki bilgisayarlar) ağaç yapısı gözükecektir (Şekil 18).



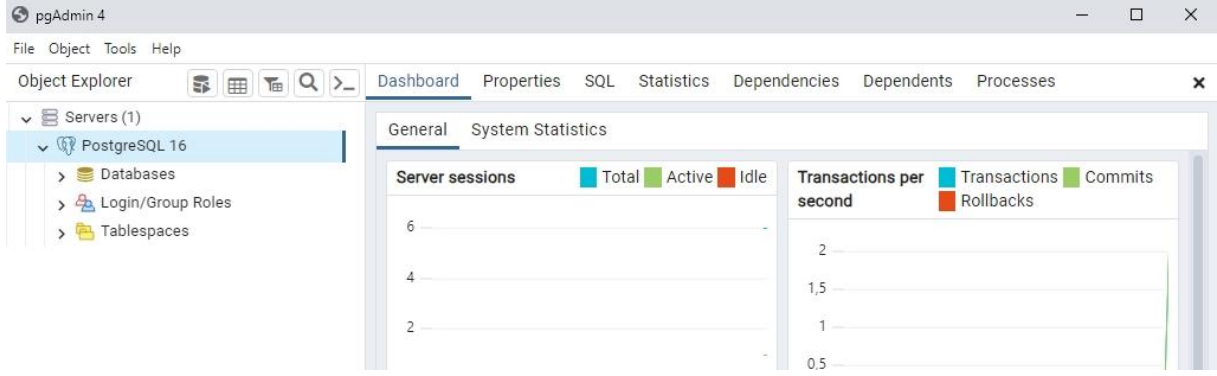
Şekil 18

Servers ağaç yapısının açılması için yanındaki ok işaretine basıldığında, PostgreSQL yazılımının kurulumunun 5 numaralı penceresinde girilen şifrenin girilmesi için giriş ekranı açılacaktır (Şekil 19). Girilen şifre Save Password seçeneği işaretlenip kaydedilirse, kullanıcının yazılıma girdiğinde bir daha şifre sorulmayacaktır.



Şekil 19

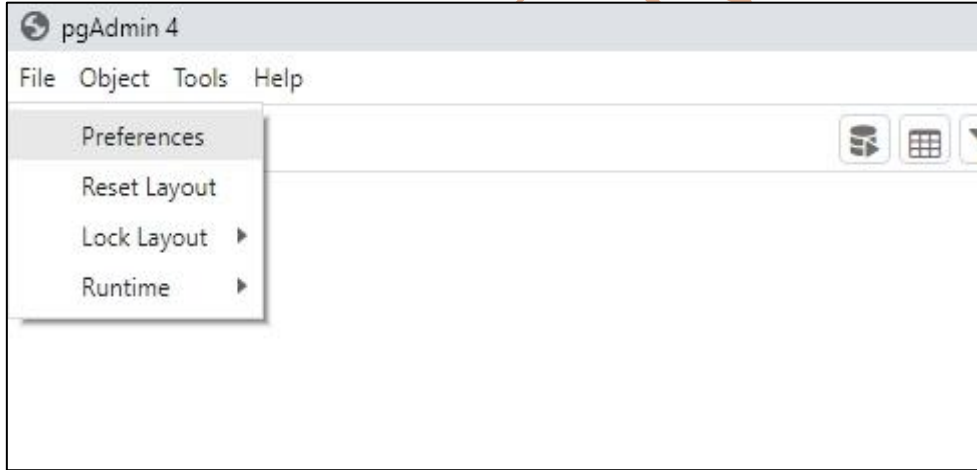
Şekil 20 şifre girildikten sonraki ekrandır. Sol tarafta Kurulum sonrası oluşturulacak Veritabanları tutacak yapı (PostgreSQL 16) görülmektedir.



Şekil 20

PgAdmin yazılımını kullanmaya başlamadan önce; yazılım içinde; kullanılan PostgreSQL veri tabanı yönetim sistemi sürümünü belirtmemiz ve sürümün çalıştırılabilir dosyasının (Exe uzantılı dosyası – Executable) dizin adresini belirtmemiz gerekli.

PgAdmin 4 yazılımında kullanılan PostgreSQL veri tabanı yönetim sisteminin sürümünü belirtmek için PgAdmin 4 yazılımı *File* (Dosya) menüsü *Preferences* (Tercihler) alt menüsü seçilir (Şekil 21).

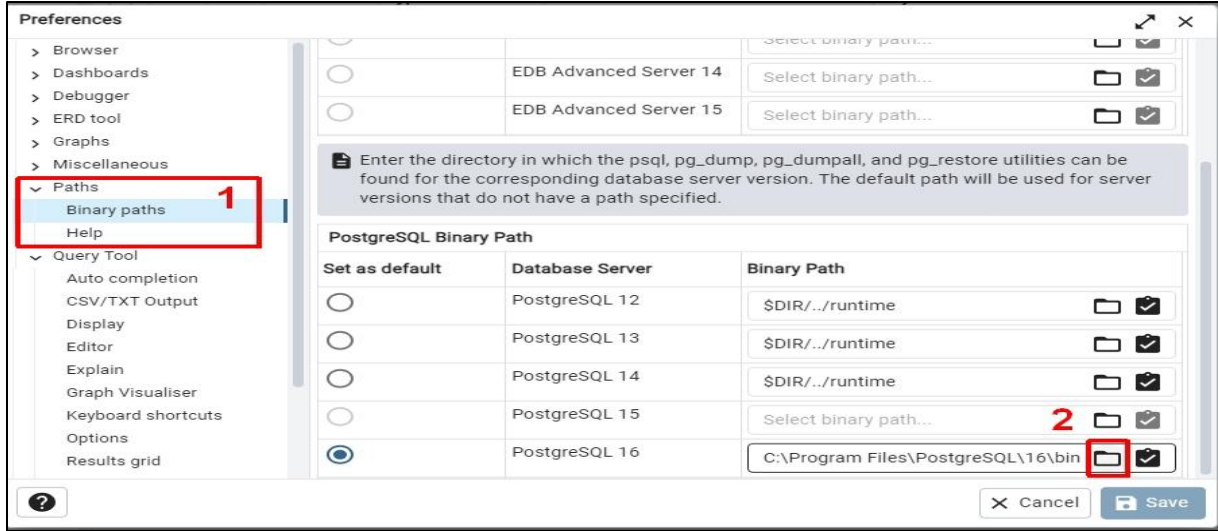


Şekil 21

Açılan *Preferences* penceresi içinde sol kısımda kalan ağaç menü yapısında *Paths* (Yollar – dizinler) alt menüsü seçilir. *Paths* alt menüsü alt menülerinden *Binary paths* seçilir (Şekil 22 1 numaralı kısım) ve pencerenin sol tarafında PostgreSQL yazılımının sürümleri listelenir. *PostgreSQL Binary Path* kısmında kullanılan sürümün olduğu kısımda dosya sembolü seçilip (Şekil 22 2 numaralı kısım) yazılımın çalıştırılabilir dosyasının dizini seçilir. Örnekte PostgreSQL yazılımının 16 sürümü kullanılmakta. Seçilmesi gereken dizin:

C:\Program Files\PostgreSQL\16\bin

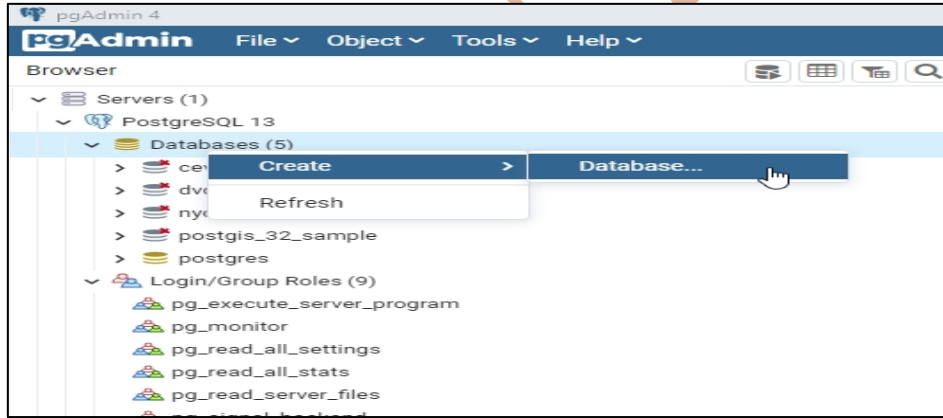
dizin adresidir.



Şekil 22

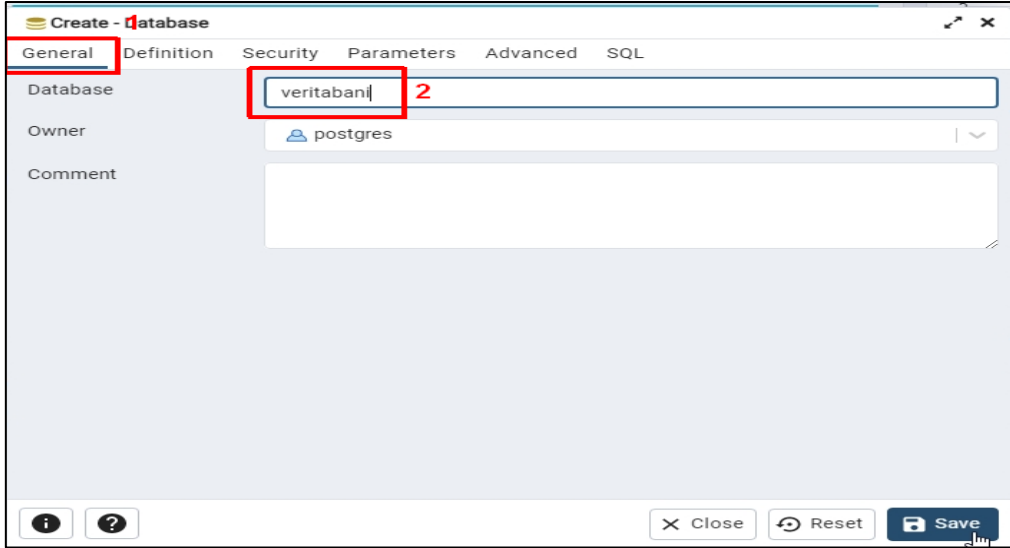
PgAdmin Programıyla PostgreSQL Veri Tabanı Yönetim Sisteminde Veri Tabanı Oluşturulma Aşamaları

PostgreSQL veri tabanı yönetim sisteminde veri tabanı oluşturabilmek için PgAdmin 4 yazılımında açılan ağaç yapısında *Databases (veri tabanları)* alt menüsü kullanılır (Şekil 23).



Şekil 23

Açılan *Create Database (veri tabanı oluştur)* penceresinde *General (Genel)* sekmesi ile açılan sayfada veri tabanının isminin girilmesi yeterlidir. Şekil 24 2 numaralı kısımda veri tabanına “veritabani” ismi verilmiştir. Oluşturacağınız veri tabanı veya öznelik sahalarında Türkçe sesli karakter kullanılmamaya çalışın. Örnekte “ı” harfi yeri “i” kullanılarak veri tabanı ismi “veritabani” olarak verilmiştir.



Şekil 24

Eğer veri tabanı SQL (Structure Query Language) sorgulama dili ile oluşturulacaksa *Create Database (Veritabanı oluştur)* penceresinde *SQL* sekmesi kullanılır (Şekil 25 1 numaralı kısım).



Şekil 25

Şekil 25 SQL sekmesiyle açılan pencerede otomatik olarak veri tabanı oluşturmak için SQL cümleciği yazılmıştır. SQL Cümleciğinde *Create Database* komutu veri tabanı oluşturmak için kullanılmaktadır. With ibaresiyle oluşturan kullanıcı Owner komutunda belirtilir. Owner kısmına yazılan postgres ifadesi PostgreSQL kurulumunda standart olarak gelen kullanıcı adıdır. Karakter veri kütüphanesi Encoding komutunda belirtilir. UTF8 karakter seti Türkçe karakterleri de içeren bir karakter setidir. Türkçe karakterler için farklı veri setleri de kullanılır. Veri tabanına bağlanacak kullanıcı limiti Connection Limit komutunda belirtilir. Kullanıcı limiti olarak -1 girilmesi veri tabanına bağlantı limitinin sonsuz olduğunu belirtir.

CREATE DATABASE veritabani

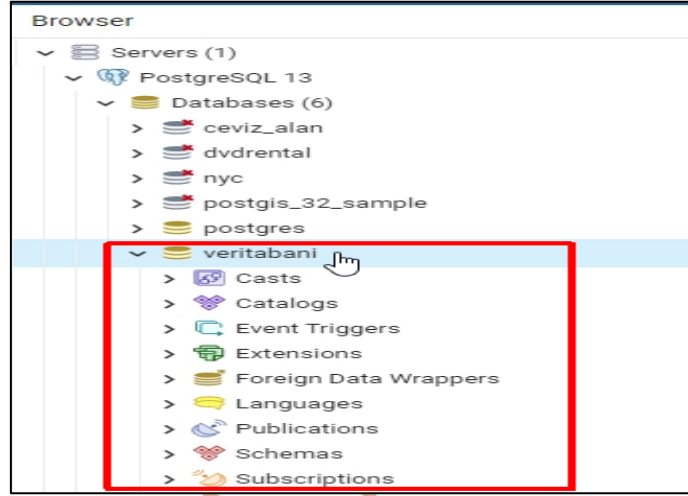
WITH

OWNER = postgres

ENCODING = 'UTF8'

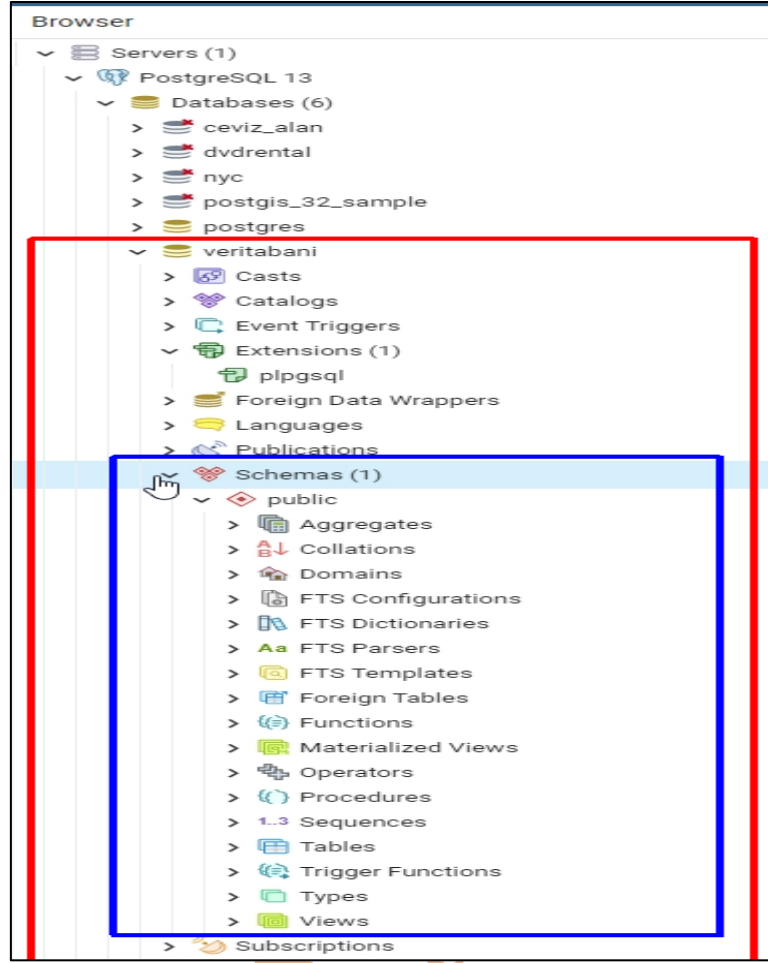
CONNECTION LIMIT = -1;

Şekil 26 veri tabanının oluştuğunu ve ağaç yapıda yerini aldığını göstermektedir. Veri tabanı yanındaki > ibaresine tıklandığında veri tabanına ait niteliklerin listelendiğini göreceksiniz (Şekil 26 kırmızı çerçeve içinde kalanlar).



Şekil 26

Schemas (Şemalar) altında *public* şeması gözükmemektedir (Şekil 27). *Public* şeması ağaç yapısı içinde, *Tables (Tablolar)*, *Functions (Fonksiyonlar)*, *Views (Görünümler)*, *Procedures (Yordamlar)* çok kullanılacak yapılar bulunmaktadır. *Public* şeması PostgreSQL tarafından otomatik oluşturulur. Kullanıcı tanımlı şemalar da oluşturulabilir.



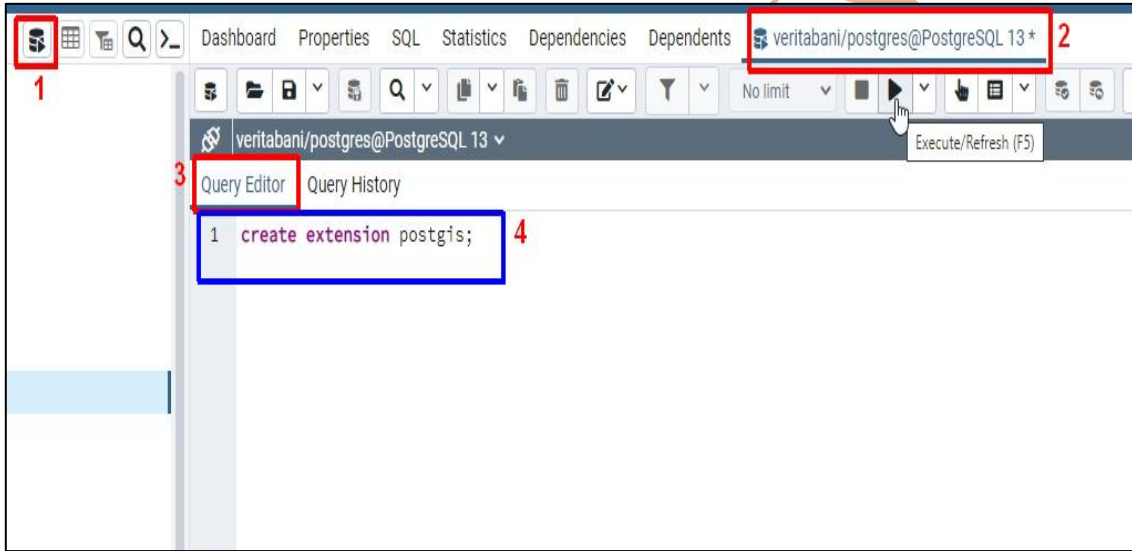
Şekil 27

Veri tabanı oluşturulduktan sonra, veri tabanın coğrafi verilerin de tutabilecek yapıda olmasının sağlanması, kullanılacak geometrik fonksiyonların veri tabanına eklenmesi için bir eklenti kurulması gerekir. Bu işlemin yapılabilmesi için SQL cümlecığı kullanılacaktır. SQL cümlecığı yazılması için de Query Editor (Sorgu Editörü) penceresi açılmalıdır (Şekil 28 kırmızı kutu içindeki düğme).



Şekil 28

Query Editor penceresinde eklentinin kurulması için ilk SQL cümlecği yazılacak:
create extension postgis; →(Şekil 29 4 numaralı kısım)



Şekil 29

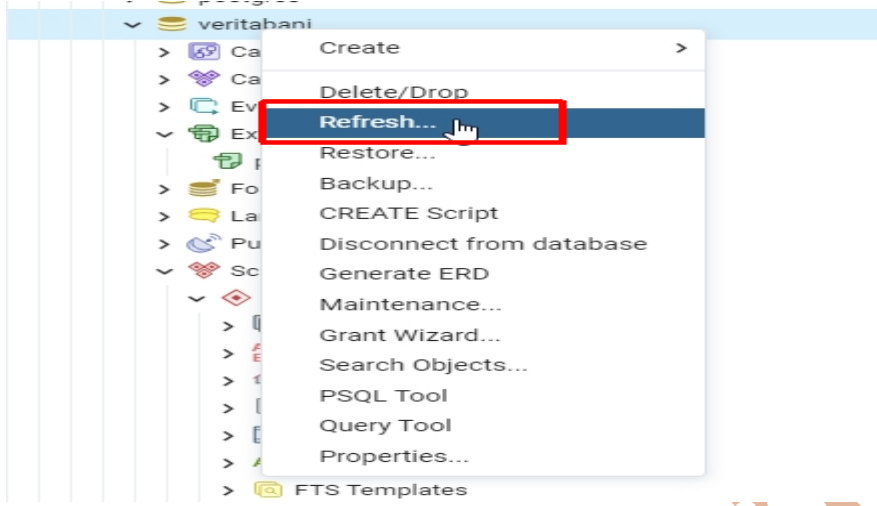
SQL sorgu cümlecği postgis eklentisinin çalıştırılması isteniyor.

Create → oluştur

Extension → Eklenti

Postgis → veri tabanında oluşacak tablolara grafik objeler ile ilişki kurabilecek özelliklerin eklenmesi, grafik objeler ile işlem yapabilecek fonksiyonların oluşturulması gibi özelliklerin eklenmesi sağlayacak eklentinin ismi *postgis*'dir. SQL cümlecğinin çalışması için Şekil 29 fare sekmesinin el olarak gözüktüğü yerdeki Execute (Çalıştır) düğmesine bir kere tıklanarak SQL cümlecği çalıştırılır.

Oluşturulan "veritabani" isimli veri tabanın ismi güncellendiğinde (Şekil 30) eklentinin çalıştırıldığı görülecektir.

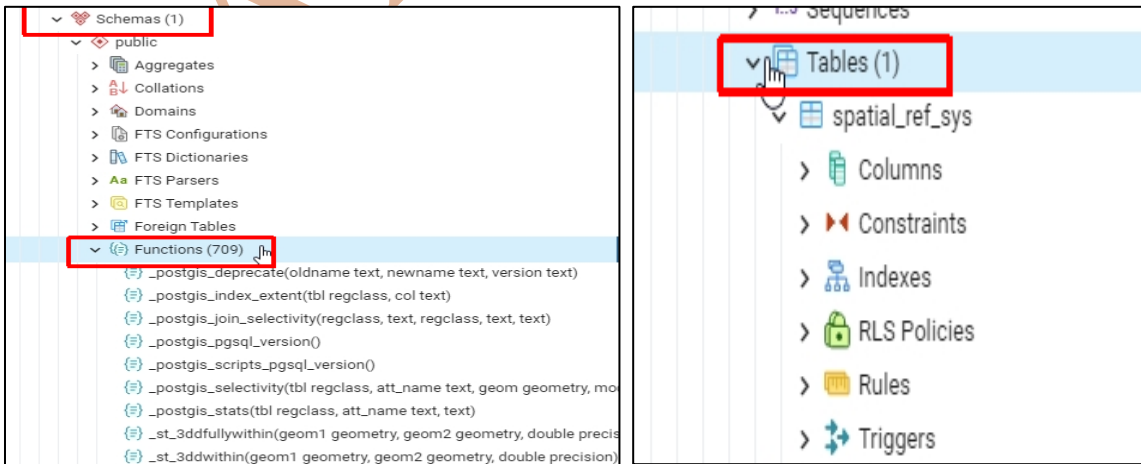


Şekil 30

Şekil 31 sol resimde postgis eklentisiyle, veri tabanına eklenen fonksiyonları, sağ resimde ise oluşturulan veri tabanı içine otomatik olarak bir adet tablonun eklendiği görülmektedir.

PostGIS eklentisi dışında birçok geometrik fonksiyon ve işlem için farklı eklentiler de gerekli olabilir. Bu eklentilerin Query Editör penceresinde yazılıp çalıştırılması için gerekli kodlar aşağıda listelenmiştir.

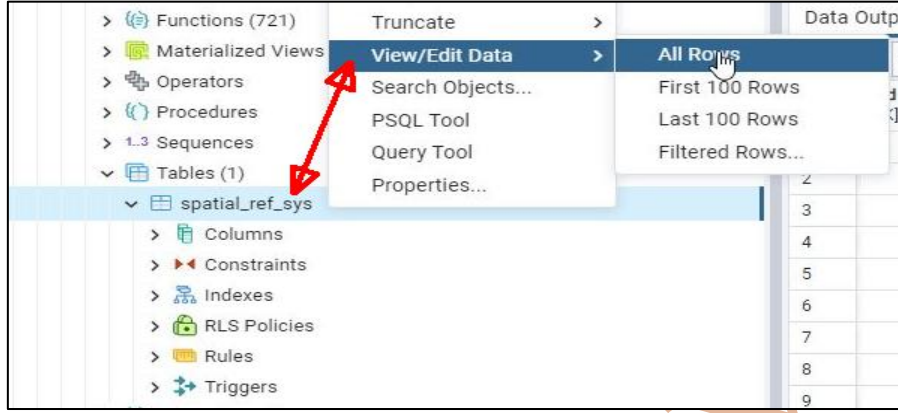
- CREATE EXTENSION postgis_raster;
- CREATE EXTENSION address_standardizer;
- CREATE EXTENSION address_standardizer_data_us;
- CREATE EXTENSION fuzzystmatch;
- CREATE EXTENSION postgis_tiger_geocoder;
- CREATE EXTENSION postgis_topology;



Şekil 31

Tables (tablolar) kısmı altında otomatik olarak “spatial_ref_sys” tablosu eklenmiştir (Şekil 32). Bu tablo daha önceden tanımlanmış olan koordinat sistemlerinin adları, EPSG

kodları, koordinat sistemine ait özelliklerin tutulduğu tablodur. Bu tablo içindeki kayıtları görebilmek için tablonun üzerine sağ tuş ile tıklandıktan sonra açılan pencerede *View/Edit Data (Veriyi Gör/Düzenle)* menüsü ve All Rows alt menüsü seçilir (Şekil 32).



Şekil 32

Ekranında “spatial_ref_sys” isimli tablo açılır. Açılan tablo içinde *srid* (*spatial reference identifier – konumsal referans tanımı*) öznitelik sahasında EPSG kodları bulunmaktadır. Liste içinde Türkiye’de kullanılan koordinat sistemlerinden 5253 EPSG kodlu TUREF / TM27 adlı koordinat sistemi bulunduğu *srtxt* öznitelik sahasında koordinat sisteminin özellikleri görülecektir (Şekil 33 kırmızı kutu içine alınmış kısım).

srid	auth_name	auth_srid	srtxt
[PK] integer	character var	integer	character varying (2048)
5253	EPSG	5253	PROJCS["TUREF / TM27",GEOGCS["TUREF",DATUM["Turkish_National_Reference_Frame",SPHEROID["GRS 1980",6378137,298.2572

Şekil 33

Kullanılan PosGIS versiyonunun Öğrenilmesi

Yapılan işlemlerde daha önce kurulu olan PostGIS eklentisinin versiyonunu öğrenmek için iki ayrı ifade kullanabiliriz. *ifade 1* ve *ifade 2* kullandığımız PostGIS eklentisinin sürümünü verecektir. *ifade 1* eklentiye dair detaylı bir açıklama verirken, *ifade 2* kullanılan eklentinin daha özet bir açıklaması verecektir.

ifade 1

```
select postgis_full_version();
```

ifade 2

```
select postgis_version();
```

Şekil 34 kullanılan PostGIS eklentisinin 3.3 sürümünde olduğunu belirtiyor.



Şekil 34

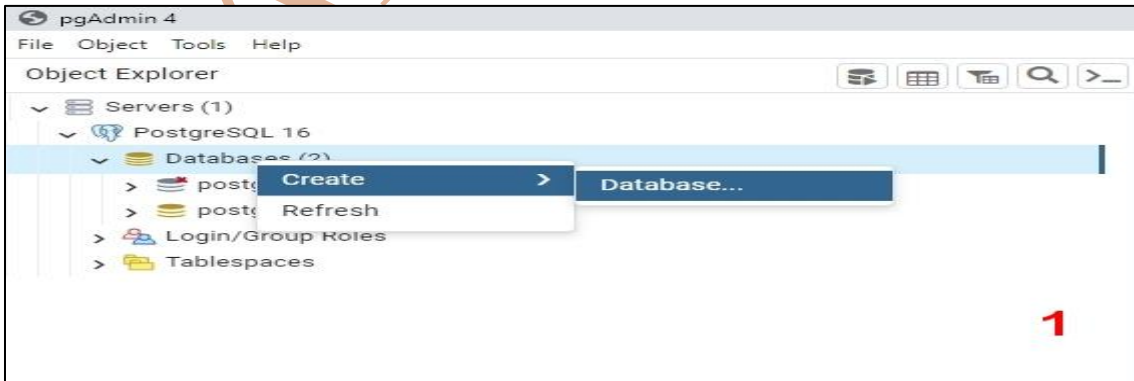
Structure Query Language (SQL)

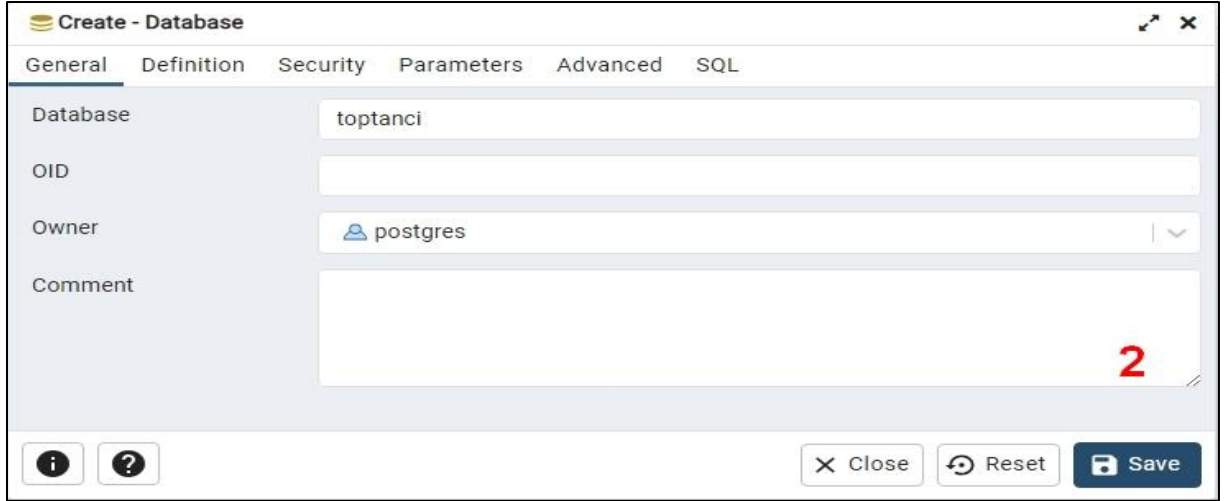
PostgreSQL ile PostGIS yapısını öğrenmeden önce sorgulama dili olarak kullanılan Structure Query Language (SQL – yapısal sorgulama dili) temel kavramlarını öğrenmemiz gerekli. Bu işlemi yaparken, daha sonra üzerinde coğrafi objeler için de çalışacağımız, PostgreSQL veri tabanı yönetim sistemi yazılımını kullanacağız.

Örnek Tablonun PostgreSQL Veri Tabanı Yönetim Sistemine Eklenmesi

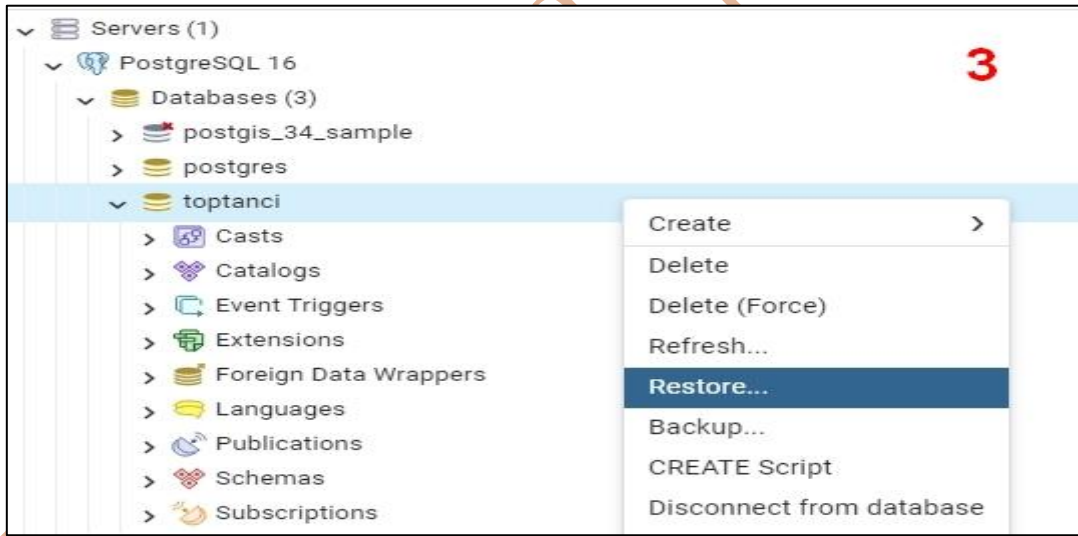
Çalışma tablolarını ve örneklerini W3Schools internet sitesinin SQL için oluşturduğu tablolar ve örnekler kullanılacaktır. Örnek ve tablolar <https://www.w3schools.com/sql/> sitesinden temin edilebilir.

Örnek tabloları PostgreSQL veri tabanı yönetim sistemine eklemek için ilk önce boş bir veritabanı oluşturulması gerekmektedir. SQL yapısal sorgulama dilinin temel komutları öğrenileceği için oluşturulacak boş veri tabanı içine PostGIS eklentisinin yüklenmesine gerek yoktur. Oluşturulacak veri tabanının adı “toptanci” olarak girilecek.





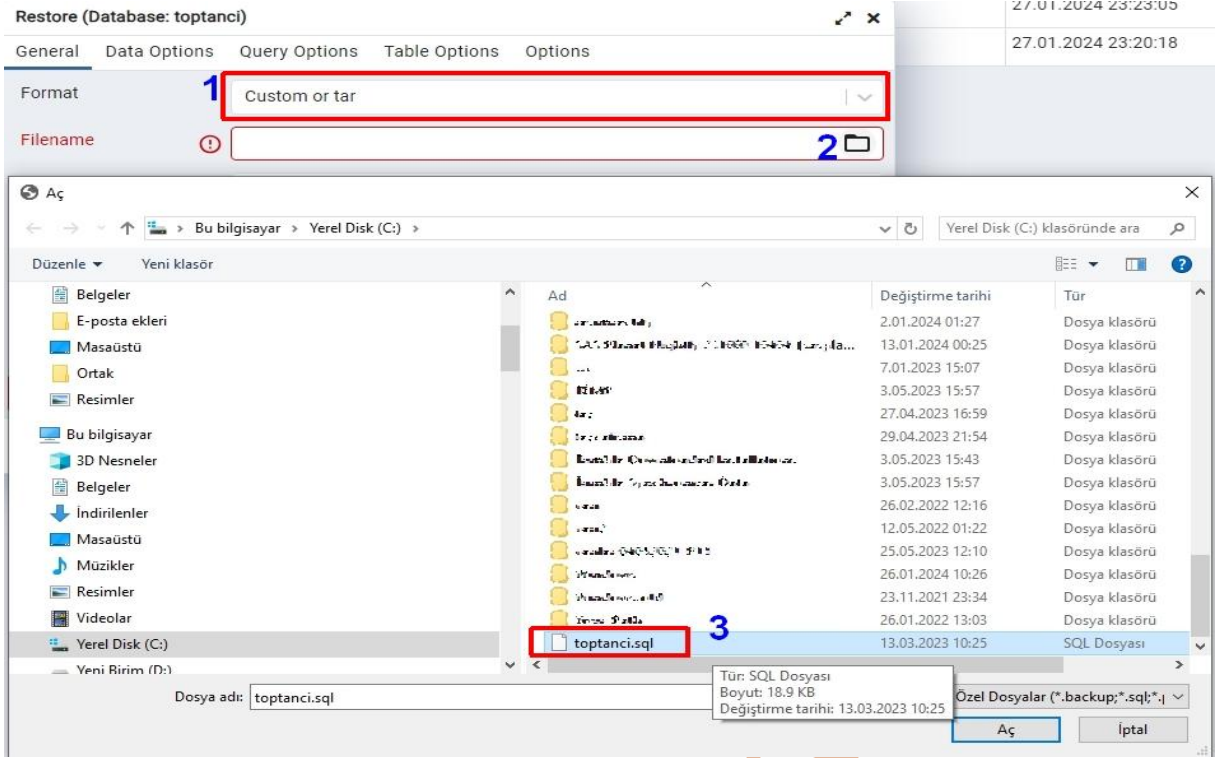
Veri tabanı oluşturulduktan sonra içerisine hazırlanmış tablolar aktarılacak. Yapılan örnekte tabloların tutulduğu dosyanın uzantısı “Sql” dosya formatındadır. Dosyayı veri tabanına eklemek için oluşturulan veri tabanının üzerine sağ tuşla tıklayıp açılan alt menülerden *Restore* seçilir (Şekil 35).



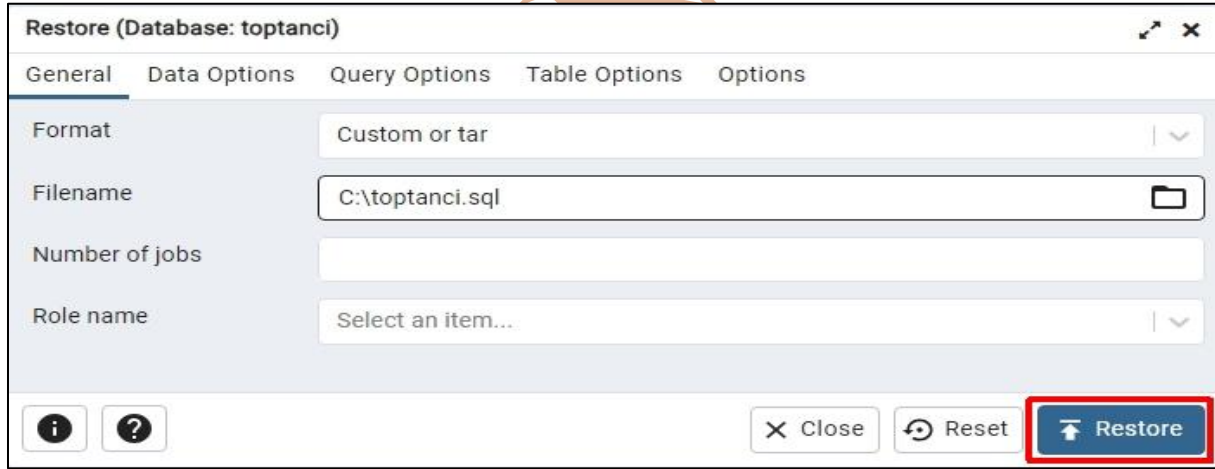
Şekil 35

Restore başlıklı pencere açılır. Sql uzantılı dosya uzantılı dosyayı açabilmek için Format kısmından *Custom or tar* dosya tipi seçilir (Şekil 36 1), ardından Filename kısmındaki dosya sembolüne tıklanarak (Şekil 36 2), dosyanın olduğu dizinden dosya seçilir (Şekil 36 3).

Coğrafi Bilgi Sistemleri II

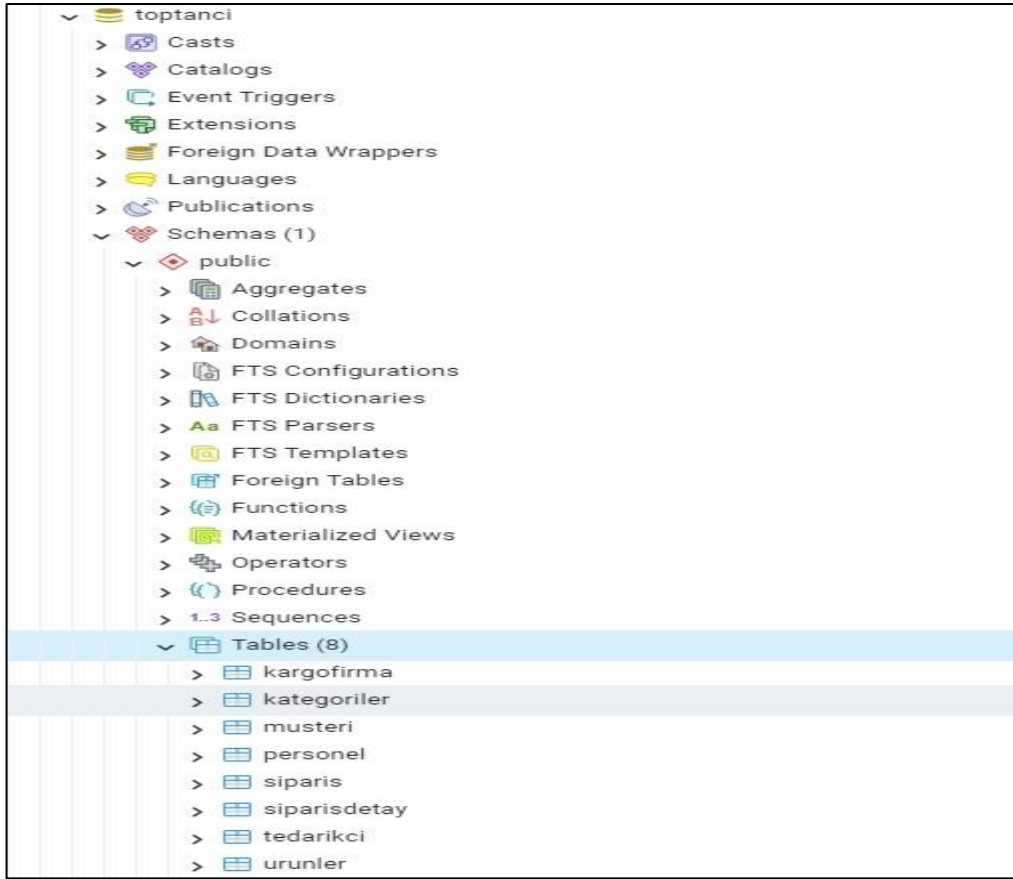


Şekil 36



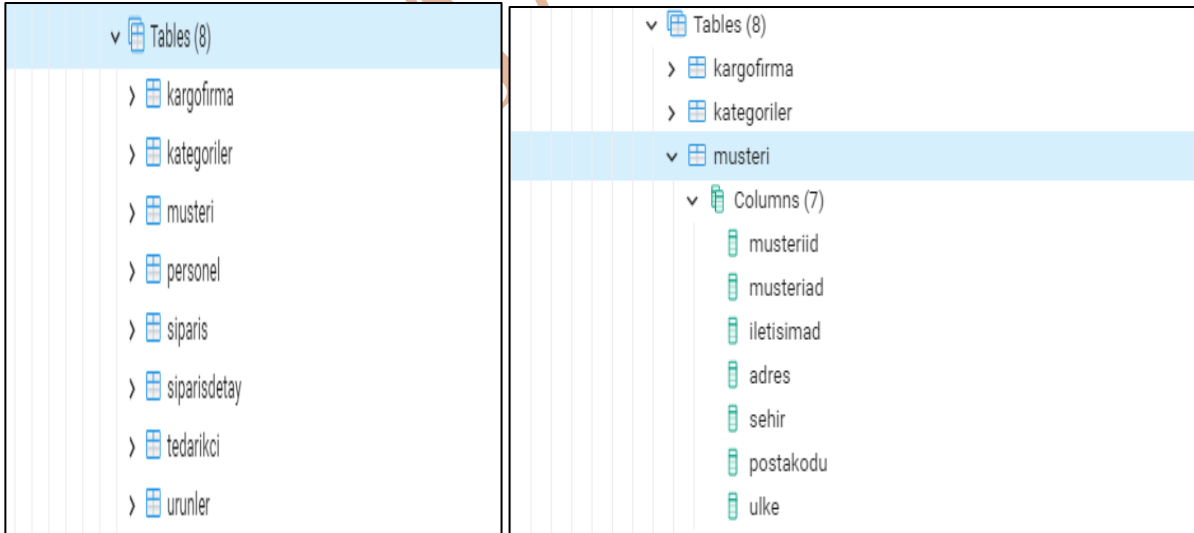
Şekil 37

toptanci adlı veri tabanına sağ tuş ile tıklayıp açılan pencerede *Refresh* alt menüsü seçildikten sonra **toptanci** veri tabanı ağaç yapısı altında Schemas (şemalar) ve public şemasının da altındaki Tables (tablolar) onun altında eklenen tablolar gözükecektir.



Şekil 38

Şekil 39 “toptanci” adında veri tabanı içinde oluşturulmuş tablolar ve kullanılacak tablolardan “müşteri” isimli tablonun öznelik sahaları görülmektedir.



Şekil 39

Structure Query Language – Yapısal Sorgulama Dili Komutları

SQL ifadeleri kurularak oluşturulacak sorgu cümlelerinde bazı basit kriterler dikkate alınmalıdır:

Her bir SQL dilinde kullanılan ifadelerin (kelimelerin) küçük harfle veya büyük harf duyarlılığı yoktur.

SELECT = select → her iki ifade de aynı işlemi yapar.

Her SQL cümleciğinin sonunda noktalı virgül kullanılır. ifade 3 örnek SQL cümleciğidir. Cümleciğin son parametresi “;” olarak görülmektedir. “;” ifadesi cümleciğin bitimini bildirmektedir.

ifade 3

```
Select * from müşteri;
```

SQL sorgu cümleleri yazılırken belirtilen iki kriter dikkate alınarak cümleler yazılmalıdır. Yapılacak örneklerde basit kriterlere de değinilecektir.

Select ifadesi

Select ifadesi tablo içinden veri seçmek için kullanılır. ifade 4 ve ifade 5 aynı SQL cümleciğidir. ifade 4 tek bir satırda yazılmıştır, ifade 5 iki ayrı satırda yazılmıştır. Dikkat edilmesi gereken “;” ifadesi SQL cümleciğinin bitimini göstermektedir. ifade 5’de ikinci satırın sonunda “;” kullanılması ikinci satırı ayrı bir cümle haline getirmiyor.

Select ifadesinden sonra yazılan öznitelik saha isimleri, from ifadesinden sonra yazılan tablo içinden, seçilecek olan sahalara ifade eder.

ifade 4’

```
Select öznitelik1,öznitelik 2, ... From Tablo_adi;
```

ifade 5

```
Select öznitelik1,öznitelik 2, ...
```

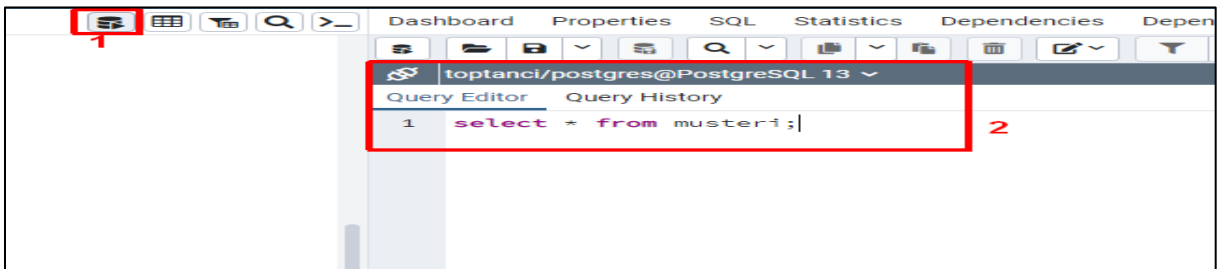
```
From Tablo_adi;
```

ifade 6’de Select ifadesinden sonra * ifadesi kullanılmıştır. * ifadesi, tüm sahalardan seçilmesi için kullanılır.

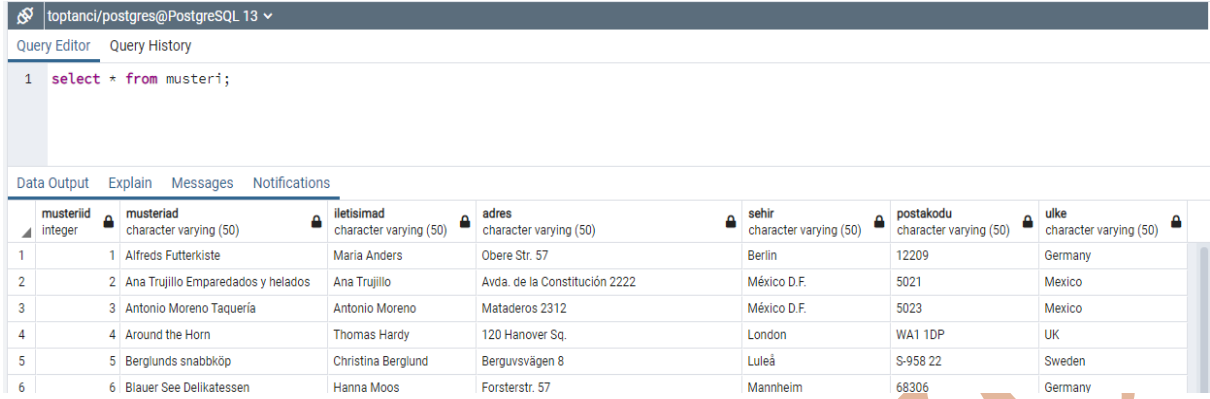
ifade 6

```
Select * From Tablo_adi;
```

Şekil 40 “müteri” tablosundaki tüm sahalardan seçilmesine dair SQL cümlesinin örneği vardır. Şekil 41 SQL cümlesinin sonuç seçim listesidir.



Şekil 40



The screenshot shows a PostgreSQL Query Editor window with the following query and results:

```
1 select * from musteri;
```

musteriid	musteriad	iletisimad	adres	sehir	postakodu	ulke
integer	character varying (50)	character varying (50)	character varying (50)	character varying (50)	character varying (50)	character varying (50)
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitución 2222	México D.F.	5021	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.	5023	Mexico
4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
5	Berglunds snabbköp	Christina Berglund	Berguvägen 8	Luleå	S-958 22	Sweden
6	Blauer See Delikatessen	Hanna Moos	Forsterstr. 57	Mannheim	68306	Germany

Şekil 41

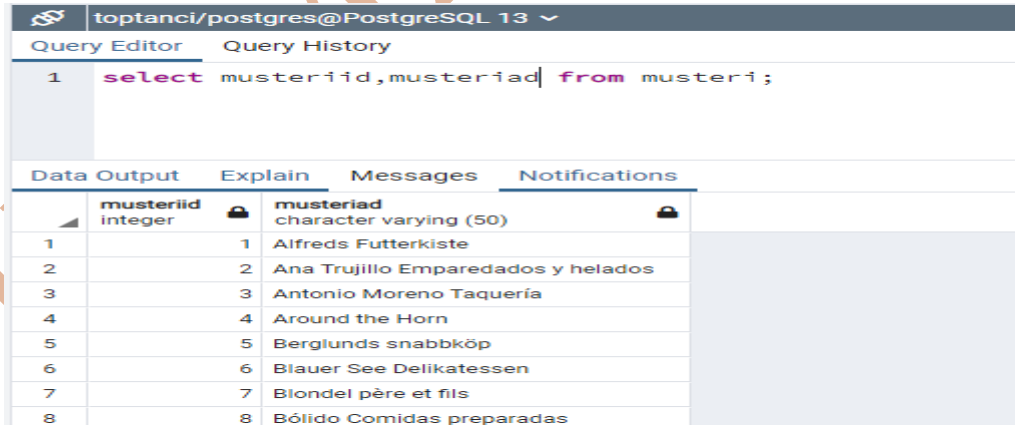


Şekil 40 örneğinde sorguyu yazabilmek için şekilde 1 numara ile gösterilen düğme ile *Query Tool* penceresi açılır ve 2 numara ile gösterilen kısımda sorgu cümlesi yazılır.

ifade 7 örneğinde sadece iki adet özneliğin seçilmesi sorgu örneği vardır. İstenilen *musteriid* sahası ile *musteriad* sahasının seçimi yapılmıştır. Kullanılan tablo adı *from* kısmında belirtilmiştir. İşlemin sonuç örneği Şekil 42’de görülmektedir.

ifade 7

Select musteriid,musteriad from musteri;



The screenshot shows a PostgreSQL Query Editor window with the following query and results:

```
1 select musteriid,musteriad from musteri;
```

musteriid	musteriad
integer	character varying (50)
1	Alfreds Futterkiste
2	Ana Trujillo Emparedados y helados
3	Antonio Moreno Taquería
4	Around the Horn
5	Berglunds snabbköp
6	Blauer See Delikatessen
7	Blondel père et fils
8	Bólido Comidas preparadas

Şekil 42

DISTINCT ifadesi

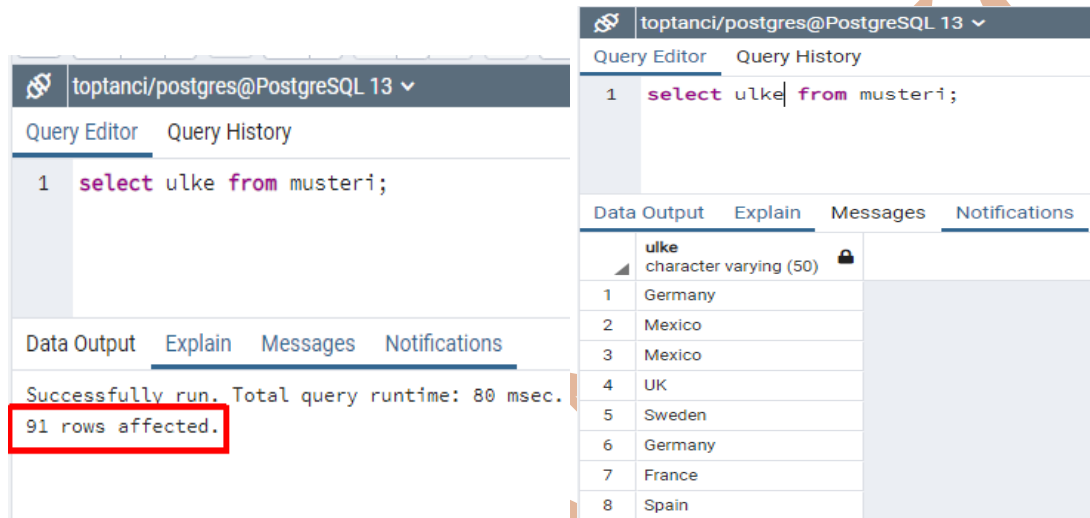
Distinct ifadesi istenilen saha veya sahalarda içindeki **tekrarlanan veriyi gruplandırarak** seçmek için kullanılır. Dikkat edilirse seçim için kullanılacağı için kendi başına kullanılmıyor, *Select* ifadesiyle beraber kullanılıyor.

ifade 8 kullanımında “müşteri” tablosundaki sadece “ülke” sahasının kayıtlarını seçilmesi istenmiştir.

ifade 8

select ulke from musteri;

Şekil 43 seçim ve sonuçları görülmektedir. *Şekil 43* Sağ resimde seçim işlemi ve seçim sonuçlarından sadece 8 tanesi görülmektedir. *Şekil 43* sol resimdeyse, SQL sorgusu sonrası 91 kaydın seçildiği gözükmemektedir. “müşteri” tablosunda “ülke” sahası içinde tekrarlı olup olmadığına bakılmaksızın (*mesagges* sekmesinde) 91 kayıt seçilmiştir.



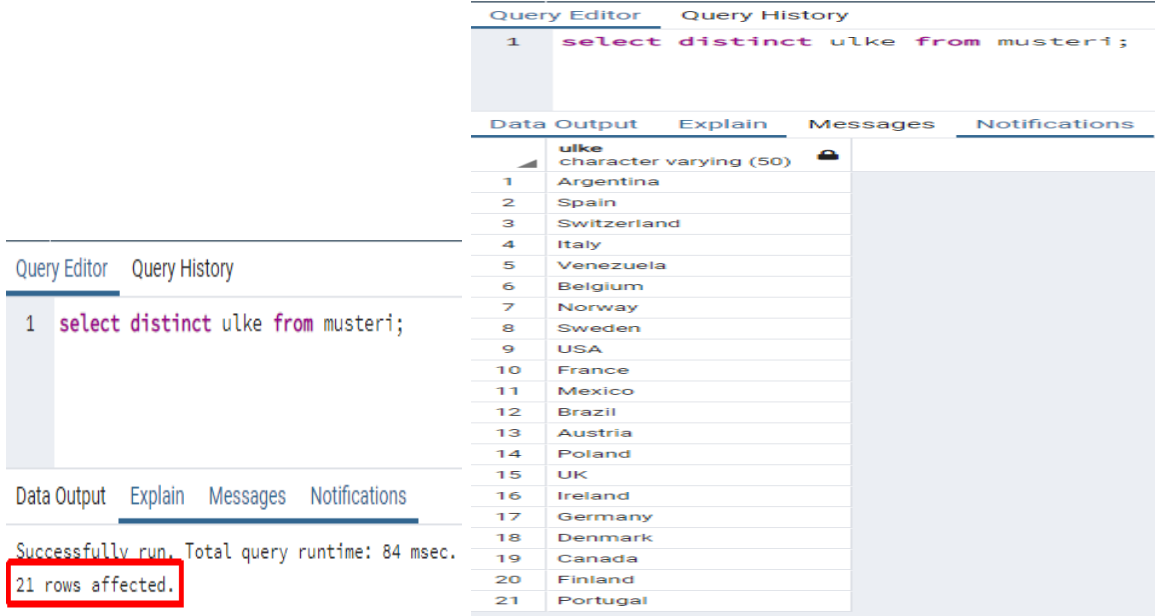
Şekil 43

ifade 9 sorgu cümlesi incelendiğinde *distinct* ifadesi Select ifadesiyle beraber kullanılmıştır.

ifade 9

select distinct ulke from musteri;

Şekil 44 sorgu örneği görülmektedir. Sağ resimde sorgu ve sonuçları tekrarsız bir şekilde gruplandırılarak gösterilmiştir. Sol resimde ise “müşteri” tablosu içindeki “ülke” sahası içindeki kayıtlar “distinct” ifadesi kullanılarak gruplandırılırsa (*mesagges* sekmesinde) toplamda 21 kayıt seçilmekte olduğunu gösteriyor.



Query Editor Query History

```
1 select distinct ulke from musterij;
```

Data Output Explain Messages Notifications

	ulke character varying (50)
1	Argentina
2	Spain
3	Switzerland
4	Italy
5	Venezuela
6	Belgium
7	Norway
8	Sweden
9	USA
10	France
11	Mexico
12	Brazil
13	Austria
14	Poland
15	UK
16	Ireland
17	Germany
18	Denmark
19	Canada
20	Finland
21	Portugal

Successfully run. Total query runtime: 84 msec.
21 rows affected.

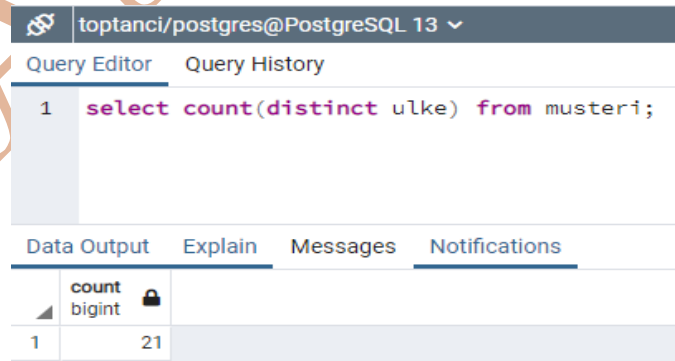
Şekil 44

Count() ifadesi

Count() ifadesi bir fonksiyon olarak çalışır. Seçilen kayıtların sayısını geri döndürmek için kullanılır. Count() ifadesini seçilen sayısını bulmak için kullanacaksak, Select ibaresi kısmında kullanmamız gerekir. *ifade 10* Count() fonksiyon ibaresinin yazımına örnektir. *ifade 9* örneğinde *Distinct* ibaresiyle saha içindeki verileri gruplandırmıştık, sonuç seçimlerin sayısı message sekmesinde gözükmekteydi. Count() seçimler değil seçim sonucu toplam sayıyı geri döndürdü ve sonuç sayı değeri Data Output sekmesinde gözükmektedir.

ifade 10

Select Count(Distinct ulke) From musterij;



toptanci/postgres@PostgreSQL 13

Query Editor Query History

```
1 select count(distinct ulke) from musterij;
```

Data Output Explain Messages Notifications

	count bigint
1	21

Successfully run. Total query runtime: 84 msec.
1 row affected.

Şekil 45

Avg() ifadesi

Avg() ifadesi, sayısal bir öznitelik sahasında ortalama almak için kullanılır. Select ile kullanılır. *Şekil 46* sağ resimde “urunler” tablosu ve “urunler” tablosunun sahalari görülmektedir.

Query Editor		Query History	
1 <code>select urunadi, fiyat from urunler;</code>			
Data Output	Explain	Messages	Notifications
urunadi character varying (50)		fiyat double precision	
1	Chais		18
2	Chang		19
3	Aniseed Syrup		10
4	Chef Anton's Cajun Seasoning		22
5	Chef Anton's Gumbo Mix		21.35
6	Grandma's Boysenberry Spread		25
7	Uncle Bob's Organic Dried Pears		30
8	Northwoods Cranberry Sauce		40
9	Mishi Kobe Niku		97
10	Ikura		31
11	Queso Cabrales		21
12	Queso Manchego La Pastora		38
13	Konbu		6
14	Tofu		23.25

Şekil 46

Şekil 46 sol resimde “urunadi” sahası ve “fiyat” sahası seçiminin bir kısmı görülmektedir. Tüm ürünlerin “fiyat” sahasının ortalaması alınmak istenirse ifade 11’de yer alan kod kullanılmalıdır. Şekil 47 sorgu sonucu görülmektedir.

ifade 11

`select avg(fiyat) from urunler;`

Query Editor		Query History	
1 <code>select avg(fiyat) from urunler;</code>			
Data Output	Explain	Messages	Notifications
avg double precision			
1	28.8663636363637		

Şekil 47

Sum() ifadesi

Sum() ifadesi Count() ve Avg() ifadeleri bir fonksiyon ifadedir. Sum() ifadesi, sayısal bir sahadaki değerlerin toplamını bulmak için kullanılır. Şekil 48 “siparisdetay” tablosunun öznitelik sahasları (sol resim) ve tablo sahaslarındaki verilerin belirli bir kısmının görüntüsüdür (sağ resim.).

Query Editor		Query History	
1 select * from siparisdetay;			
Data Output	Explain	Messages	Notifications
siparisdetayid integer	siparisid integer	urunid integer	miktar integer
1	1	10248	11
2	2	10248	42
3	3	10248	72
4	4	10249	14
5	5	10249	51
6	6	10250	41
7	7	10250	51
8	8	10250	65
9	9	10251	22
10	10	10251	57
11	11	10251	65
12	12	10252	20
13	13	10252	33
14	14	10252	60
15	15	10253	31

Şekil 48

Örnek: “siparisdetay” tablosunda “miktar” sahasının toplam değeri istenmektedir.

select sum(miktar) from siparisdetay;

Query Editor		Query History	
1 select sum(miktar) from siparisdetay;			
Data Output	Explain	Messages	Notifications
sum bigint			
1	12743		

Şekil 49

Where İfadesi

Where ifadesinde belirtilen kısıtlamalar ile seçilecek kayıtlar filtrelenir. ifade 12 Where kullanım örneği verilmiştir.

ifade 12

Select saha_adi

From tablo_adi

Where kısıtlama

ifade 13 “müşteri” tablosundaki tüm kayıtlar arasında ülke sahasında “Mexico” kaydı olanların seçilmesini sağlar. Bu sorgu ile kayıtların ülke sahasında ve Mexico kısıtıyla filtrelenmesi sağlanır. Şekil 50 seçim sonuçlarını gösterir.

ifade 13

*Select * from musteri where ulke = 'Mexico';*

1 `select * from musteri where ulke='Mexico';`

musteriid integer	musteriad character varying (50)	iletisimad character varying (50)	adres character varying (50)	sehir character varying (50)	postakodu character varying (50)	ulke character varying (50)
1	2 Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitución 2222	México D.F.	5021	Mexico
2	3 Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.	5023	Mexico
3	13 Centro comercial Moctezuma	Francisco Chang	Sierras de Granada 9993	México D.F.	5022	Mexico
4	58 Pericles Comidas clásicas	Guillermo Fernández	Calle Dr. Jorge Cash 321	México D.F.	5033	Mexico
5	80 Tortuga Restaurante	Miguel Angel Paolino	Avda. Azteca 123	México D.F.	5033	Mexico

Şekil 50

ifade 13 ile *ifade 14* karşılaştırıldığında iki sorgu da aynı görünmektedir, fakat *ifade 14* kısıtlama yapılırken, Mexico kısıtlaması, küçük harfle başlamaktadır. *ifade 14* sorgusunun sonucu Şekil 51'de görülmektedir. Hiçbir seçim yapılamamıştır. Kısıtlamalar yazılırken öznitelik sahasındaki veriler dikkate alınarak yapılmalıdır.

ifade 14

*Select * from musteri where ulke = 'mexico';*

1 `select * from musteri where ulke='mexico';`

musteriid integer	musteriad character varying (50)	iletisimad character varying (50)	adres character varying (50)	sehir character varying (50)	postakodu character varying (50)	ulke character varying (50)
----------------------	-------------------------------------	--------------------------------------	---------------------------------	---------------------------------	-------------------------------------	--------------------------------

Şekil 51

Where ifadesinde kısıtlama yapılırken karşılaştırma operatörleri de kullanılır. Tablo 6 *Where* ifadesinde kullanılacak karşılaştırma operatörleri verilmiştir.

Tablo 6

Karşılaştırma Operatörler	Tanımları
=	Eşittir
>	Büyüktür
<	Küçüktür
>=	Büyük veya Eşittir (büyük eşit)
<=	Küçüktür veya Eşittir (küçük eşit)
<>	Eşit değil, != olarak da kullanılır
BETWEEN	belirli bir aralıkta kısıtlama yapmak için kullanılır.
LIKE	Benzer olanları tespit etmek için kullanılır.

IN	Bir sütun için birden çok olası değer belirtmek için kullanılır.
----	--

Tablo 7’de Where ifadesinde birden fazla kısıtlamanın aynı anda sınırlama yapmasını sağlamak için kullanılan mantıksal operatörler bulunmaktadır.

Tablo 7

Mantıksal Operatörler	Tanımları
And	İki kısıtlama da ya doğru olacak ya da yanlış olacak
Or	İki kısıtlamadan biri doğru olsa dahi işlem olur
Not	Eğer koşul doğru değilse işlem olur.

Örnek: Müşteri tablosundaki “ulke” sahasında “Germany” verisi ve “sehir” sahasında “Berlin” verisi olan kayıt isteniyor. Sorgu yapılırken Where ifadesinde AND mantıksal operatörü kullanılacak.

ifade 15 istenilen sorgu cümlesidir. İlk kısıtlama *ulke = 'Germany'* ile yapılmıştır. İkinci kısıtlama *sehir = 'Berlin'* ile yapılmıştır. Her iki kısıtın da beraber işlemi için and kullanılmıştır. Yapılan sorgu sonucu tek bir kayıt seçilmiştir (Şekil 52).

ifade 15

*select * from musteri where ulke = 'Germany' and sehir = 'Berlin';*



musterid	musteriad	iletisimad	adres	sehir	postakodu	ulke
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany

Şekil 52

Örnek: “musteri” tablosu içinde, “ulke” saha kayıtları arasında ‘Germany’ veya ‘Spain’ kayıtlarının seçilmesi isteniyor. Sorguya dikkat edilirse aynı anda hem ‘Germany’ hem de ‘Spain’ kayıtları istenmiyor. “ulke” sahasında ya ‘Germany’ ya da ‘Spain’ kayıtlarını istiyor. Where kısıtlamasında OR mantıksal ifadesi kullanılmalıdır.

ifade 16 istenilen sorgu

ifade 16

*select * from musteri where ulke = 'Germany' or ulke = 'Spain';*

Şekil 53 sorgu sonucu yapılan seçimin listesidir. Seçim listesinde “ülke” sahasına bakıldığında “Germany” veya “Spain” verisi olan kayıtlar seçilmiştir.

Query Editor		Query History				
1 select * from musteriler where ulke='Germany' or ulke='Spain';						
Data Output		Explain	Messages	Notifications		
musteriid integer	musteriad character varying (50)	iletisimad character varying (50)	adres character varying (50)	sehir character varying (50)	postakodu character varying (50)	ulke character varying (50)
1	1 Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	6 Blauer See Delikatessen	Hanna Moos	Forsterstr. 57	Mannheim	68306	Germany
3	8 Bólido Comidas preparadas	Martin Sommer	C/ Araquil, 67	Madrid	28023	Spain
4	17 Drachenblut Delikatessend	Sven Ottlieb	Waisenweg 21	Aachen	52066	Germany
5	22 FISSA Fabrica Inter. Salchichas S.A.	Diego Roel	C/ Moralzarzal, 86	Madrid	28034	Spain
6	25 Frankenversand	Peter Franken	Berliner Platz 43	München	80805	Germany
7	29 Galería del gastrónomo	Eduardo Saavedra	Rambla de Cataluña, 23	Barcelona	8022	Spain
8	30 Godos Cocina Típica	José Pedro Freyre	C/ Romero, 33	Sevilla	41101	Spain
9	39 Königlich Essen	Philip Cramer	Maubelstr. 90	Brandenburg	14776	Germany
10	44 Lehmanns Marktstand	Renate Messner	Magazinweg 7	Frankfurt a.M.	60528	Germany
11	52 Morgenstern Gesundkost	Alexander Feuer	Heerstr. 22	Leipzig	4179	Germany
12	56 Ottilies Käseladen	Henriette Pfalzheim	Mehrheimerstr. 369	Köln	50739	Germany
13	63 QUICK-Stop	Horst Kloss	Taucherstraße 10	Cunewalde	1307	Germany
14	69 Romero y tomillo	Alejandra Camino	Gran Vía, 1	Madrid	28001	Spain
15	79 Toms Spezialitäten	Karin Josephs	Luisenstr. 48	Münster	44087	Germany
16	86 Die Wandernde Kuh	Rita Müller	Adenauerallee 900	Stuttgart	70563	Germany

Şekil 53

Örnek: “musteriler” tablosu “ülke” sahası kayıtlarında “Spain” verisi olmayan kayıtların gruplandırılarak seçilmesi istenmektedir.

ifade 17 istenilen sorgu cümlecği, Şekil 54 sorgu sonucunun görünümüdür.

ifade 17

select distinct ulke from musteriler where NOT ulke = 'Spain';

Query Editor		Query History																																										
1	<code>select distinct ulke from musteriler where NOT ulke='Spain';</code>																																											
Data Output	Explain	Messages																																										
<table border="1"> <thead> <tr> <th>ulke</th> <th>character varying (50)</th> </tr> </thead> <tbody> <tr><td>1</td><td>Argentina</td></tr> <tr><td>2</td><td>Switzerland</td></tr> <tr><td>3</td><td>Italy</td></tr> <tr><td>4</td><td>Venezuela</td></tr> <tr><td>5</td><td>Belgium</td></tr> <tr><td>6</td><td>Norway</td></tr> <tr><td>7</td><td>Sweden</td></tr> <tr><td>8</td><td>USA</td></tr> <tr><td>9</td><td>France</td></tr> <tr><td>10</td><td>Mexico</td></tr> <tr><td>11</td><td>Brazil</td></tr> <tr><td>12</td><td>Austria</td></tr> <tr><td>13</td><td>Poland</td></tr> <tr><td>14</td><td>UK</td></tr> <tr><td>15</td><td>Ireland</td></tr> <tr><td>16</td><td>Germany</td></tr> <tr><td>17</td><td>Denmark</td></tr> <tr><td>18</td><td>Canada</td></tr> <tr><td>19</td><td>Finland</td></tr> <tr><td>20</td><td>Portugal</td></tr> </tbody> </table>	ulke	character varying (50)	1	Argentina	2	Switzerland	3	Italy	4	Venezuela	5	Belgium	6	Norway	7	Sweden	8	USA	9	France	10	Mexico	11	Brazil	12	Austria	13	Poland	14	UK	15	Ireland	16	Germany	17	Denmark	18	Canada	19	Finland	20	Portugal		
ulke	character varying (50)																																											
1	Argentina																																											
2	Switzerland																																											
3	Italy																																											
4	Venezuela																																											
5	Belgium																																											
6	Norway																																											
7	Sweden																																											
8	USA																																											
9	France																																											
10	Mexico																																											
11	Brazil																																											
12	Austria																																											
13	Poland																																											
14	UK																																											
15	Ireland																																											
16	Germany																																											
17	Denmark																																											
18	Canada																																											
19	Finland																																											
20	Portugal																																											

Şekil 54

Örnek: “müşteri” tablosunda “ülke” sahasında “Germany” olan ve “şehir” sahasında “Berlin” veya “München” olan kayıtların seçilmesi isteniyor.

İstenilen sorgu ifade 18’de yazılmış, Şekil 55’de istenilen sonuçlar gösterilmiştir.

ifade 18

`select * from musteriler where ulke = 'Germany' and (sehir = 'Berlin' or sehir = 'München');`

Query Editor		Query History																												
1	<code>select * from musteriler where ulke='Germany' and (sehir='Berlin' or sehir='München');</code>																													
Data Output	Explain	Messages																												
<table border="1"> <thead> <tr> <th>musteriid</th> <th>musteriad</th> <th>iletisimad</th> <th>adres</th> <th>sehir</th> <th>postakodu</th> <th>ulke</th> </tr> <tr> <th>integer</th> <th>character varying (50)</th> <th>character varying (50)</th> <th>character varying (50)</th> <th>character varying (50)</th> <th>character varying (50)</th> <th>character varying (50)</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>1 Alfreds Futterkiste</td> <td>Maria Anders</td> <td>Obere Str. 57</td> <td>Berlin</td> <td>12209</td> <td>Germany</td> </tr> <tr> <td>2</td> <td>25 Frankenversand</td> <td>Peter Franken</td> <td>Berliner Platz 43</td> <td>München</td> <td>80805</td> <td>Germany</td> </tr> </tbody> </table>	musteriid	musteriad	iletisimad	adres	sehir	postakodu	ulke	integer	character varying (50)	character varying (50)	character varying (50)	character varying (50)	character varying (50)	character varying (50)	1	1 Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany	2	25 Frankenversand	Peter Franken	Berliner Platz 43	München	80805	Germany		
musteriid	musteriad	iletisimad	adres	sehir	postakodu	ulke																								
integer	character varying (50)	character varying (50)	character varying (50)	character varying (50)	character varying (50)	character varying (50)																								
1	1 Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany																								
2	25 Frankenversand	Peter Franken	Berliner Platz 43	München	80805	Germany																								

Şekil 55

Örnek: “müşteri” tablosu içinde “ülke” sahası “Germany” olmayan ve “USA” olmayan kayıtların sayısı isteniyor.

İstenilen sorgu cümlesi ifade 19’da sonuç Şekil 56’de gösterilmiştir.

ifade 19

`select count(*) from musteriler where not (ulke = 'Germany' and ulke = 'USA');`

Query Editor		Query History
1	<code>select count(*) from musteriler where not (ulke='Germany' and ulke='USA');</code>	
Data Output		
	count	bigint
1	91	

Şekil 56

Order By ifadesi

Seçili kayıtlar, Order By ifadesinde yazılacak öznitelik sahası ismine göre, sıralanırlar. Sıralamada artan sıralama ASC (ascending), azalan sıralama DESC (descenging) parametreleri kullanılarak ifade ediliyor. ifade 20 Order By ifadesinin sorgu içinde kullanımının temsildir.

ifade 20

*Select saha_adi
From Tablo_ad
Order By saha_adi ASC/DESC*

Eğer sıralamanın artan veya azalan olduğu belirtilmez is azalan sıralama yapılacaktır.

Örnek: “musteriler” tablosu içinde “ulke” sahasındaki verileri azalan sıralama yaparak ve gruplandırarak seçiniz.

ifade 21

select distinct ulke from musteriler order by ulke;

Query Editor		Query History
1	<code>select distinct ulke from musteriler order by ulke;</code>	
Data Output		
	ulke	character varying (50)
1	Argentina	
2	Austria	
3	Belgium	
4	Brazil	
5	Canada	
6	Denmark	
7	Finland	
8	France	
9	Germany	
10	Ireland	
11	Italy	
12	Mexico	
13	Norway	
14	Poland	
15	Portugal	
16	Spain	
17	Sweden	
18	Switzerland	
19	UK	
20	USA	
21	Venezuela	

Şekil 57

Örnek: Yapılacak seçimde, “musteriler” tablosunda “ulke” ve “musteriler” sahaslarına göre sıralama yapılınsın.

Seçim işlemi yapılırken, sıralama işlemi de yapılacak. Sıralama işleminde iki sahanında aynı anda sıralamasının yapılması isteniyor. Bu işlem aynı anda yapılamayacağı için, sıralama işleminde ilk saha ismine bakılıyor. Sıralama işleminde İlk saha ismi “ulke” bu sahaya göre sıralama yapılacak. Fakat “ulke” sahasında aynı veriden birden fazla varsa ikinci sıralama sahası devreye girecek ve aynı “ulke” verileri içinde “musteriad” sahasına göre sıralama yapılacak. ifade 22 dikkatle incelendiğinde iki sıralama ifadesi arasında And mantıksal sınaması değil, “,” parametresi konulmuştur. Şekil 58 incelenirse, “ulke” sahasındaki veriler alfabetik olarak artan sıralamadır. “ulke” sahasında “Argentina” verisi birden fazladır. “ulke” sahasında “Argentina” verisi olan kayıtlar bu sefer “musteriad” sahasına göre sıralanmıştır.

ifade 22

*select * from musterier order by ulke, musteriad;*

musterierid	musteriad	iletisimad	adres	sehir	postakodu	ulke	
integer	character varying (50)	character varying (50)	character varying (50)	character varying (50)	character varying (50)	character varying (50)	
1	12	Cactus Comidas para llevar	Patricio Simpson	Cerrito 333	Buenos Aires	1010	Argentina
2	54	Océano Atlántico Ltda.	Yvonne Moncada	Ing. Gustavo Moncada 8585 Pl...	Buenos Aires	1010	Argentina
3	64	Rancho grande	Sergio Gutiérrez	Av. del Libertador 900	Buenos Aires	1010	Argentina
4	20	Ernst Handel	Roland Mendel	Kirchgasse 6	Graz	8010	Austria
5	59	Piccolo und mehr	Georg Pipps	Geislweg 14	Salzburg	5020	Austria
6	50	Maison Dewey	Catherine Dewey	Rue Joseph-Bens 532	Bruxelles	B-1180	Belgium
7	76	Suprêmes délices	Pascale Cartrain	Boulevard Tirou, 255	Charleroi	B-6000	Belgium
8	15	Comércio Mineiro	Pedro Afonso	Av. dos Lusíadas, 23	São Paulo	05432-043	Brazil
9	21	Familia Arquibaldo	Aria Cruz	Rua Orós, 92	São Paulo	05442-030	Brazil
10	31	Gourmet Lanchonetes	André Fonseca	Av. Brasil, 442	Campinas	04876-786	Brazil
11	34	Hanari Carnes	Mario Pontes	Rua do Paço, 67	Rio de Janeiro	05454-876	Brazil
12	61	Que Delícia	Bernardo Batista	Rua da Panificadora, 12	Rio de Janeiro	02389-673	Brazil
13	62	Queen Cozinha	Lúcia Carvalho	Alameda dos Canários, 891	São Paulo	05487-020	Brazil
14	67	Ricardo Adocicados	Janete Limeira	Av. Copacabana, 267	Rio de Janeiro	02389-890	Brazil
15	81	Tradição Hipermercados	Anabela Domingues	Av. Inês de Castro, 414	São Paulo	05634-030	Brazil
16	88	Wellington Importadora	Paula Parente	Rua do Mercado, 12	Resende	08737-363	Brazil

Şekil 58

Like ifadesi

Like ifadesi Where ifadesinde belirtilen saha içinde belirtilen kıstasta arama yapmak için kullanılır.

Örnek: “musterier” tablosu “musteriad” sahasındaki veriler içinde “A” ile başlayan kayıtların seçilmesi.

ifade 23, istenilen sorgu cümlesidir. Like kullanımında “A” harfinin aranması için % karakteri kullanılmıştır. Like 'A%' ifadesinde A harfi % ifadesinden geldiği için, verinin başında olan A aranmakta olduğu belirtiliyor.

Eğer verinin sonunda A harfi aransaydı *Like '%A'* kullanılırdı.

Eğer verini içinde A harfi aransaydı *Like '%A%'* kullanılırdı.

ifade 23

*select * from musteriler where musteriad like 'A%';*

Query Editor Query History

```
1 select * from musteriler where musteriad like 'A%';
```

Data Output Explain Messages Notifications

musteriid integer	musteriad character varying (50)	iletisimad character varying (50)	adres character varying (50)	sehir character varying (50)	postakodu character varying (50)	ulke character varying (50)
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana Trujillo Emparedado...	Ana Trujillo	Avda. de la Constitución ...	México D.F.	5021	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.	5023	Mexico
4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK

Şekil 59

Örnek: “musteriler” tablosu “musteriad” sahasına kayıtlı veriler içinde ikinci harfi “r” olan kayıtların seçilmesi isteniyor.

İkinci harfin aranması için “_” alt çizgi karakteri kullanıldı. “r” harfi ikinci karakter olmuş oldu (*ifade 24*). Sorgu sonucu *Şekil 60*’de görülmektedir.

ifade 24

*select * from musteriler where musteriad like '_r%';*

Query Editor Query History

```
1 select * from musteriler where musteriad like '_r%';
```

Data Output Explain Messages Notifications

musteriid integer	musteriad character varying (50)	iletisimad character varying (50)	adres character varying (50)	sehir character varying (50)	postakodu character varying (50)	ulke character varying (50)
4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
17	Drachenblut Delikatessend	Sven Ottlieb	Walserweg 21	Aachen	52066	Germany
20	Ernst Handel	Roland Mendel	Kirchgasse 6	Graz	8010	Austria
25	Frankenversand	Peter Franken	Berliner Platz 43	München	80805	Germany
26	France restauration	Carine Schmitt	54, rue Royale	Nantes	44000	France
27	Franchi S.p.A.	Paolo Accorti	Via Monte Bianco 34	Torino	10100	Italy
32	Great Lakes Food Market	Howard Snyder	2732 Baker Blvd.	Eugene	97403	USA
60	Princesa Isabel Vinhos	Isabel de Castro	Estrada da saúde n. 58	Lisboa	1756	Portugal
81	Tradição Hipermercados	Anabela Domingues	Av. Inês de Castro, 414	São Paulo	05634-030	Brazil
82	Trail's Head Gourmet Pro...	Helvetius Nagy	722 DaVinci Blvd.	Kirkland	98034	USA

Şekil 60

Örnek: “musteriler” tablosu “musteriad” sahasında “a” karakteri ile başlayıp “s” karakteriyle biten verilerin seçilmesi.

ifade 25

*select * from musteriler where musteriad like 'A%s';*

Query Editor		Query History	
1 <code>select * from musteriler where musteriad Like 'A%';</code>			
Data Output			
musteriid	musteriad	iletisimad	adres
integer	character varying (50)	character varying (50)	character varying (50)
1	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitución ...
sehir	postakodu	ulke	
character varying (50)	character varying (50)	character varying (50)	
México D.F.	5021	Mexico	

Şekil 61

Between İfadesi

Between ifadesi, iki değer arasında sınıma yapmak için kullanılır. *Like* ifadesi gibi *where* ifadesi kısmında kullanılmakta.

Select öznelik_ad
from tabaka
where öznelik_ad *Between* deger1 *and* deger2

Örnek: “urunler” tablosunda “fiyat” sahasındaki değerler içinde 10 ile 20 arasındaki değerlerin seçilmesi.

ifade 26
*select * from urunler where fiyat between 10 and 20;*

Query Editor		Query History	
1 <code>select * from urunler where fiyat between 10 and 20;</code>			
Data Output			
urunid	urunadi	tedarikciid	kategoriid
integer	character varying (50)	integer	integer
1	Chais	1	1
2	Chang	1	1
3	Aniseed Syrup	1	2
4	Genen Shouyu	6	2
5	Pavlova	7	3
6	Sir Rodney's Scones	8	3
7	NuNuCa Nuß-Nougat-Creme	11	3
8	Gorgonzola Telino	14	4
9	Sasquatch Ale	16	1
10	Steeleye Stout	16	1
11	Inlagd Sill	17	8
12	Chartreuse verte	18	1
13	Boston Crab Meat	19	8
14	Singaporean Hokkien Fried Mee	20	5
15	Gula Malacca	20	2
16	Spegesild	21	8
17	Chocolade	22	3
18	Maxilaku	23	3
19	Valkoinen suklaa	23	3
20	Ravioli Angelo	26	5
21	Escarots de Bourgogne	27	8
birim	fiyat		
character varying (50)	double precision		
10 boxes x 20 bags	18		
24 - 12 oz bottles	19		
12 - 550 ml bottles	10		
24 - 250 ml bottles	15.5		
32 - 500 g boxes	17.45		
24 pkgs. x 4 pieces	10		
20 - 450 g glasses	14		
12 - 100 g pkgs	12.5		
24 - 12 oz bottles	14		
24 - 12 oz bottles	18		
24 - 250 g jars	19		
750 cc per bottle	18		
24 - 4 oz tins	18.4		
32 - 1 kg pkgs.	14		
20 - 2 kg bags	19.45		
4 - 450 g glasses	12		
10 pkgs.	12.75		
24 - 50 g pkgs.	20		
12 - 100 g bars	16.25		
24 - 250 g pkgs.	19.5		
24 pieces	13.25		

Şekil 62

Örnek: “urunler” tablosu fiyat sahasında 10 ile 20 arasında olan ve “kategoriid” sahasında ise 1,2 veya 3 değeri çermeyen kayıtların seçilmesi.

İşlemin yapılması için hem “fiyat” sahasında *Between* ifadesi kullanılacak hem de “kategoriid” sahasında is *IN* ifadesi kullanılacak. *IN* ifadesi, kendisinden sonra parantez içinde yazılan değerler arasında OR mantıksal operatörünün kullanılmasını sağlıyor. NOT IN ifadesi bir nevi çermeyenler olarak kullanılmasını sağlıyor.

ifade 27

*select * from urunler where fiyat between 10 and 20 and kategoriid not in (1,2,3);*

Query Editor		Query History									
1 <i>select * from urunler where fiyat between 10 and 20 and kategoriid not in (1,2,3);</i>											
Data Output		Explain	Messages	Notifications							
urunid	integer	urunadi	character varying (50)	tedarikciid	integer	kategoriid	integer	birim	character varying (50)	fiyat	double precision
1	31	Gorgonzola Telino		14	4	12 - 100 g pkgs	12.5				
2	36	Inlagd Sill		17	8	24 - 250 g jars	19				
3	40	Boston Crab Meat		19	8	24 - 4 oz tins	18.4				
4	42	Singaporean Hokkien Fried Mee		20	5	32 - 1 kg pkgs.	14				
5	46	Spegesild		21	8	4 - 450 g glasses	12				
6	57	Ravioli Angelo		26	5	24 - 250 g pkgs.	19.5				
7	58	Escargots de Bourgogne		27	8	24 pieces	13.25				
8	73	Röd Kaviar		17	8	24 - 150 g jars	15				
9	74	Longlife Tofu		4	7	5 kg pkg.	10				

Şekil 63

Cast(as) (Veri Dönüşümü)

Cast fonksiyonu postgresql veri tabanı yönetim sisteminde sahalarda içindeki verilerin veri dönüşümü işlemi için kullanılır. Kullanımda:

Cast(sahanın_adi as dönüşeceği_veritipi)

Örneğin imar adasına bir kimlik numarası verilmesi isteniyor. Kimlik numarası verilirken de içinde olduğu mahallenin kimlik verisi de kullanılmak isteniyor. Örneğin adanın numarası 102, 102 numaralı adanın içinde bulunduğu mahalle için verilen kimlik numarası 101. Her ikisi de sayısal veri tipinde sahalarda tutulmaktadır. Bu numaraların sql parametreleri kullanılarak 101102 haline dönüştürülmesi ve sonuç değerın sayısal hale dönüştürülmesi istenmektedir.

select cast(cast(ada_no as text) || cast(mahalle_id as text) as numeric(24,0)) from ada

cast(ada_no as text) → ada_no sahasındaki veriyi text veriye dönüştürüyor

cast(mahalle_id as text) → mahalle_id sahasındaki veriyi text verisine dönüştürüyor

|| → ile metin verisine dönüşen iki veriyi birleştiriyor

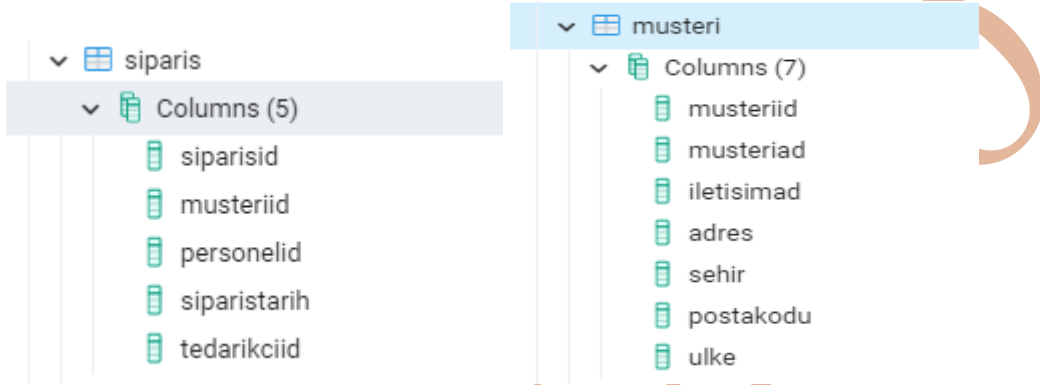
cast(... as numeric(24,0)) → as ifadesinden öncesini sayısal veriye dönüştürüyor

Join (Bağlantı Kurma)

Join (Bağlantı) işlemi, farklı tablolarda yer alan (aynı veriye sahip) sahalarda ile bağlantının sağlanması ve bağlantılı tablodan seçilen veriden bir diğer tablodaki veriye erişilmesini sağlar. Join işlemini yapabilmek için iki yol vardır.

Örnek: Yapılacak listelemede “siparis” tablosundaki “siparisid”, “siparistarih” sahalarındaki verileri ile sipariş tablosundaki verilerle bağlantılı “musteri” tablosundaki “musteriad” sahasının da seçili verilerle gösterilmesi istenmektedir.

Şekil 64 “siparis” ve “musteri” tablolarının sahaslarını göstermekte. Dikkat edilirse her iki tabloda da “musteriid” sahası ortaktır. Bu ortak saha sayesinde bağlantı kurulabilir. Bu bağlantı sayesinde "siparis" tablosunda seçilecek bir kayıttan “musteri” tablosundaki bağlantılı kayda ulaşılabilir ve bağlantılı kayıttaki diğer veriler elde edilebilir.



Şekil 64

Sipariş bilgilerinin olduğu “siparis” tablosundaki “musteriid” sahası ile müşteri bilgilerinin olduğu “musteri” tablosuna ulaşılacak ve bağlantılı kayıttan da “musteriad” bilgisindeki veri tablodan elde edilecek.

Inner Join ifadesiyle bağlantı kurulması:

Biraz önceki “siparis” tablosuyla “müşteri” tablosu arasındaki bağlantı kurulması işlemi ifade 28 istenilen sorgu cümlesidir. *From* ifadesinde bağlantı kurulacak olan tablonun bir tanesi (siparis), *inner join* ifadesinde bağlantı kurulacak diğer tablo (musteri) belirtilmiştir. *On* ifadesinden sonra bağlantı kurulacak kayıtlar eşleşmiştir.

ifade 28

```
select siparis.siparisid, musteriiid, siparis.siparistarih
from siparis
inner join musteriiid on siparis.musteriid = musteriiid;
```

Where ifadesiyle bağlantı kurulması:

Inner join örneğindeki bağlantı kurulması işlemi *Where* komutu kullanılarak da yapılabilir. ifade 29, ifade 28’de yapılan bağlantı işleminin *Where* komutu kullanılarak yapılan sorgu cümlesidir. ifade 28’den farklı olarak, *From* ifadesinde bağlantı kurulacak olan tablolar belirtilmiştir. *Where* ifadesinde de bağlantı kurulacak olan sahaslar eşleşmiştir.

ifade 29

```
select siparis.siparisid,musteri.musteriad,siparis.siparistarih
from siparis,musteri
where siparis.musteriid = musterii.musteriid;
```

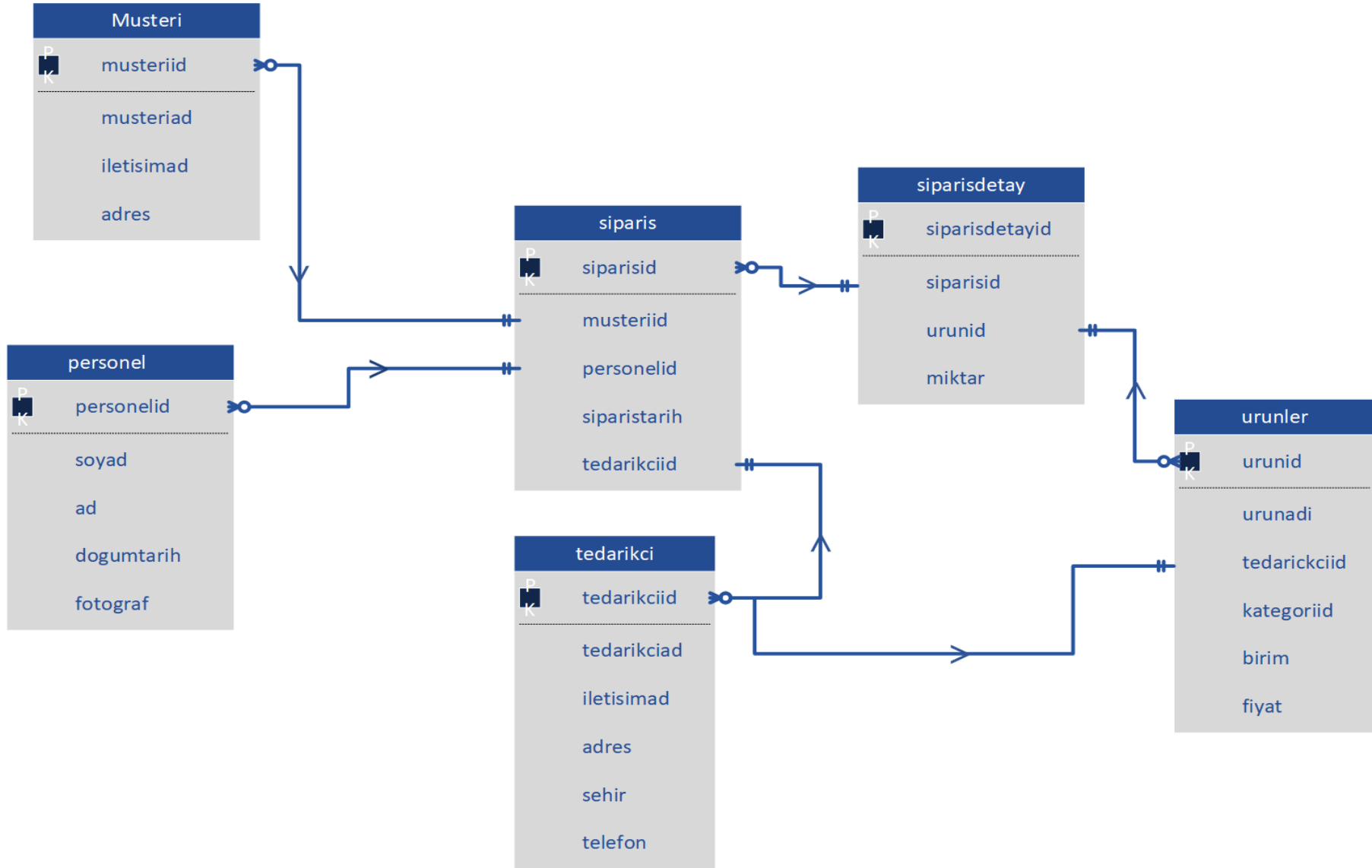
Şekil 65 her iki sorgu cümlesinin de listelenen seçilen kayıtları gösterir.

	Data Output	Explain	Messages	Notifications
	siparisid integer	musteriad character varying (50)		siparistarih date
1	10248	Wilman Kala		1996-07-04
2	10249	Tradição Hipermercados		1996-07-05
3	10250	Hanari Carnes		1996-07-08
4	10251	Victuailles en stock		1996-07-08
5	10252	Suprêmes délices		1996-07-09
6	10253	Hanari Carnes		1996-07-10
7	10254	Chop-suey Chinese		1996-07-11
8	10255	Richter Supermarkt		1996-07-12
9	10256	Wellington Importadora		1996-07-15
10	10257	HILARIÓN-Abastos		1996-07-16
11	10258	Ernst Handel		1996-07-17
12	10259	Centro comercial Moctezuma		1996-07-18

Şekil 65

Şekil 66 örneklerde kullanılan tablolar ve tablolar arasında oluşan bağlantılar gözükmemektedir. Eğer bir siparişe ait “siparisid” bilgisine ait veri bilinirse, siparişi veren müşterinin “musteri” tablosundaki bilgilerine ait verilere veya “tedarikci” tablosundaki tedarikçi bilgilere ait verilere veya “urunler” tablosundaki bilgilere ait verilere erişilebilir.

Coğrafi Bilgi Sistemleri II



Şekil 66

Örnek: “siparis” tablosunda siparisid değeri 10282 olan siparişi yapan müşterinin adı ve siparişi hazırlayan personel adı bilgilerine ait verilerin listesi istenmektedir.

Şekil 67 istenilen sorgu için bağlantının *inner join* komutuyla tablolar arasında bağlantını yapıldığı *where* komutuyla “siparisid” değerinin kısıtlama yapıldığı , Şekil 68 istenilen sorgu için bağlantının ve kısıtlamanın sadece *where* komutu ile yapıldığı sorgudur. Şekil 69 iki sorgunun da verdiği aynı sonuçtur.

```
Query Query History
1 select musterimusteriad,personel.ad
2 from siparis
3 inner join musterimusteri on siparis.musteriid=musterimusteriid
4 inner join personel on siparis.personelid=personel.personelid
5 where siparis.siparisid=10282;
```

Şekil 67

```
Query Query History
1 select musterimusteriad,personel.ad
2 from musterimusteri,personel,siparis
3 where musterimusteriid=siparis.musteriid
4 and personel.personelid=siparis.personelid
5 and siparis.siparisid=10282;
```

Şekil 68

Data Output		Messages	Notifications
	musteriad character varying (50)	ad character varying (50)	
1	Romero y tomillo	Margaret	

Şekil 69

Örnek: id değeri 7 numaralı müşterinin verdiği siparişlerde, müşterinin adı, satın aldığı ürünlerin adı, satın aldığı tarih, aldığı ürünün miktar bilgilerinin listelenmesi ve sonuç listenin sipariş tarihlerine göre (en yakın tarihten en uzak tarihe göre) sıralanması istenmektedir.

Şekil 70 sadece *where* komutu kullanılarak bağlantı ve kısıtlamanın yapıldığı kod bloğudur. Şekil 71 tablolar arasında bağlantının *inner join* komutuyla kısıtlamanın ise *where* komutuyla yapıldığı sorgu bloğudur. Şekil 72 her iki sorgunun da aynı olduğu sonuç listedir. Dikkat edilirse tarih bilgisine göre yapılan sıralamada *DESC* kodu kullanılmıştır.

Query Query History

```

1 select musterimusteriad, siparis.siparistarih,urunler.urunadi,siparisdetay.miktar
2 from siparis, musterimusteri,siparisdetay,urunler
3 where siparis.musteriid=musterimusteri.musteriid and
4 siparis.siparisid=siparisdetay.siparisid and
5 siparisdetay.urunid=urunler.urunid and
6 musterimusteri.musteriid=7
7 order by siparis.siparistarih DESC;
```

Şekil 70

Query Query History

```

1 select musterimusteriad, siparis.siparistarih,urunler.urunadi,siparisdetay.miktar
2 from siparis
3 inner join musterimusteri on siparis.musteriid=musterimusteri.musteriid
4 inner join siparisdetay on siparis.siparisid=siparisdetay.siparisid
5 inner join urunler on siparisdetay.urunid=urunler.urunid
6 where musterimusteri.musteriid=7
7 order by siparis.siparistarih DESC;
```

Şekil 71

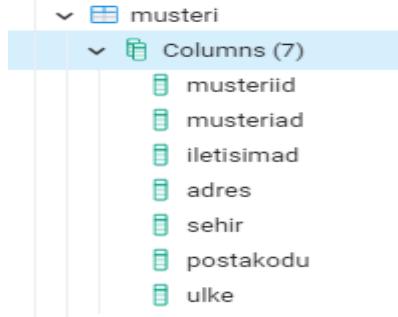
Data Output Messages Notifications

	musteriad character varying (50)	siparistarih date	urunadi character varying (50)	miktar integer
1	Blondel père et fils	1997-02-05	Spegesild	5
2	Blondel père et fils	1997-02-05	Wimmers gute Semmelknödel	30
3	Blondel père et fils	1997-02-05	Rhönbräu Klosterbier	24
4	Blondel père et fils	1997-02-05	Gnocchi di nonna Alice	40
5	Blondel père et fils	1996-11-22	Tourtière	28
6	Blondel père et fils	1996-11-22	Maxilaku	35
7	Blondel père et fils	1996-11-22	Rössle Sauerkraut	30
8	Blondel père et fils	1996-11-22	Thüringer Rostbratwurst	35
9	Blondel père et fils	1996-11-22	Côte de Blaye	10
10	Blondel père et fils	1996-09-04	Mozzarella di Giovanni	20
11	Blondel père et fils	1996-09-04	Chartreuse verte	60
12	Blondel père et fils	1996-07-25	Alice Mutton	30
13	Blondel père et fils	1996-07-25	Outback Lager	20

Şekil 72

Insert Into ifadesi

Insert into ifadesi, tabloya yeni bir kayıt eklemek için kullanılıyor. “musteri” tablosuna bir kayıt eklenmesi isteniyor. Şekil 73’de “müşteri” tablosunun sahaları gözükmemektedir. Kayıt yapılırken, var olan sahalara kayıt eklenmelidir.



Şekil 73

Insert into iki türlü kayıt yapılabilir. Birincisinde *insert into* ifadesinden sonra tablonun ifadesinin yazılması yeterli olacaktır (ifade 30).

ifade 30

```
insert into musterii
values(92,'emre','emre_kaman','kaman myo','kaman','40300','Turkey');
```

İkinci kullanımda tablo adı yanında parantez için sahalara isimleri yazılabilir (ifade 31.)

ifade 31

```
insert into
musterii(musteriid,musteriad,iletisimad,adres,sehir,postakodu,ulke)
values(93,'ahmet','ahmet_kaman','uygulamalı bilimler','kaman','40300','Turkey');
```

Şekil 74 *insert into* ifadesiyle yapılan kayıtların seçimini göstermektedir.

Query Editor		Query History	
1 select * from musterii where musteriid in (92,93);			
Data Output		Explain	Messages
musteriid	musteriad	iletisimad	adres
integer	character varying (50)	character varying (50)	character varying (50)
1	92 emre	emre_kaman	kaman myo
2	93 ahmet	ahmet_kaman	uygulamalı bilimler

Şekil 74

Update ifadesi

Update ifadesi, tablo içinde var olan bir kaydın güncellenmesi (yenilenmesi) için kullanılır. Kullanımında *update* ifadesinde güncelleme yapılacak tablonun adı, *set* ifadesinde güncel verinin girileceği saha ve güncel verisi, *where* ifadesinde kayda ulaşılabilmesi için kısıtlamanın yapılması (ifade 32).

ifade 32

update tabaka_adi

set öznitelik_ad = güncel_değer

where kısıtlama;

update musteri set iletisimad = 'emre_ince' where musteriid = 92;

Query Editor Query History

```
1 select * from musteri where musteriid=92;
```

Data Output Explain Messages Notifications

	musteriid integer	musteriad character varying (50)	iletisimad character varying (50)	adres character varying (50)	sehir character varying (50)	postakodu character varying (50)	ulke character varying (50)
1	92	emre	emre_ince	kaman myo	kaman	40300	Turkey

Şekil 75

Delete ifadesi

Delete ifadesi, tabloda var olan bir kaydın silinmesi için kullanılır. *ifade 33 delete* ifadesi için kullanılacak sql cümlesi kullanımı vardır.

ifade 33

Delete

from tablo_ad

where kısıtlama;

Query Editor Query History

```
1 delete from musteri where musteriid=93;
```

Data Output Explain Messages Notifications

```
DELETE 1
Query returned successfully in 67 msec.
```

Şekil 76

Şekil 76 delete ifadesiyle 93 “musteriid” değerine sahip kayıt silinmiştir. Şekil 77 93 “musteriid” değerine sahip kaydın olmadığını göstermektedir.

Query Editor		Query History	
1	<code>select * from musterid where musterid=93;</code>		
Data Output			
Column Name	Column Type	Column Name	Column Type
musterid	integer	musteriad	character varying (50)
		iletisimad	character varying (50)
		adres	character varying (50)
		sehir	character varying (50)
		postakodu	character varying (50)
		ulke	character varying (50)

Şekil 77

Select into ifadesi

Select into ifadesi, bir tabakadaki kayıtların yeni tabakaya aktarılmasında veya bir tabakadaki sahalarn boş bir tabakaya aktarılmasında kullanılır. ifade 34 select into ifadesine kullanımına bir örnektir.

ifade 34

Select tablodan_seçilecek_kayıtlar
into yeni_tabaka_adi
from tabaka_adi
where kısıtlama;

Örnek: “musteri” tablosu “musteriid” ve “musteriad” sahalarnındaki verilerin alınıp “müşteri_id_ad adlı bir tabloya aktarılması.

İstenilen sorgu cümlecigi ifade 35’de gösterilmektedir. Dikkat edilirse, “müşteri_id_ad” tablosu ve sahalari önceden tanımlanmamıştır. Şekil 79 sol resimde

ifade 35

select musteriid, musteriad **into** musterid_ad **from** musterid;


```
Query Query History
1 select musteriid, musteriad into musterii_id_ad from musterii;
```

Şekil 78

The screenshot shows a database interface with a query window and a data output window. The query window contains the following SQL statement:

```
1 select * from musterii_id_ad;
```

The data output window displays the following table:

	musteriid integer	musteriad character varying (50)
1	1	Alfreds Futterkiste
2	2	Ana Trujillo Emparedados y helados
3	3	Antonio Moreno Taquería
4	4	Around the Horn
5	5	Berglunds snabbköp
6	6	Blauer See Delikatessen
7	7	Blondel père et fils
8	8	Bólido Comidas preparadas
9	9	Bon app'
10	10	Bottom-Dollar Marketse
11	11	Bis Beverages

On the left side, there is a tree view showing the database structure with tables: kargofirma, kategoriler, musterii, musterii_id_ad, personel, siparis, siparisdetay, tedarikci, and urunler. The 'musterii_id_ad' table is selected.

Şekil 79

Örnek: “musterii” tablosu “ülke” sahası verileri arasında “Germany” olan kayıtların ayrı bir tabloya aktarılması.

Kısıtlama getirilerek verinin aktarılması için where ifadesi kullanılması gerekir. Sorgu cümlecği ifade 36’de yazılmıştır, oluşan yeni tablo içeriği Şekil 80’de görülmektedir.

ifade 36

*select * into musterii_almanya from musterii where ülke = 'Germany';*

The screenshot shows a database interface with a query window and a data output window. The query window contains the following SQL statement:

```
1 select * from musterii_almanya;
```

The data output window displays the following table:

	musteriid integer	musteriad character varying (50)	iletisimad character varying (50)	adres character varying (100)	sehir character varying (50)	postakodu character varying (50)	ülke character varying (50)
1	1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	6	Blauer See Delikatessen	Hanna Moos	Forsterstr. 57	Mannheim	68306	Germany
3	17	Drachenblut Delikatessend	Sven Ottlieb	Walserweg 21	Aachen	52066	Germany
4	25	Frankenversand	Peter Franken	Berliner Platz 43	München	80805	Germany
5	39	Königlich Essen	Philip Cramer	Maubelstr. 90	Brandenburg	14776	Germany
6	44	Lehmanns Marktstand	Renate Messner	Magazinweg 7	Frankfurt a.M.	60528	Germany
7	52	Morgenstern Gesundkost	Alexander Feuer	Heerstr. 22	Leipzig	4179	Germany
8	56	Ottlies Käseladen	Henriette Pfalzheim	Mehrheimerstr. 369	Köln	50739	Germany
9	63	QUICK-Stop	Horst Kloss	Taucherstraße 10	Cunewalde	1307	Germany
10	79	Toms Spezialitäten	Karin Josephs	Luisenstr. 48	Münster	44087	Germany
11	86	Die Wandernde Kuh	Rita Müller	Adenauerallee 900	Stuttgart	70563	Germany

Şekil 80

Create Table İfadesi

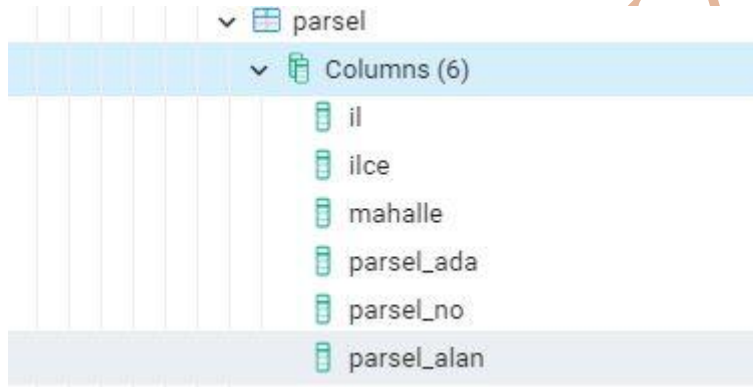
Create Table ifadesi, aktif veri tabanı içinde tablo oluşturmak için kullanılır.

Örnek: Kadastro parsellerinin bilgilerinin tutulduğu bir tablo oluşturulması isteniyor.

İstenilen sorgu cümlesi, ifade 37’de yazılmıştır. ifade 37 incelendiğinde, Create table ifadesinden sonra tablonun adı yazılıyor, açılan parantez içinde tablo içindeki sahalanın isimleri ve sahada tutacakları verinin veri tipi bilgisi belirtiliyor.

ifade 37

```
Create table parsel(il varchar(50),
ilce varchar(50), mahalle varchar(50),
parsel_ada int, parsel_no int,
parsel_alan float);
```



Şekil 81

PostgreSQL Veri Tipleri:

PostgreSQL veri tabanı yönetim sisteminde tablo altında tutulacak sahalaları oluştururken saha içinde tutulacak verilerin tiplerini de belirtmeliyiz. Aşağıda kullanılacak genel veri tiplerinin listesi verilmiştir (Tablo 8).

Tablo 8 (Data types in Postgres, 2023)

Veri Tipi	Tanım
bigint	-9 223 372 036 854 775 808 ile 9 223 372 036 854 775 807
integer	-2 147 483 648 ile 2 147 483 647
smallint	-32 768 ile 32 767
tinyint	0 ile 255
bit	0 veya 1.
boolean	0 veya 1 veya unknown (null)
decimal(precision, scale)	Noktadan önce 131072 hane noktadan sonra 16383 hane

numeric(precision, scale)	Decimal veri tipi ile aynı
real	6 haneli ondalıklı (reel) sayı
double precision	Ondalık hane sayısı dahil 15 haneli reel sayı
money	Ekonomik veri tipi -92233720368547758.08 ile +92233720368547758.07
timestamp (precision)	Tarih ve saat değerleri için kullanılır. Zaman dilimi yokur. Zaman hassasiyeti 0 saniye ile 6 mili saniye arası.
timestampz (precision)	Tarih ve saat değerleri için kullanılır. Zaman dilimi vardır. Zaman hassasiyeti 0 saniye ile 6 mili saniye arası.
date	Takvimdeki tarih formatı (yıl, ay, gün)
char(n)	sabit uzunluklu karakter veri tipi. Maksimum 8000 karakter alır.
varchar(n)	Metin ifade için kullanılır. Maksimum 8000 karakter alır.
text	Metin ifadeler için kullanılır. Karakter limiti yok diye belirtilir ama maksimum karakter limiti 2 147 483 647 .

Geometry veri tipi içinde tutulacak grafik obje tipleri:

Veri Tipi	Tanım
point	Nokta geometrik objesini tutmak için kullanılan veri tipidir.
linestring	Çizgi geometrik objesini tutmak için kullanılan veri tipidir.
polygon	Çokgen geometrik objesini tutmak için kullanılan veri tipidir.

Tanımlanan Değişkenlerin Ek Özellikleri

- **Not Null:** sahanın boş bırakılmayacağını belirtir,
- **Unique:** sahadaki değer, saha içinde tekil değere sahip olacağını belirtir,
- **Primary Key:** sahadaki değer, tablo içinde kaydın tekil hale gelmesini sağlar,
- **Serial:** otomatik olarak artan tamsayı veri tipi.

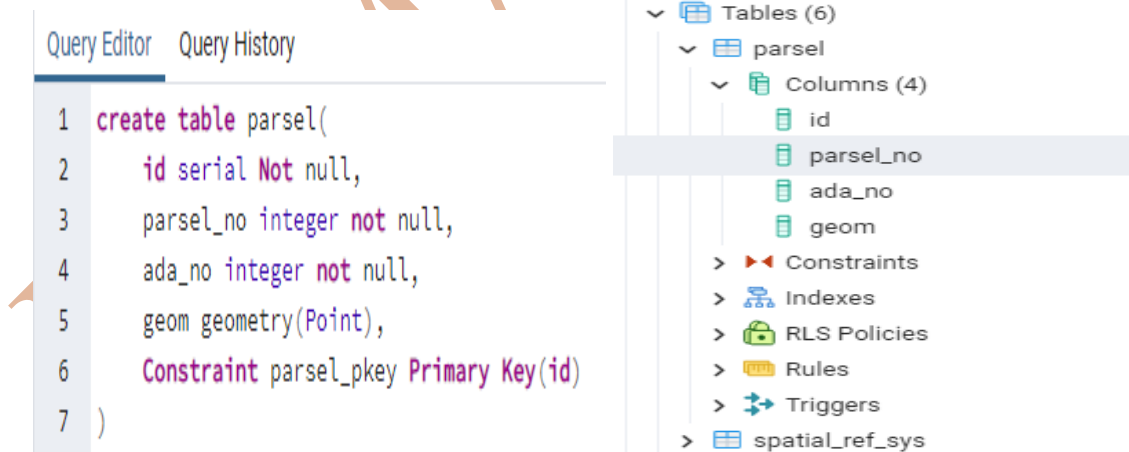
Örnek: id adlı sahası *anahtar saha* olacak ve id değeri *otomatik olarak artacak*, parsel numarasını parsel_no sahasında tam sayı değerle tutacak, ada numarasını ada_no sahasında tamsayı değerle tutacak, parsel_no ve ada_no sahalari *boş saha olmayacak*, *nokta grafik* objelerini tutacak tablo oluşturulması isteniyor.

ifade 38

CREATE TABLE parsel(

id serial Not Null,
parsel_no integer NOT NULL,
ada_no integer NOT NULL,
geom geometry(Point),
CONSTRAINT parsel_pkey PRIMARY KEY (id)
);

ifade 38 istenilen SQL kod bloğudur. Tablo adı *Create Table* ifadelerinden sonra “parsel” olarak belirtilmiştir. “id” sahası otomatik artması için *serial* veri tipinde oluşturulmuş ve *Not Null* denilerek boş bırakılmayacağı belirtilmiş. id sahasının diğer bir özelliği tablo içinde kayıtların tekil hale gelmesi istenmiştir. Bu özellik kod bloğunun en son satırlarında belirtilmiştir. *Constraint* ifadesi sınırlandırma getirildiğini, *parsel_pkey* ifadesi parsel tablosunun tekil sahasının belirlendiğini, *PRIMARY KEY (id)* ifadesiyle de “id” adlı sahanın tablonun anahtar sahası (tablo içindeki tüm kayıtlar arasında, yapılan yeni kayıdı tekil hale getiren saha) olduğu belirtilmiştir. *parsel_no* ve *ada_no* sahaları tam sayı değer alacak şekilde integer veri tipinde ve boş bırakılmayacak şekilde Not Null olarak belirtilmiş. *geom* adlı saha nokta grafik objelerini tutacak şekilde belirtilmiş bu işlemi yapmak için *geometry* ifadesi kullanılmış. Dikkat edilirse *geometry* ifadesinde sadece tabloda tutulacak olan grafik obje türü belirtilmiştir. Bu sayede tabloda sadece tek tipte grafik obje türü kayıt altına alınabilecektir (*Şekil 82*).



Şekil 82



Eğer Geometry tipindeki sahada geometri tipi belirtilmez ise tabloya tüm geometri tiplerinde grafik obje eklenebilir. Örneğin ArcMAP yazılımında her bir tablo için tek bir geometrik obje tipi tanımı zorunludur. Her bir tabakada tek bir grafik obje tipinde obje eklenebilir.



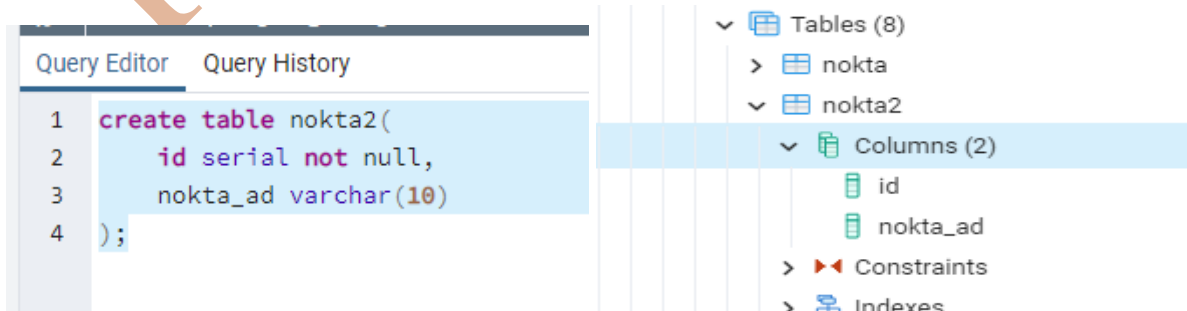
Verilen örnekte geometry tipindeki sahada koordinat sistemi tanımı yapılmamıştır.

Örnek: “nokta” adlı bir tablo oluşturulmak isteniyor. Tablo içinde “id” isimli saha otomatik olarak artacak ve boş kalmayacak şekilde oluşturulmalı. “id” isimli saha tablonun anahtar sahası olmalı. Tabloda noktanın adının kaydedileceği “nokta_ad” isimli 10 karakterle sınırlı bir saha oluşturulmalı. Tabloda nokta hakkında ek bilginin tutulduğu (poligon veya detay noktası olduğu gibi ek bilgi) “kod” isimli 10 karakter alan bir saha oluşturulmalı. Tabakada nokta grafik verilerini tutulacağı “geom” isimli bir saha oluşturulmalı. Nokta grafik objeleri jeodezik enlem ve jeodezik boylam verilerini tutacağı 4326 EPSG kodlu koordinat sisteminde kayıt altına alınmalı.

ifade 39 istenilen SQL kod bloğudur. Kod bloğu dikkatli incelenirse, “geom” sahasının veri tipi tanımı yapılırken hem nokta objesinin tutulması için *point* grafik veri tipi belirtilmiştir. Hem de WGS 84 elipsoidine göre jeodezik enlem ve jeodezik boylam koordinatlarının tutulacağı koordinat sistemi belirtilmiştir.

ifade 39
**Create table nokta(
 id serial not null,
 nokta_ad varchar(10),
 kod varchar(10),
 geom geometry(point, 4326),
 Constraint nokta_pkey Primary key (id)
);**

Örnek: nokta2 adlı tablo oluşturulmuş. Tabloda nokta adlarının tutulacağı “nokta_ad” isminde 10 karakterle sınırlı saha bulunmakta. “geom” isimli geometry tipinde geometrik objelerin tutulacağı sahanın eklenmesi unutulmuş. “geom” sahası nokta grafik objesinin tutulacağı ve nokta koordinatları 4326 EPSG kodlu koordinat sisteminde olduğu bir sahadır. Eklenmesi unutulmuş “geom” sahası nasıl eklenir?



Şekil 83

Şekil 83 sol resimde “nokta2” adlı tablonun eklenmesine dair SQL kod bloğu bulunmakta, sağ resimde ise tablo eklendikten sonra tablolar arasındaki durumu gözükmektedir.

ifade 40 var olan bir tabloya geometri verisi eklenebilecek bir sahanın eklenebilmesi için gereken AddGeometryColumn() fonksiyonunun kullanımına örnektir. Fonksiyonun ilk parametresi “public”, nokta2 tablosunun içinde bulunduğu şemanın adıdır. ikinci parametre sahanın ekleneceği “nokta2” tablosunun adı, üçüncü parametre tabloya eklenecek ve geometrik obje verisini kaydedecek “geom” sahasının adı, dördüncü parametre kaydedilecek olan geometrik objenin koordinat sistemidir. Örnekte WGS 84 datumuna göre jeodezik enlem ve jeodezik boylam eklenmesi için 4326 EPSG değeri girilmiştir. Beşinci parametre geometrik objenin obje tipidir. Örnekte tablo içinde sadece nokta grafik objeleri girilebilmesi için “POINT” ibaresi yazılmıştır. Altıncı ve son parametre 2 ifadesi, iki boyutlu koordinat sisteminin oluşturulacağını belirtiyor.



Şekil 84

Tabloya Grafik Obje Ekleme

PostgreSQL veri tabanı yönetim sisteminde eğer PostGIS eklentisi eklendiyse oluşturulan veri tabanı içinde tanımlanan tablolarda grafik obje kaydı yapılabilir. Bu işlemin yapılması için tabloda *Geometry* veri tipinde değer tutan bir saha eklenir. Bu şekilde bir tabloya

kayıt yapabilmek için birden fazla yöntem (fonksiyon) bulunmaktadır. Konu anlatımda bu yöntemlerden bahsedilecek.

ST_GeomFromText() fonksiyonu:

ST_GeomFromText() fonksiyonu içine metin ifade olarak geriye geometri döndürür. Fonksiyon içerisine yazılacak metin ifade ' ' içine yazılır. İfade yazılırken Open Geospatial Consortium (OGS) Well-known Text (WKT – iyi bilinen metin) standart formatında yazılmaktadır. WKT formatı, daha önceki EPSG kodları hakkında konu anlatımında bahsedilen www.epsg.io sitesinde görülmüştü. WKT formatındaki bir metin, Coordinate Reference System (CRS – Koordinat Referans Sistemi) tanımını ifade etmek için kullanılır (Open Geospatial Consortium, 2023).

St_GeomFromText() fonksiyonu, geriye geometri değeri döndürür. Ama esas kullanım amacı *Geometry* veri tipinde sahası olan bir tabloya geometrik obje eklenmesidir. İşlem yapılırken tabloya kayıt işlemiyle beraber kullanılmalıdır. Sonuçta temel amaç geometrik bir objenin tabloya kayıt işlemi olduğu için tabloya veri eklemek için kullanılan *Insert Into* ifadesiyle kullanılmalıdır.

ST_GeomFromText() Fonksiyonu ile Nokta (Point) Grafik Objesinin Eklenmesi:

Örnek: nokta2 adlı tabloda “nokta_ad” adlı 10 karakter tutabilen saha ve “geom” adlı geometry veri tipinde, nokta grafik objesini kayıt altına alan ve koordinat sistemi 4326 EPSG kodunda olan saha mevcuttur. Bu tabloya bir kayıt eklenmek istenmektedir.

ifade 41 istenilen kod bloğudur. Insert Into ifadesi sonrası tablonun adı “nokta2” ve saha adları vardır. *Values* kodu içinde sahaların tablodaki sırasına göre noktaya ait nokta adı ve noktaya ait geometrik kayıt yapılmıştır. Geometrik kayıt yapılırken ST_GeomFromText() fonksiyonu kullanılmıştır.

ifade 41

```
insert into nokta2(nokta_ad, geom)
values('KamanMyo',st_geomfromtext('POINT(33.697203725661474 39.35609534526361)',
4326));
```

ST_GeomFromText() fonksiyonu kullanılırken: (Eğer Nokta grafik objesi ekleniyorsa)

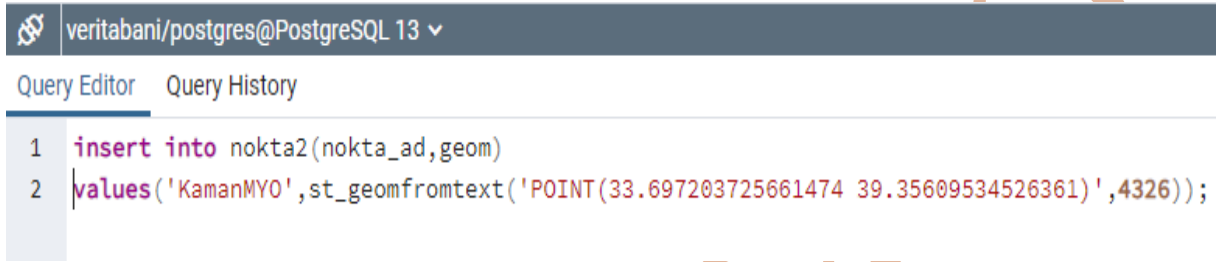
ST_GeomFromText('POINT(*boylam enlem*)',*EPSG_KODU*)

formatı dikkate alınır. ST_GeomFromText() fonksiyonunun ilk parametresi, WKT formatında grafik obje tipi tanımı, ' ' içinde yazılmıştır.

Deneme amaçlı Google Map uygulaması üzerinden alınan (*Tablo 9*), Kaman Meslek Yüksekokuluna ait ve Kaman Uygulamalı Bilimler Yüksekokuluna ait, jeodezik enlem ve jeodezik boylam verileri kullanılarak noktalar eklenmiştir.

Tablo 9

Okul Adı	Boylam	Enlem
Kaman Myo	33.697203725661474	39.35609534526361
Kaman Uygulamalı Bilimler	33.70107683533369	39.35728162709027



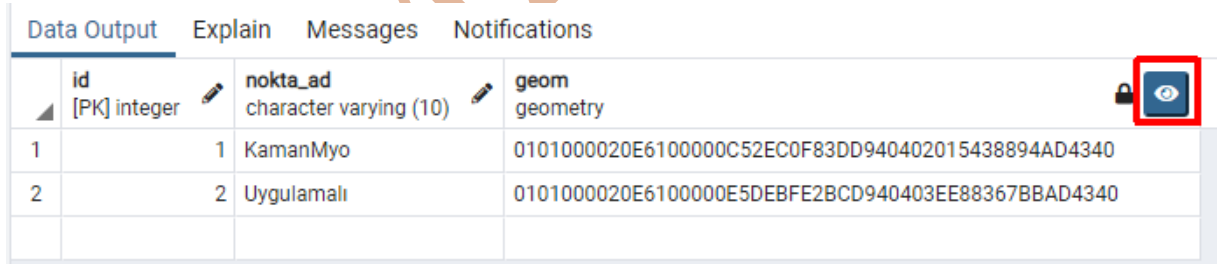
```

veritabani/postgres@PostgreSQL 13
Query Editor Query History
1 insert into nokta2(nokta_ad,geom)
2 values('KamanMYO',st_geomfromtext('POINT(33.697203725661474 39.35609534526361)',4326));

```

Şekil 85

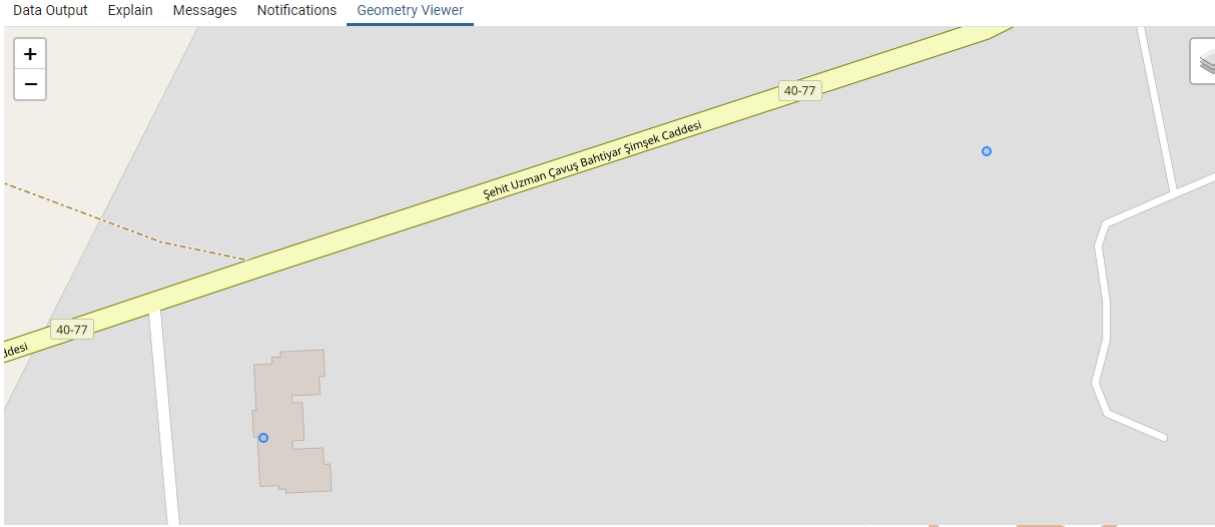
Eklenen kayıtlar koordinatlı coğrafi objelerdir. Bu coğrafi objeleri harita üzerinde görebilmek için pgadmin penceresinin bir özelliğinden yararlanılır. Bu işlem için internetinizin olması gerekir. *Şekil 86* nokta2 tabakasına eklenmiş iki kaydın listelenmesi görüntüsü vardır. Resmin sağ üst köşesindeki kırmızı kutu içindeki düğme seçildiğinde, eklenen grafik objelerin harita üzerindeki yerlerini gösterecek *Geometry Viewer* sekmesi aktif hale gelir.



Data Output Explain Messages Notifications			
	id	nokta_ad	geom
	[PK] integer	character varying (10)	geometry
1	1	KamanMyo	0101000020E6100000C52EC0F83DD940402015438894AD4340
2	2	Uygulamalı	0101000020E6100000E5DEBFE2BCD940403EE88367BBAD4340

Şekil 86

Şekil 87 Geometry Viewer sekmesinin aktif olduğu görüntüdür. "nokta2" tablosuna eklenen iki nokta harita üzerinde mavi noktalar olarak görülmektedir.



Şekil 87

Tabloya grafik obje eklerken, zorunlu olmayan alanların girilmesi gerekmez. Hatta sadece grafik obje sahası girilebilir.

Örnek: Kaman içinde var olan üç adet okul nokta grafik objesi olarak eklenmesi istenmektedir.

ifade 42 istenilen tablonun oluşturulması için gerekli SQL komut bloğudur. Kod içinde sadece “id” isimli saha boş bırakılmayacağı belirtilmiş. Fakat “id” sahası *serial* veri tipinde tanımlanmış, yani saha *integer* veri tipinde tam sayı değerler ile otomatik olarak doldurulacaktır.

ifade 42

```
create table okulnokta(
  id serial not null,
  okul_ad varchar(50),
  geom geometry('point', 5255),
  constraint okulnokta_pkey primary key(id)
);
```

Tablo 10

Okul Adı	Y	X
MELİKSAH	561588.633	4358670.148
CUMHURİYET	561024.605	4358827.728
FAİKGÜNGÖR	562189.395	4358592.207

Tablo 10 okul adları ve koordinat bilgilerinin listesini içerir. *ifade 42* içerisinde tablo oluşturulurken, grafik obje tipi nokta olarak belirtilmiş. Koordinat sistemi de 5255 EPSG kodu ile TUREF datumu 33° dilim orta meridyeni Transverse Mercator projeksiyon koordinatları olarak belirtilmiş. *ifade 43* peşi sıra üç nokta objesinin eklenmesine dair kod bloğunu içerir. Nokta koordinatları önce Y sonra X değerleri olacak sıralamada eklenmiştir.

Kod bloğu içinde sadece geometry veri tipindeki sahaya ait veriler eklenmiştir. Diğer saha okul adının eklenmesine gerek duyulmamıştır.

ifade 43

```
insert into okulnokta(geom)
values(st_geomfromtext('point(561588.633 4358670.148)',5255));
insert into okulnokta(geom)
values(st_geomfromtext('point(561024.605 4358827.728)',5255));
insert into okulnokta(geom)
values(st_geomfromtext('point(562189.395 4358592.207)',5255));
```

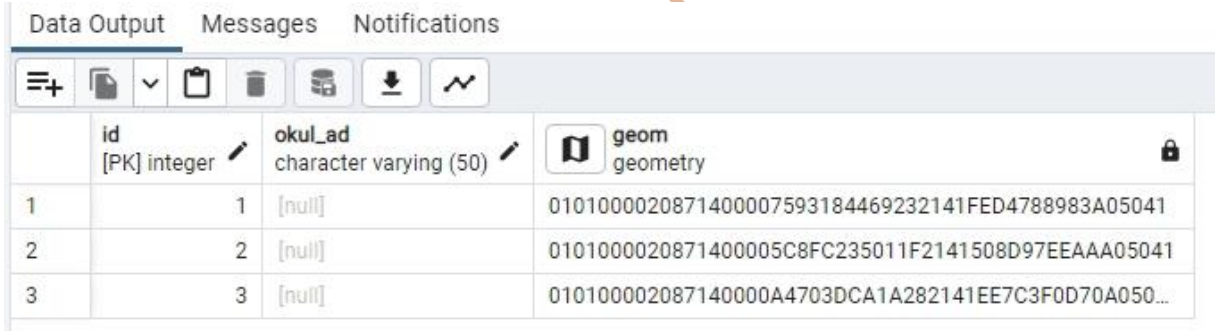
Şekil 88 pgadmin içinde kod bloğu yazımını göstermektedir, Şekil 89 eklenen kayıtların tabloda listelenmesi, Şekil 90 ise eklenen kayıtlara göre noktaların haritada gösterilmesinin tasviridir.



Query Query History

```
1 insert into okulnokta(geom) values(st_geomfromtext('point(561588.633 4358670.148)',5255));
2 insert into okulnokta(geom) values(st_geomfromtext('point(561024.605 4358827.728)',5255));
3 insert into okulnokta(geom) values(st_geomfromtext('point(562189.395 4358592.207)',5255));
```

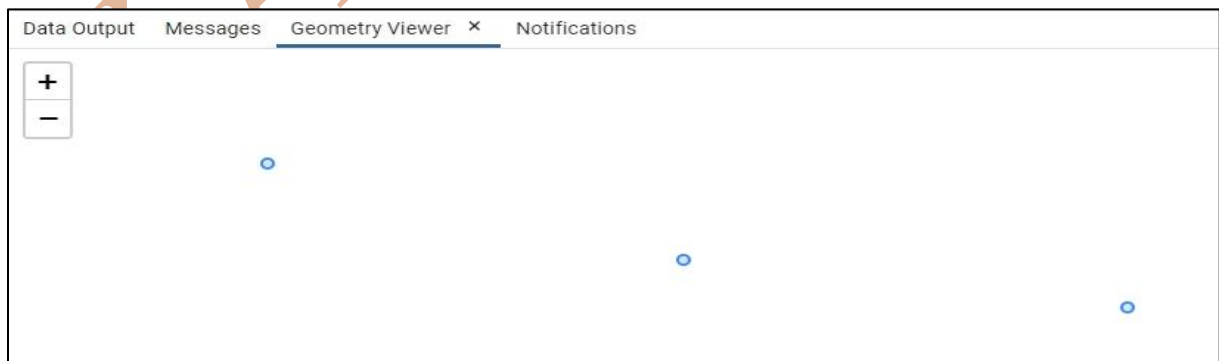
Şekil 88



Data Output Messages Notifications

	id [PK] integer	okul_ad character varying (50)	geom geometry
1	1	[null]	0101000020871400007593184469232141FED4788983A05041
2	2	[null]	0101000020871400005C8FC235011F2141508D97EAAA05041
3	3	[null]	010100002087140000A4703DCA1A282141EE7C3F0D70A050...

Şekil 89



Şekil 90

ST_GeomFromText() Fonksiyonu ile Çizgi (LineString) Grafik Objесinin Eklenmesi:

Çizgi grafik objesi, PostGreSQL veri tabanı yönetim sisteminde tanımlanan bir tabloda çizgi geometrik objelerinin de kayıt alınması istenirse, sahanın veri tipi *Geometry* olarak belirlenmelidir. Veri tipinin özelliği çizgiye uygun olacak şekilde *LineString* olarak düzenlenmelidir.

ifade 44 “cizgiTM” isimli tablo oluşturulması için kullanılan SQL kod bloğu görülmektedir. Tabloda çizgi geometrik objelerini tutacak “geom” isimli *geometry* veri tipinde saha *LineString* özelliğinde oluşturulmuştur. Ayrıca önceki örneklerden farklı olarak koordinat sistemin TM projeksiyonunun 33° boylamının dilim orta meridyeni olması için EPSG kodu 5255 olarak ayarlanmıştır.

```
ifade 44
create table cizgitm(
  id serial not null,
  cizgi_ad varchar(10),
  geom_ciz geometry(linestring, 5255)
);
```

ifade 45 oluşturulan noktaTM tablosuna çizgi objesinin eklenmesi işlemini SQL kod bloğudur.

```
ifade 45
insert into cizgitm(cizgi_ad, geom_ciz)
values('kaldırım',
st_geomfromtext('LINESTRING(560052.278 4358237.440,
560061.006 4358239.563,
560074.812 4358240.968)',
5255));
```



ifade 45'de koordinat sistemi olarak TUREF datum TM projeksiyon dilim orta 33° boylam değerinde meridyen olacak şekilde 5255 EPSG kodu verilmiştir. Koordinat değerleri X– Y koordinatları ile ifade edilir. Koordinat değerleri yazılırken ilk önce Y koordinatı sonra X koordinatı yazılmıştır.

Coğrafi Bilgi Sistemleri II

Data Output Messages Notifications

	id [PK] integer	cizgi_ad character varying (50)	geom_ciz geometry
1	2	kaldırım	010200002087140000030000001904568E68172141C3F5285C17A05041986E12037A172141273108E417A05041C9768E9F9517214146B6F33D18A05...

Şekil 91

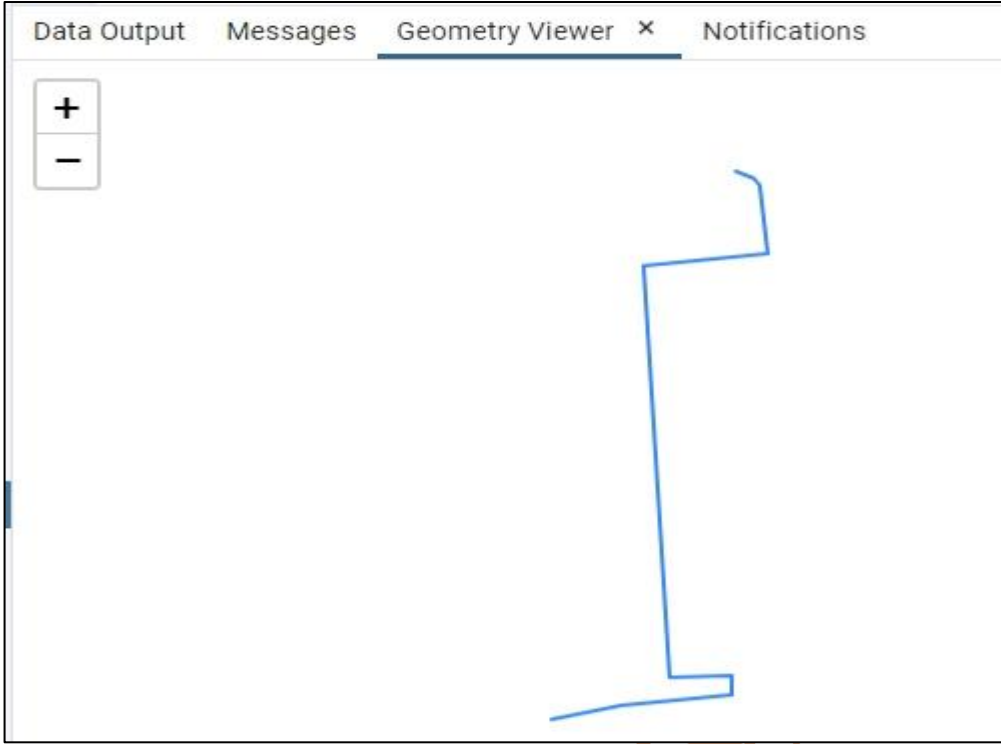


Şekil 92

Query Query History

```
1 insert into cizgitm(cizgi_ad,geom_ciz)
2 values('kaldırım2',st_geomfromtext('linestring(560074.812 4358240.968,
3 560074.725 4358243.739,
4 560067.105 4358243.606,
5 560063.799 4358302.340,
6 560079.168 4358303.877,
7 560078.219 4358313.677,
8 560077.977 4358314.254,
9 560077.574 4358314.677,
10 560076.686 4358315.166,
11 560075.222 4358315.630
12 )'
13 ,5255));
```

Şekil 93



Şekil 94

ST_GeomFromText() Fonksiyonu ile Alan/Çokgen (Polygon) Grafik Objesinin Eklenmesi:

ifade 46 alan grafik objesinin ekleneceği bir tablonun oluşturulduğu kod bloğu. Alan grafik objesinin tablo içinde eklenmesi için Geometry saha veri tipi özelliği Polygon olarak belirtilmiştir.

ifade 46

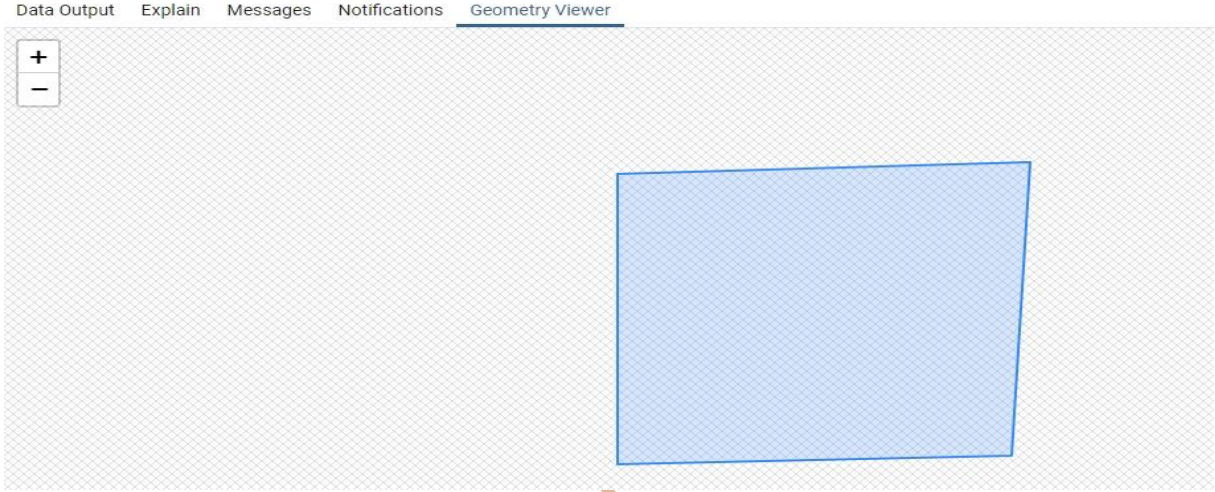
```
create table alanTM(
  id serial not null,
  alan_ad text,
  geoma geometry(polygon, 5255),
  constraint alanTM_pkey primary key(id)
);
```

ifade 47 “alanTM” isimli tabloya alan grafik objesinin eklenmesi kod bloğudur. polygon objesinin eklenmesinde dikkat edilmesi gerekenler: başlangıç ve bitiş noktaları aynı olmak zorunda, POLYGON ifadesinden sonra iki ayrı parantez açılmıştır.

ifade 47

```
insert into alanTM(alan_ad,geoma)
values('mesken1',st_geomfromtext(
'POLYGON((560497.119 4358546.462,
560497.119 4358554.940,
560506.868 4358546.283,
560506.427 4358555.716,
560497.119 4358546.462
))',5255));
```

Şekil 95 eklenen alan objesinin görüntüsüdür.



Şekil 95

St_GeomFromText() fonksiyonunun farklı kullanımı

Altındaki kod bloğunda Kaman ilçesi içindeki marketlerin nokta grafik objesiyle kayıt altına alınması için tablo oluşturulması örneği bulunmaktadır. Oluşturulan tabloda geom adlı geometry veri tipinde ve nokta grafik objelerini saklayacak saha bulunmaktadır.

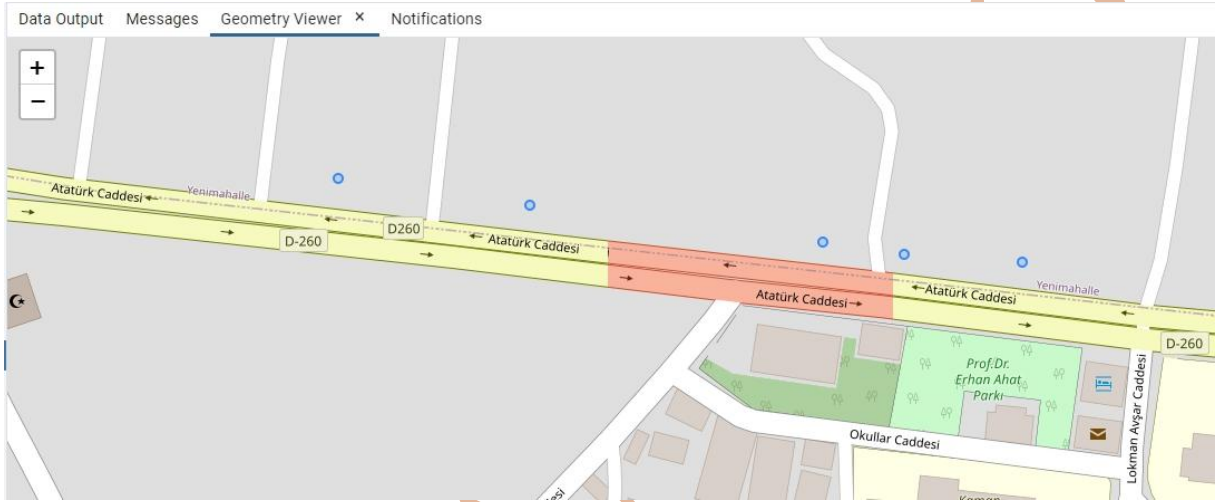
```
create table noktawgs84(
  id serial not null,
  nokta_ad varchar(50),
  geom geometry('point',4326),
  primary key(id)
);
```

Daha önceki tabloya veri ekleme işlemlerinden farklı olarak aşağıda **Values() ifadesi kullanılmamıştır**. Ekleme işlemi için *select* ifadesi ve peşinden *st_geomfromtext()* fonksiyonu kullanılmıştır.

```
insert into market(geom) select st_geomfromtext('point(33.717117 39.359722)',4326);
```

Atatürk Caddesi üzerindeki marketlerin wgs 84 datumunda jeodezik enlem ve jeodezik boylam değerlerine göre nokta grafik objelerin eklenmesine dair kod satırları aşağıda listelenmiştir.

```
insert into market(geom) select st_geomfromtext('point(33.718039 39.359624)',4326);  
insert into market(geom) select st_geomfromtext('point(33.719448 39.359487)',4326);  
insert into market(geom) select st_geomfromtext('point(33.719840 39.359443)',4326);  
insert into market(geom) select st_geomfromtext('point(33.720408 39.359414)',4326);
```



Şekil 96

St_MakePoint(), St_Makeline(), St_MakePolygon() Fonksiyonları

St_GeomFromText() fonksiyonunun iki parametresi vardı. İlk parametre metin ifade olarak nokta, çizgi, alan objelerini eklemek için kullanılıyordu. İkinci parametre olarak grafik objenin koordinatlarının bağlı olduğu koordinat sistemi için kullanılıyordu. St_GeomFromText() üç grafik objenin eklenmesi için kolay ve program kodu için kullanıma uygundur.

St_MakePoint(), St_MakeLine ve St_MakePolygon() fonksiyonları içerisine coğrafi objenin koordinat değerlerini alır. Grafik objenin koordinat değerlerini almazlar. Fakat St_GeomFromText() fonksiyonundan daha hızlı çalışırlar (The Open Source Geospatial Foundation, 2023).

St_MakePoint() Fonksiyonu ile Nokta Grafik Objesi Ekleme

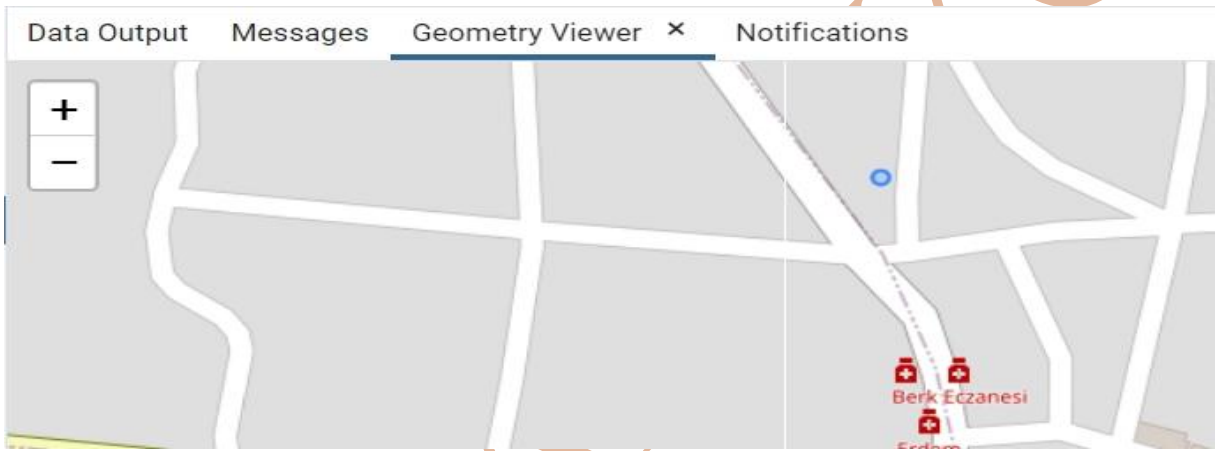
ifade 48 St_MakePoint() fonksiyonunun kullanımına örnektir. St_MakePoint() fonksiyonu içine sadece noktanın sağa ve yukarı artan koordinat değerlerini alır. Örnekte WGS 84 datumunda jeodezik boylam ve jeodezik enlem değerleri eklenmiş. St_MakePoint() fonksiyonu grafik objenin koordinat sistemi bilgisi almaz. Bu yüzden objenin koordinatlarının

tanımlanması için *st_setsrid()* fonksiyonu ile beraber kullanılmıştır. *St_SetSRID()* fonksiyonu iki parametre alıyor, ilk parametre geometrik obje, ikinci parametre ise koordinat sisteminin EPSG kodu. *ifade 48* “market” tablosunun “geom” sahasına bir nokta grafik objesinin eklenmesinin örneğidir. *Şekil 97* noktanın koordinatlarına göre interaktif haritadaki yerinin temsilidir.

```

ifade 48
insert into market(geom)
values(st_setsrid(
st_makepoint(33.723014, 39.360723)
,4326));

```



Şekil 97

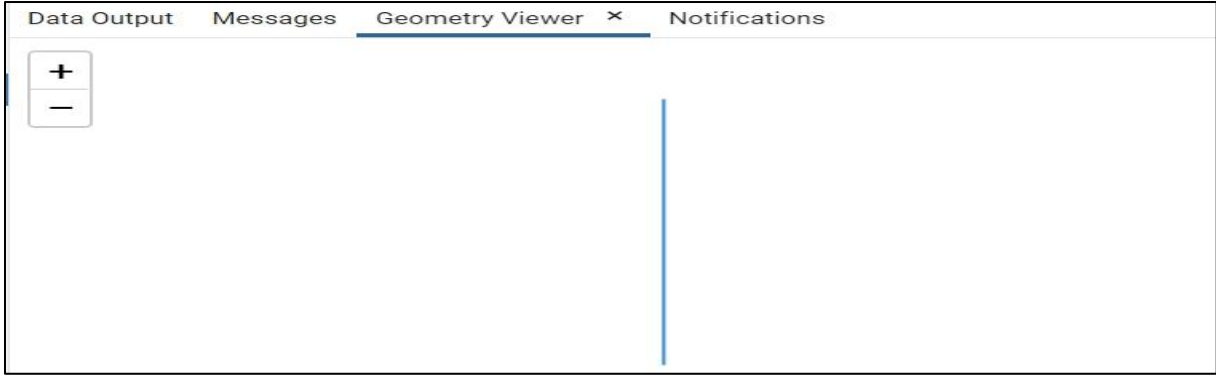
St_MakeLine() Fonksiyonu ile Çizgi Grafik Objesi Ekleme

ifade 49 *st_makeline()* fonksiyonu ile çizgi objesini ekleme örneğidir. Örnek incelendiğinde, çizgiyi oluşturan başlangıç ve bitiş noktaları *st_makepoint()* fonksiyonu ile oluşturulmuştur. Dikkat edilmesi gereken diğer bir husus *st_makeline()* fonksiyonu içinde dizi değişken oluşturulmamışsa, *ifade 49* örneğinde olduğu gibi, sadece iki noktadan oluşan bir çizgi objesi oluşur.

```

ifade 49
insert into cizgi_tm(geom_c)
values(st_makeline(
st_setsrid(st_makepoint(562342.04135135,4358612.70300687),5255),
st_setsrid(st_makepoint(562341.94044837,4358667.33662531),5255)
));

```



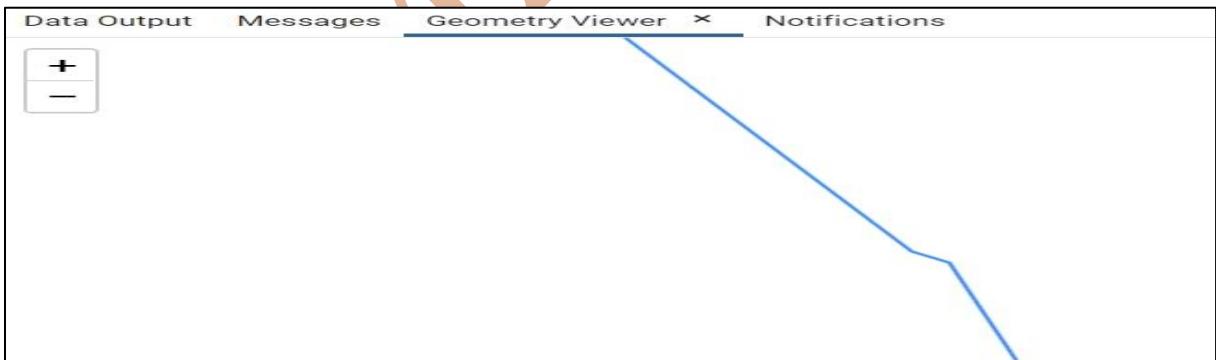
Şekil 98

Eğer *st_makeline()* fonksiyonu ile iki noktadan fazla kırılma noktası olan bir çoklu doğru oluşturmak istiyorsak, *st_makeline()* fonksiyonu içinde dizi şeklinde noktalar oluşturmalıyız. ifade 50 *st_makeline()* fonksiyonu ile bir çoklu doğru objesinin oluşturulmasına dair bir sql kod örneğidir. Kod içinde mavi ile boyanmış *array[]* parametresi içine dört adet nokta objesi eklenmiştir. Şekil 99 çoklu doğrunun haritada temsilidir.

ifade 50

```

insert into cizgi_tm(geom_c)
values(st_makeline(array[
st_setsrid(st_makepoint(562341.94044837,4358667.33662531),5255),
st_setsrid(st_makepoint(562311.56864818,4358761.94868483),5255),
st_setsrid(st_makepoint(562294.65226766,4358773.20927341),5255),
st_setsrid(st_makepoint(562163.94506370,4358985.22436296),5255)
]));
  
```



Şekil 99

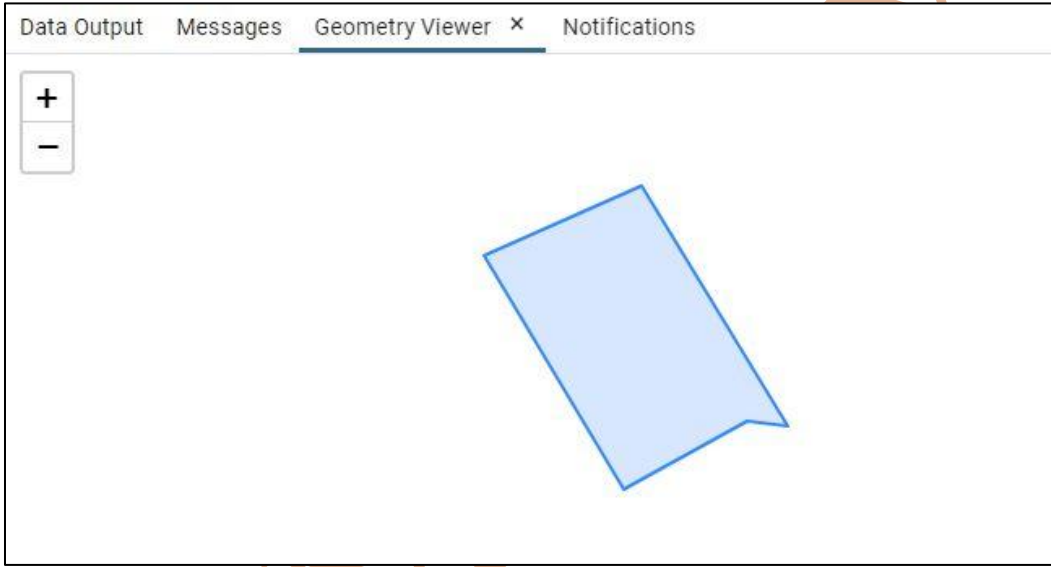
St_MakePolygon() Fonksiyonu ile Alan Grafik Objesi Ekleme

ifade 51 *st_makepolygon()* fonksiyonu kullanılarak alan grafik objesinin eklenmesinin örneğidir. Örnek incelendiğinde, fonksiyon içine direkt olarak nokta objeleri direkt olarak eklenmiyor, *st_makeline()* fonksiyonu içine noktalar eklenerek alan grafik objesinin köşe

noktaları oluşturuluyor. Köşe noktalar sayesinde alan grafik objesi de oluşmuş oluyor. *Şekil 100* oluşan alan grafik objesinin interaktif harita üzerinde gösterilmesidir.

ifade 51

```
insert into alan(alan_geom) values(st_makepolygon(st_makeline(
array[
st_setsrid(st_makepoint(562117.48156790,4358960.57836045),5255),
st_setsrid(st_makepoint(562165.83204226,4358981.79900161),5255),
st_setsrid(st_makepoint(562211.10988734,4358908.14258878),5255),
st_setsrid(st_makepoint(562198.33126386,4358909.41119608),5255),
st_setsrid(st_makepoint(562160.40737841,4358888.27360155),5255),
st_setsrid(st_makepoint(562117.48156790,4358960.57836045),5255)
]
)));
```



Şekil 100

Tabloda Kayıtlı Grafik Objelerin Koordinat Sisteminin Dönüşümü

PostgreSQL veri tabanı yönetim sisteminde tabloları oluştururken grafik objeleri tutacak sahan oluşturulurken, grafik objenin koordinat bilgilerinin bağlı olduğu koordinat sistemi de belirlenmelidir. Koordinat sistemi belirlenirken, koordinat sisteminin EPSG koordinatlarından yararlanılabilir.

Eğer tabloda kayıtlı grafik objelerin tanımlı koordinat sistemi farklı bir koordinat sistemine dönüştürülmek isteniyorsa *st_transform()* fonksiyonu kullanılır. Fonksiyon içine iki adet parametre alır. İlk parametre grafik objenin tutulduğu sahanın adı, ikinci parametre **dönüştürülecek olduğu** koordinat sisteminin EPSG kod değeridir.

Şekil 101 “market” tablosunda kayıt edilen grafik objelerin koordinat bilgisini 4326 EPSG kodu ile göstermektedir. Koordinatlar WGS 84 datumunda jeodezik enlem ve jeodezik boylam olarak temsil edilmektedir.

Query		Query History			
1	<code>select srid from geometry_columns where f_table_name='market';</code>				
Data Output		Messages		Notifications	
	srid integer				
1	4326				

Şekil 101

Tablodaki objelerin koordinat sistemi TUREF datum Transverse Mercator Projeksiyon Koordinatlarında dilim orta meridyeni 33° olan koordinat sistemine dönüştürülmek istenmektedir. Bu koordinat sistemi 5255 EPSG kodu değeriyle temsil edilir.



Koordinat dönüşümü işleminde grafik verilerin zarar görmesi olasılığını düşünerek, ilk önce koordinat dönüşümü yapılacak grafik objeleri tutan tablonun yedeklenmesi veri kaybını önleyebilir (*ifade 52*).

ifade 52

*Select * into table kopya_tablo from tablonun_kendisi*

ifade 53 koordinat sistemi dönüşümü için kullanılan SQL cümlecini göstermektedir.

ifade 53

*select st_astext(
st_transform(st_setsrid(geom,4326),5255))
from market;*

Data Output		Messages		Notifications	
	st_astext text				
1	POINT(561805.8092899208 4358685.230796895)				
2	POINT(561885.3605509795 4358674.981629072)				
3	POINT(562006.919903419 4358660.737483097)				
4	POINT(562313.1669921591 4358800.416958849)				
5	POINT(562040.7444046775 4358656.121554967)				
6	POINT(562089.7247227713 4358653.292082658)				

Şekil 102

Grafik Veri Tutan Tablodaki Grafik Bilgilerin Sorgulanması

Oluşturulan tabloda grafik verilerin kaydını tutabilmek için tabloda *Geometry* veri tipinde saha oluşturulmalı. Sahada hangi tipte grafik objenin kaydedileceği ve grafik objelerin detaylarına ait koordinatların tanımlı olduğu koordinat sistemi belirtilmelidir (Koordinat sistemi belirtilmesi zorunluluğu yoktur. Tanımlanması tabloda yapılacak işlemlerin etkinliği için önem arz eder.).

SQL sorgulama dili yardımıyla tablo içindeki sahalarda tutulan veri, veriler arası bağlantı sayesinde diğer sahaların verileri sorgulanabilir. Eğer tablo sahalarında tutulacak veri tipi *geometry* olarak belirlenmişse, o sahada kaydı tutulacak grafik obje bilgileri sorgulanabilir veya grafik objenin diğer grafik objeler ile olan ilişkileri (kesişmesi, üst üste binmesi, diğerinin içinde olması,...) sorgulanabilir.

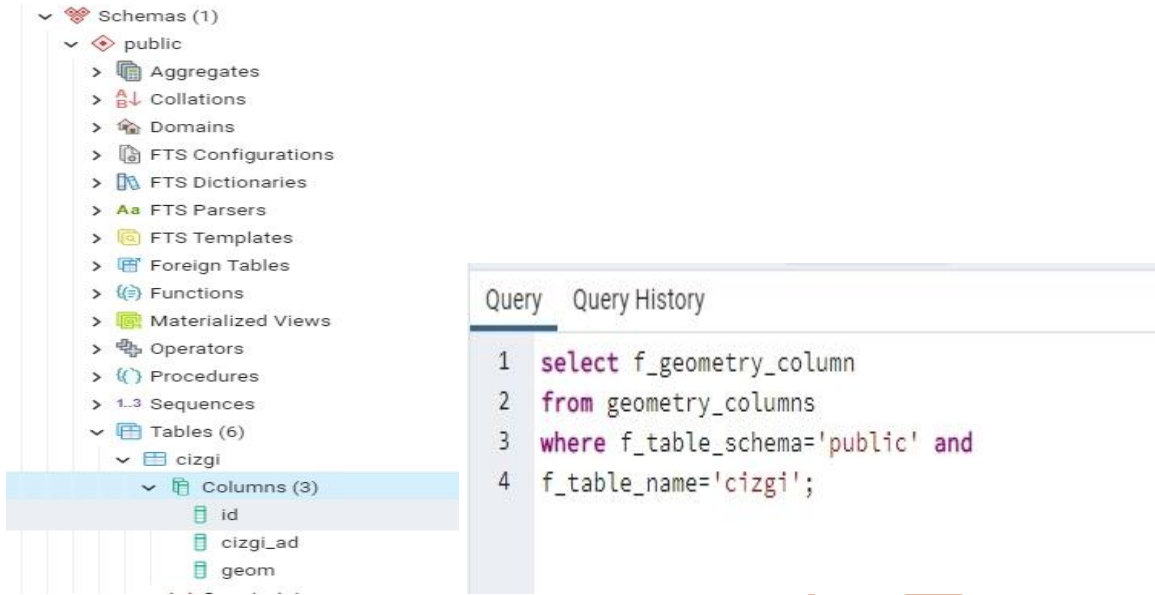
PostgreSQL veri tabanı yönetim sistemine eklenen PostGIS eklentisinin bazı özel SQL komutları sayesinde *Geometry* veri tipinde olan sahaların sorgulanması için özel komut satırları bulunmaktadır. İşlemlerin yapılması için *From* komutunda tablonun adı direkt olarak verilmez. *From* komutunda *geometry_columns* komutu kullanılır. *Where* komutunda ise *schema* ve tablonun adı sınıma olarak kullanılır.

Tablo Sahaları İçinde Geometry Veri Tipinde Veri Tutan Sahanın Adının Sorgulanması

Tablo içinde hem coğrafi objeye ait grafik verilerin tutulduğu (*Geometry* veri tipinde verilerin tutulduğu) saha da olabilir, hem de coğrafi objenin öznitelik verilerinin tutulduğu sahalarda olabilir. Tablo içinde sadece *Geometry* veri tipinde sahanın tutulduğu sahanın adını öğrenmek için kullanılması gereken kod satırı:

Şekil 103 sağ resimde 2 numaralı satırdaki *from* komutunda *geometry_columns* ifadesi, tablodaki grafik verileri tutan sahanın özelliklerini sorgulanacağını belirtir. Normal bir SQL cümlecğinde *from* komutunda sorgulamada kullanılacak olan tablo veya tabloların isimleri kullanılır. *Şekil 103* sağ resim 3. ve 4. satırda *Where* komutunda kullanılan kısıtlamalarda, *geometry* sahası sorgulanacak tablo 4. satırda *f_table_name* ifadesiyle sınırlandırılmış, 3. satırda ise kısıtlamada belirtilen tablonun içinde bulunduğu şema *f_table_schema* ifadesiyle sınırlandırılmış. Sorguda *select* komutunda, *geometry* veri tipinde verinin kaydedildiği sahanın ad bilgisinin listelenmesi (seçilmesi) için *f_geometry_column* ifadesi kullanılmıştır.

Sorguda “public” şeması altındaki “cizgi” adlı tabloda yer alan *geometry* veri tipinde veri tutan sahanın isminin listelenmesi sağlanmıştır (*Şekil 103* sağ resim). “cizgi” tablosundaki *geometry* veri tipinde verinin tutulduğu sahanın “geom” adındadır (*Şekil 104*).

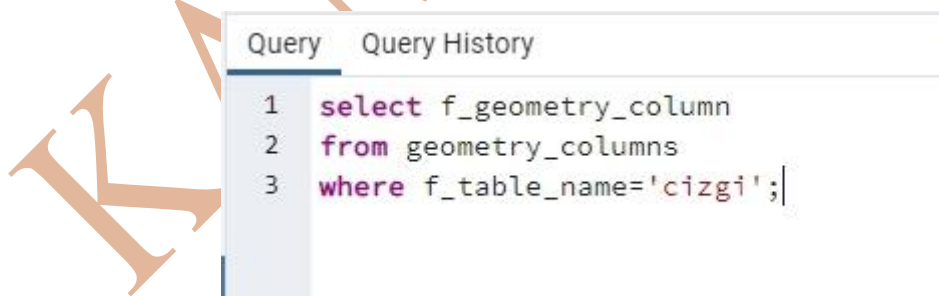


Şekil 103

f_geometry_column	
name	
1	geom

Şekil 104

Eğer aktif (seçili olan) veritabanı “cizgi” veritabanı ise aşağıdaki kod bloğu da çalışacaktır ve aynı sonucu verecektir (Şekil 105).



Şekil 105

Tablo Sahaları İçinde Geometry Veri Tipinde Veri Tutan Sahanın Geometrik Veri Tipinin Sorgulanması

Tablo içinde *Geometry* veri tipinde veri saklayan sahanın, sakladığı coğrafi objeyi temsil etmek için kullanılan geometri veri tipini öğrenmek için *Select* komutunda *type* ifadesi kullanılır.

Şekil 106, Kaman ilçesinde Atatürk caddesi üzerindeki marketlere ait verilerin tutulduğu, “market” isimindeki tabloda coğrafi objeyi temsil etmek için kullanılan grafik obje tipinin nokta olduğunu (POINT) sorgulayan komut bloğunun (sol resim) hem sonucunu (sağ resim) göstermektedir.

Query		Query History	
1	<code>select type</code>		
2	<code>from geometry_columns</code>		
3	<code>where f_table_name='market';</code>		

Data Output		Messages		Notifications	
	type				
	character varying (30)				
1	POINT				

Şekil 106

Şekil 107, Kaman Meslek Yüksekokulundaki kaldırım coğrafi objelerinin tutulduğu, “cizgi” tablosunda coğrafi objenin temsili için kullanılan grafik obje tipinin çizgi (LINESTRING) olduğunu göstermektedir.

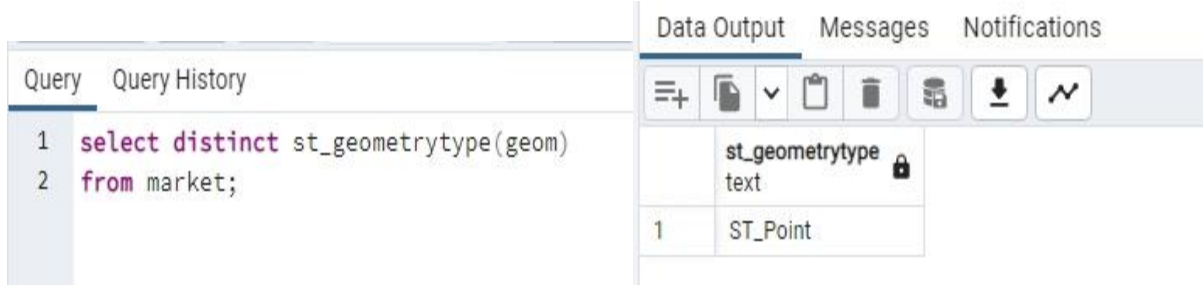
Query		Query History	
1	<code>select type</code>		
2	<code>from geometry_columns</code>		
3	<code>where f_table_name='cizgi';</code>		

Data Output		Messages		Notifications	
	type				
	character varying (30)				
1	LINESTRING				

Şekil 107



Eğer tablo içinde *geometry* veri tipindeki verinin kayıt altına alındığı sahanın ismi biliniyorsa, kayıt yapılan grafik objenin geometri tipi aşağıdaki kod bloğu ile farklı bir yöntemle belirlenebilir. Şekil 108 “market” tablosu içinde *geometry* veri tipinde kayıtların yapıldığı sahanın ismi “geom” sahasıdır. Saha içinde kayıt altına alınan coğrafi objeyi temsil eden geometrik obje tipini öğrenmek için *st_geometrytype()* fonksiyonu kullanılmıştır. Fonksiyon her kayıta ait aynı obje tipini geri döndürecektir. Bunu engellemek için *distinct* ifadesiyle gruplandırma yapılmıştır.



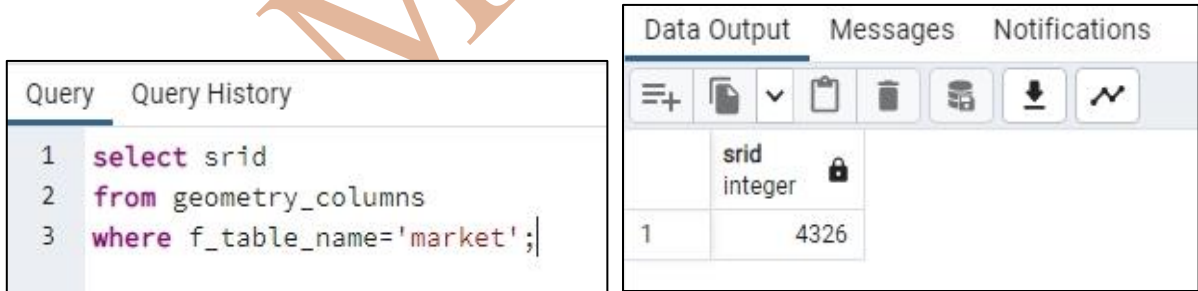
Şekil 108

Tablo İçinde Geometry Veri Tipinde Veri Tutan Sahada Verinin Koordinat Sisteminin Sorgulanması

Tablo içinde Geometry veri tipinde veri kaydının yapıldığı saha varsa, veri bir coğrafi objeyi temsil eden geometrik objedir. Coğrafi objenin kendisi (bir nokta grafik objesiyle temsil ediliyor olabilir) veya detayları/köşeleri (çizgi veya alan grafik objesiyle temsil ediliyor olabilir) bir koordinat sistemine göre koordinatlandırılmıştır.

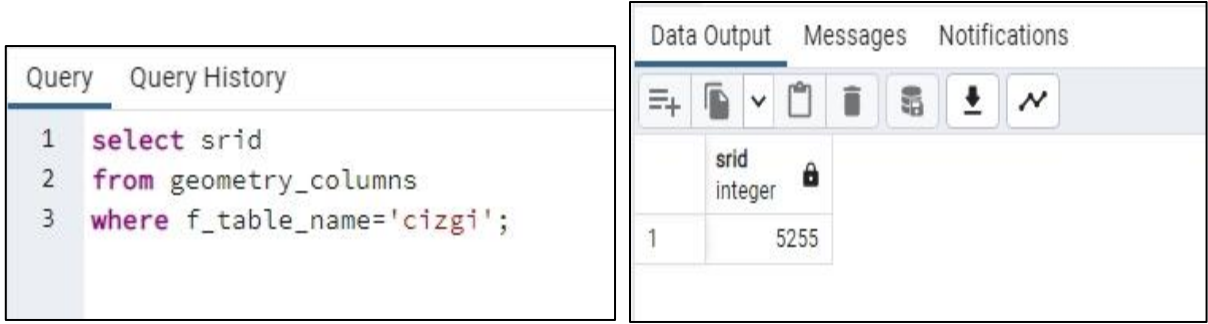
Tablodaki sahanın koordinat sistemi tanımlanmışsa ve bu koordinat sistemi öğrenilmesi gerekiyorsa *select* komutunda *srid* ifadesi kullanılır. SRID, spatial reference identifier (konumsal referans tanımlayıcı) kelimelerinin kısaltmasıdır. EPSG kodlarıyla ifade edilmektedir.

Şekil 109 “market” tabakasında 4326 EPSG kodlu koordinat sistemine göre nokta objeleri kayıt altına alınmaktadır. 4326 EPSG kodunda nokta koordinatları, wgs 84 datumunda jeodezik enlem ve jeodezik boylam koordinatlarıyla kayıt altına alınmaktadır.



Şekil 109

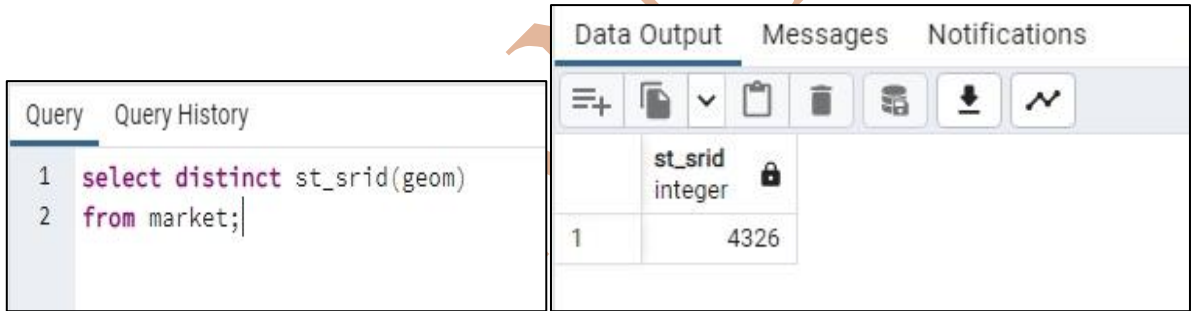
Şekil 110 “çizgi” tablosunda kayıt altına alınan çizgi grafik objesinin köşe koordinatlarının 5255 EPSG kodlu koordinat sisteminde kayıt altına alınmakta olduğunu belirtmektedir. 5255 EPSG kodunda noktalar, TUREF datumunda TM projeksiyon koordinatlarında kayıt altına alınmaktadır.



Şekil 110



Eğer tablo içinde geometry veri tipindeki verinin kayıt altına alındığı sahanın ismi biliniyorsa kayıtların yapıldığı koordinat sistemi aşağıdaki kod bloğu ile farklı bir yöntemle belirlenebilir. Şekil 111 (sol resim) *select* kodunda koordinat sisteminin sorgulanması için *st_srid()* fonksiyonu kullanılmaktadır. Fonksiyon içinde *geometry* veri tipindeki sahanın ismi “geom” sahasının adı kullanılmıştır. *Select* kodunda *distinct* ifadesi kullanılmasaydı, kayıtlı olan tüm grafik objelerin koordinat sistemi EPSG kodu aynı şekilde gelecekti. *Distinct* ifadesi gruplandırma yapılmasını sağladı.



Şekil 111

Tabloda Kayıtlı Nokta Grafik Obje Tipinin Koordinatlarının Sorgulanması

Tablo içinde kaydedilen nokta grafik objelerinin koordinat değerlerini listelemek için *St_x()*, *St_y()* ve *St_z()* fonksiyonları kullanılır. Üç fonksiyonda içine geometrik objenin kaydedildiği, sahanın adını alır. ifade 54 “nokta_ad” isimli tabloda nokta grafik objelerinin tutulduğu “geom” sahasındaki kayıtlı noktaların “nokta_ad” sahasındaki adları, x ve y değerlerinin listelenmesi için gereken SQL kodu bulunmaktadır. Şekil 112 SQL sorgu sonucu listenin görünümüdür.

ifade 54

```
select nokta_ad, st_x(geom), st_y(geom)
from nokta;
```


Data Output Messages Notifications				
	nokta_ad character varying (10)	st_x double precision	st_y double precision	geom geometry
1	1	33.697203725661474	39.35609534526361	0101000020E6100000C52EC0F83DD940402015438894AD4340
2	2	33.70107683533369	39.35728162709027	0101000020E6100000E5DEBFE2BCD940403EE88367BBAD43...
3	3	33.723014	39.360723	0101000020E61000006B4606B98BDC40401FF5D72B2CAE4340

Şekil 112

Tablo Sahaları İçinde Grafik Obje Tipinin Çiziminin Doğruluğunun Sorgulanması

Tablo içine kaydedilmiş objenin geometrisi yanlış çizilmiş olabilir. Özellikle alan grafik obje tipindeki objelerin çiziminde başlangıç noktası ile bitiş noktası aynı nokta olarak çizilmemiş, ya da çokgen çizgi grafik objesiyle çizilmiş olabilir. Objenin sorgulanması için *isvalid()* fonksiyonu kullanılır. *Isvalid()* fonksiyonu içine *geometry* veri tipinde sahanın adını alır ya da bir *geometry* veri tipinde obje alır.

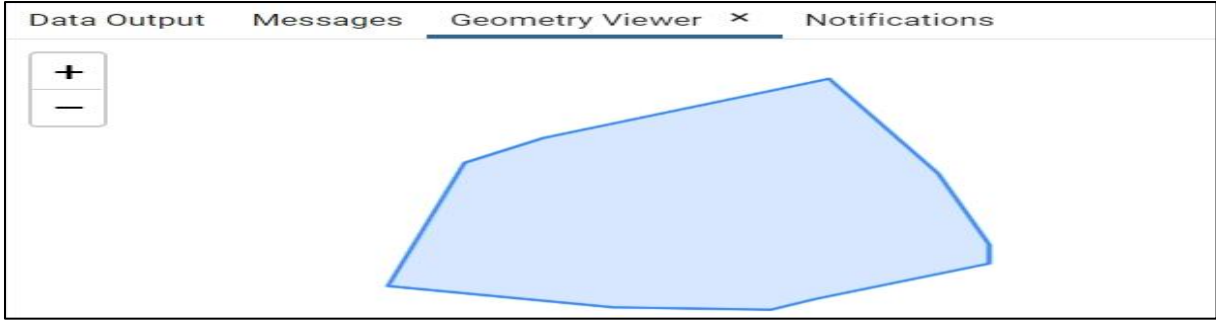
Şekil 113 “alan_tm” tablosu “alan_ad” sahasında “ada2” kaydı olan kaydın geometrisinin doğruluğu *st_isvalid()* fonksiyonu ile sorgulanmasını gösteren sorgu cümlecğini göstermektedir. Sorgu cümlecğinin sonucu *Data Output* sekmesinde gözükmemektedir. Sonuç *true* şeklinde dönmüştür. Çizimde bir sıkıntı olmadığını belirtir.

Query Query History	
1	<code>select st_isvalid(geom_a) from alan_tm where alan_ad='ada2';</code>

Data Output Messages Geometry Viewer × Notifications	
	st_isvalid boolean
1	true

Şekil 113

Şekil 113 resmindeki sorgulanan grafik obje Şekil 114’da gözükmemektedir. Grafik obje alan grafik obje tipindedir ve çizimde bir sıkıntı gözükmemektedir.



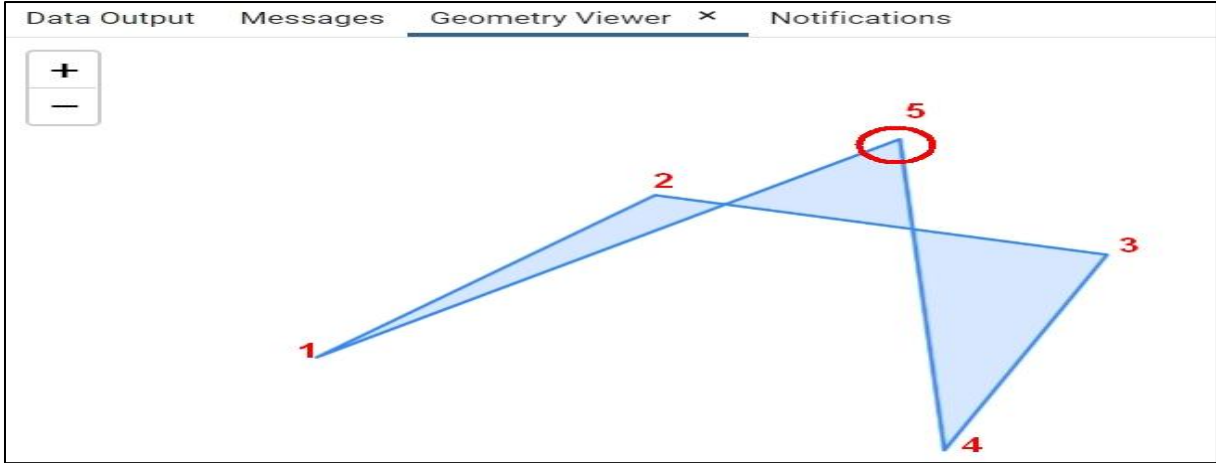
Şekil 114

Şekil 115 “alan_tm” tablosu “alan_ad” sahasında “ada4” verisi olan kayıttaki grafik objenin çiziminin doğru olduğunun sorgu cümlecği gözükmemektedir. Sorgu sonucu *Data Output* sekmesinde gözükmemektedir. *St_isvalid()* fonksiyonun sonucu *false* olarak dönmüştür ve *false* çizimde hata olduğunu söylemektedir.



Şekil 115

Şekil 115’de sorgulanan objenin çizimi Şekil 116’de görülmektedir. Obje çizilirken sırasıyla 1, 2, 3, 4 noktalarından sonra 5 numaralı nokta 2 ve 3 numaralı noktaların arasındaki doğruyu keserek geçmiştir. Çizimde obje kendi kendini kesmiştir. Çizimde bir hata vardır.

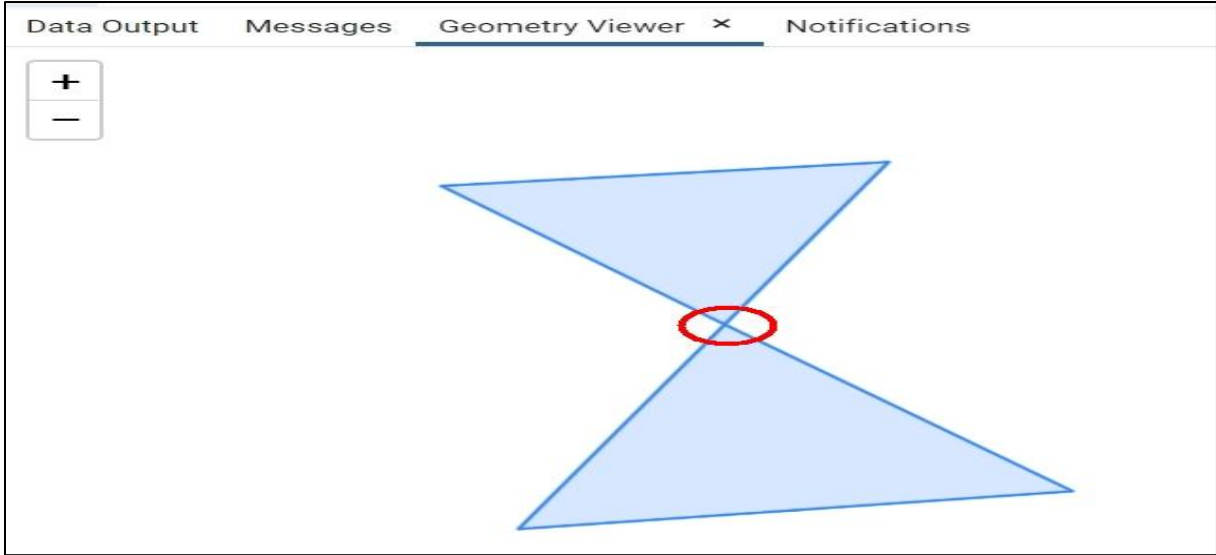


Şekil 116

Şekil 117 benzer bir sorgu örneğidir. Aynı şekilde Isvalid() fonksiyonunun sonucunun false olduğu yani grafik objenin çiziminde bir sorun olduğu belirtilmektedir. Şekil 118 sorgusu yapılan grafik objenin kendisi gözükmemektedir. Obje çiziminde obje kendini keserek çizilmiştir.



Şekil 117



Şekil 118

Şekil 119 bir tablo içinde çizimi yanlış olmuş objelerin listelenmesi örneği vardır. Örnekte alan_tm tablosu içinde *st_isvalid()* fonksiyonu sonucu *false* olan kayıtların tüm kayıt listesi istenmiştir.

Query Query History

```
1 select * from alan_tm where st_isvalid(geom_a)='false';|
```

Data Output Messages Notifications

	id [PK] integer	alan_ad text	geom_a geometry
1	6	ada5	01030000208714000001000000060000005EABF76F5F2621418B730FF2D6A0504114045D
2	5	ada4	0103000020871400000100000006000000578A6DDBFE262141C664FC80D4A0504196DEC

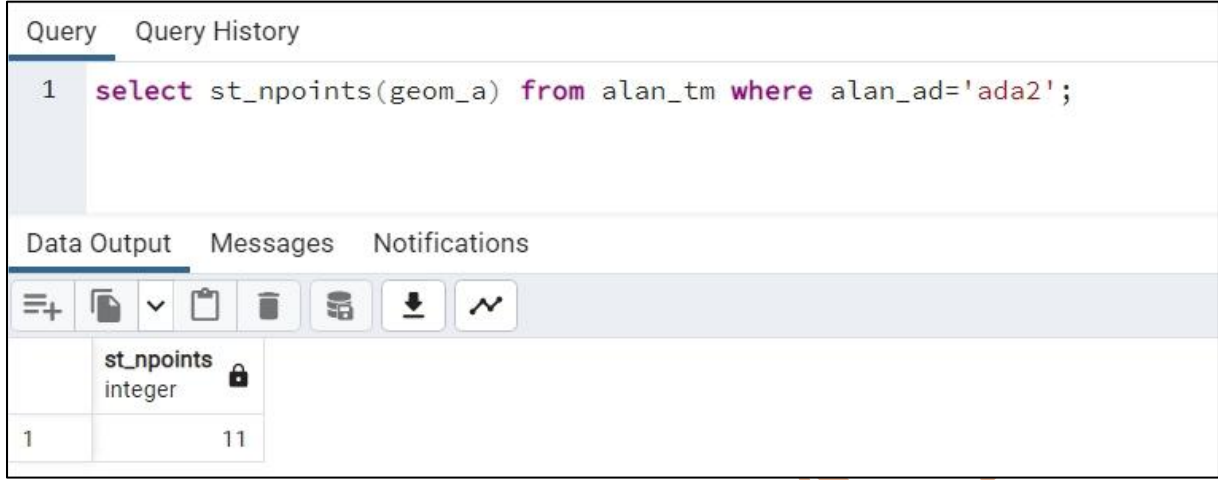
Şekil 119

Tabloda Kayıtlı Grafik Objeyi Oluşturan Nokta Sayısının Sorgulanması

Çizgi objeleri en az 2, alan grafik objeleri en az 3 noktadan oluşur. Grafik objeler arasındaki ilişkilerin sorgulanmasında (alan grafik objesinin bir köşesinin diğer bir alan objesi içinde kalması, çizgi objesinin bir köşesinin diğer bir obje ile kesişmesi,...) bir döngü içinde her bir köşenin sorgulanması gerekebilir. Bu işlemin yapılması için ilk önce objenin kaç noktadan oluştuğunun bilinmesi gerekir. Belirlenen sayıda yapılacak döngü ile her bir köşe sorgu içinde kıyaslanabilir.

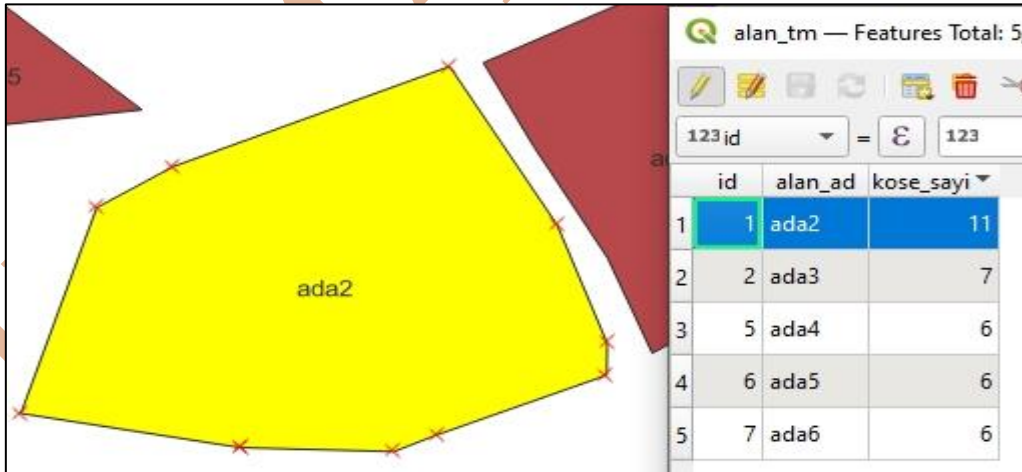
Objelerin kaç noktadan oluştuğunu sorgulamak için *St_npoints()* fonksiyonu kullanılır. Fonksiyon içine *geometry* veri tipinde sahanın adını alır.

Şekil 120 st_npoints() fonksiyonunun kullanımına bir örnektir. Query sekmesinde sorgu cümleciği yazılmıştır. Sorgu cümleciğinde, alan_tm tablosu alan_ad sahasında ada2 verisi olan kayda ait geometrik objenin köşe sayısı st_npoints(geom_a) ile listelenmiştir.



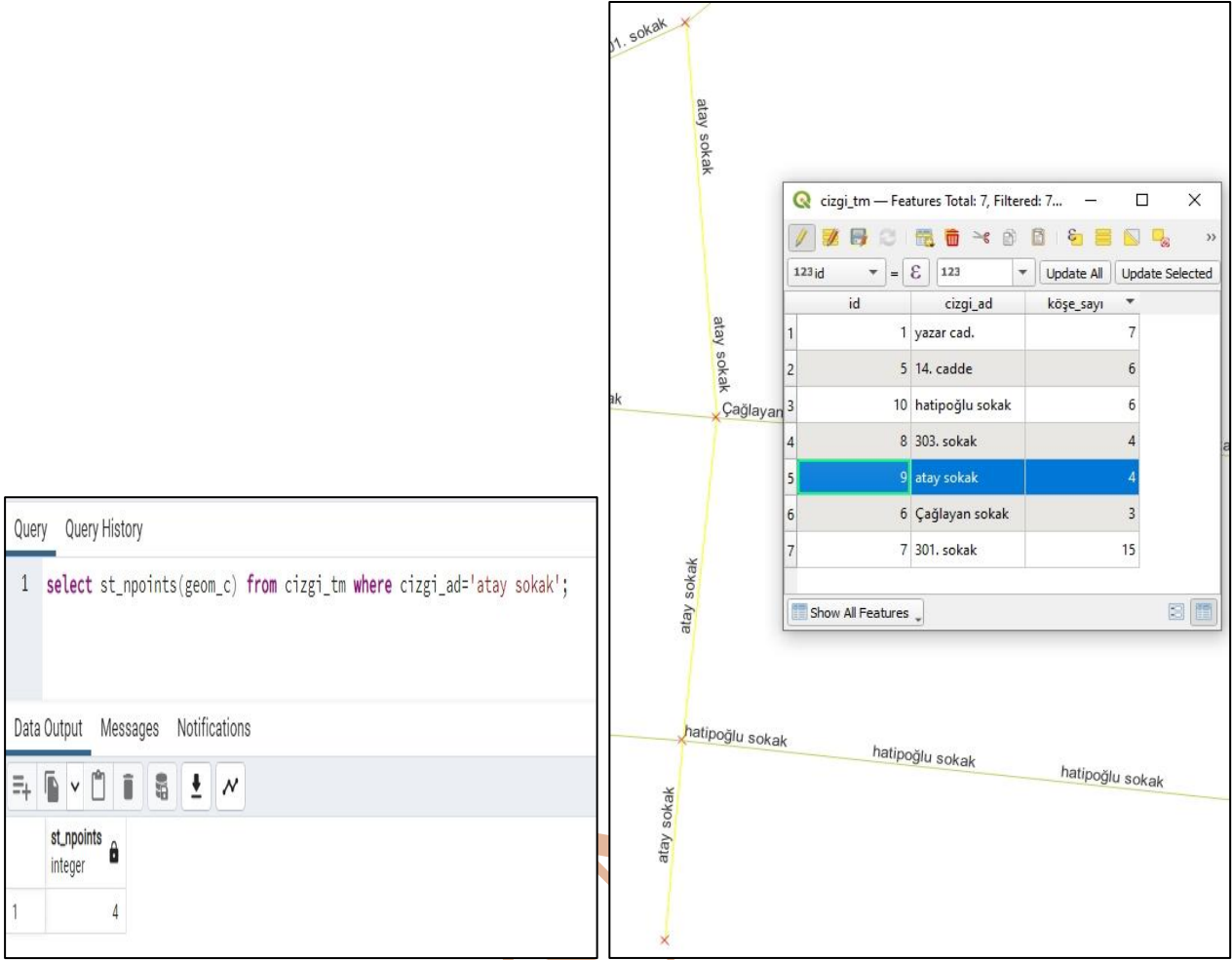
Şekil 120

Şekil 121, Şekil 120’de yapılan sorgunun ispatıdır. Qgis yazılımında alan_tm tablosu (tabakası) içinde alan_ad sahasında ada2 veri kaydı olan saha seçilmiş ve bağlantılı olduğu grafik obje harita üzerinde seçili hale gelmiştir. Seçilen objenin köşelerinde ^x sembolü konulmuştur. Her CAD veya CBS yazılımında alan grafik objesine ait köşe sayısı, görünen köşe sayısının bir fazlası olacaktır. Çünkü alan grafik objesinde başlangıç ve bitiş noktaları aynıdır. Bitiş noktası da toplam köşe sayısına eklenir.



Şekil 121

Şekil 122 çizgi objesinin köşe sayısının sorgulanması örneğinin temsilidir. Sorgu cümleciğinde, “çizgi_tm” tabakası “çizgi_ad” sahasında “atay_sokak” verisi olan kayıttın geometrik objesinin nokta sayısı sorgulanmıştır (Şekil 122 sol resim). Sorgu sonucu 4 sayısı geri dönmüştür. Şekil 122 sağ resimde grafik objenin kendisi Qgis yazılımında gösterilmiştir. Seçilen objenin her bir köşesinin ^x sembolü konmuştur.



Şekil 122

Tabloda Kayıtlı Grafik Objenin veya Objeler Arasında Hesaplama İşlemleri

Kullanıcı tabloda kayıtlı olan noktalar arasında oluşan uzunluk bilgisinin hesaplanması, noktalar arasında oluşan semt açısı bilgisi; çizgi grafik objesini oluşturan noktalar arasındaki uzunluk bilgisinin hesaplanması, alan grafik objesini oluşturan noktalar sayesinde objenin alan bilgisi gerektiğinde hesaplanmasını isteyebilir. Bu işlemlerin yapılabilmesi için PostgreSQL yazılımının PostGIS eklentisi içindeki fonksiyonlardan yararlanılacaktır.

Tabloda Kayıtlı Grafik Objelerin Arasındaki Mesafenin Belirlenmesi

Mesafe bilgisi ile uzunluk bilgisi birbirinden farklıdır. Mesafe iki grafik obje arasındaki yatay mesafe değeridir. Uzunluk bilgisi ise çizgi veya alan objesini oluşturan çizgilerin yatay mesafesidir. Mesafe bilgisinin sorgulanmasında kullanılacak iki grafik objenin aynı tipte olması zorunlu değildir. Nokta ile alan objesi arasındaki mesafe veya çizgi ile nokta arasındaki mesafe veya alan ile çizgi grafik objeleri arasındaki yatay mesafe değerleri sorgulanabilir. Mesafe işleminin sorgulanması için *St_distance()* fonksiyonu kullanılır. *St_distance()* fonksiyonu içine

iki adet *geometry* veri tipinde obje istemektedir. *St_distance()* fonksiyonu sonucunda dönen mesafe değeri metre uzunluk birimindedir.

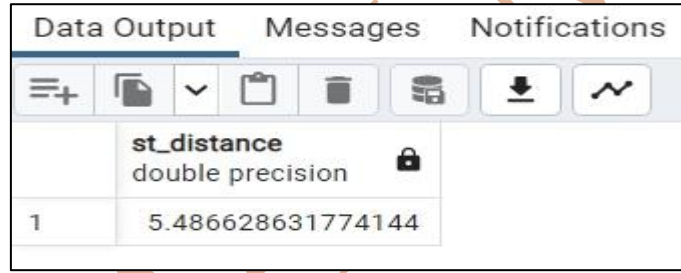
İki Alan Arasındaki Mesafe Hesabı

ifade 55 “alan_tm” adlı tabloda kayıtlı olan ada2 ve ada3 adlarındaki iki alan grafik objenin arasındaki mesafe değerinin listelenmesi (sonucu) işleminin SQL kodudur. Sorgu dikkatli incelendiğinde, *st_distance()* fonksiyonu içine eklenecek iki grafik obje için iki ayrı sorgu eklenmiştir. *St_distance()* fonksiyonu içine bu şekilde grafik objenin eklenmesi gerekmiyor. Örnekte fonksiyon içine grafik objenin eklenmesi için bir yöntem gösterilmiştir.

ifade 55

```
Select distinct st_distance(
(select geom_a from alan_tm where alan_ad = 'ada2'),
(select geom_a from alan_tm where alan_ad = 'ada3')
)
from alan_tm;
```

Şekil 123 iki alan objesi arasındaki



Data Output		Messages	Notifications
	st_distance double precision		
1	5.486628631774144		

Şekil 123

İki Nokta Arasındaki Mesafe Hesabı

ifade 56 iki nokta grafik objesi arasındaki mesafe değerinin elde edilmesi için yazılan SQL dilindeki sorgudur. *Şekil 124* iki nokta arasındaki mesafe sonucunun göstermektedir. Sorgu incelendiğinde, *distinct* ifadesi kullanılmıştır. Eğer *distinct* kullanılmazaydı, aynı sonuç iki kere gösterilecektir. Hem “çarşı camii” noktasından “savcılı taksi” arasındaki mesafe, hem de “savcılı taksi” noktası ile “çarşı camii” arasındaki mesafe değerini gösterecekti. Her iki mesafe değeri de aynı çıkacaktı. *Distinct* ifadesiyle bu sonuç teke indirilmiştir.

ifade 56

```
select distinct st_distance(
(select geom_n from nokta_tm where nokta_ad = 'çarşı camii'),
(select geom_n from nokta_tm where nokta_ad = 'savcılı taksi')
)
from nokta_tm;
```

Data Output		Messages	Notifications
	st_distance double precision		
1	147.68934135574878		

Şekil 124

Çizgi Uzunluğu Hesabı

Çizgi grafik objesinin uzunluğunu belirlemek için *st_length()* fonksiyonu kullanılır. ifade 57 “cizgi_tm” tablosu “cizgi_ad” sahasında “yazar cad.” kaydındaki çizgi geometrik objesinin uzunluğunu öğrenmek için kullanılan SQL kodu bulunmaktadır.

ifade 57

```
select st_length(geom_c)
from cizgi_tm
where cizgi_ad = 'yazar cad.';
```

Şekil 125 yapılan sorgunun sonucunu göstermektedir. Sorgu sonucu objemizin tanımlı olduğu koordinat sistemine göre değişecektir. Örnekteki tabloda geometrik objenin koordinat sistemi 5255 Epsg kodludur. Sonuç uzunluk metre olarak çıkmıştır.

Data Output		Messages	Notifications
	st_length double precision		
1	473.0178330242911		

Şekil 125

Alan Bilgisi Hesabı

Çokgen geometrik objelerin alan bilgisini sorgulayabilmek için *st_area()* fonksiyonu kullanılır. Fonksiyon içine geometrik obje veri tipinde değer alır. Fonksiyon ile dönecek sonuç değer birimi, geometrik objenin tanımlı olduğu koordinat sistemine bağlıdır. ifade 58 “alan_tm” tablosu “alan_ad” sahasında “ada2” verisi olan kayıttaki geometrik objenin alan değeri sorgulanmaktadır.

ifade 58

```
select st_area(geom_a)
from alan_tm
where alan_ad = 'ada2';
```

Şekil 126 sorgu sonucunu göstermektedir. “alan_tm” tablosundaki “geom_a” sahasının koordinat sistemi 5255 Epsg kodundadır. Bu sebeple sonuç alan değeri metrekare birimindedir.

Data Output		Messages	Notifications
	st_area double precision		
1	8500.655783248332		

Şekil 126

Alan Grafik objesinin Çevre Bilgisi Hesabı

Alan grafik objesinin çevre bilgisini sorgulamak için St_perimeter() fonksiyonu kullanılır.

st_perimeter(alang)

Örnek: 101 ada içerisindeki 1 numaralı parsel ait çevre bilgisi isteniyor.

```
select round(st_perimeter(parsel.parselg)::numeric, 2) as alan
from ada, parsel
where ada.ada_id = parsel.adaid
and ada.ada_no = 101
and parsel.parsel_no = 2;
```

Semt Açısı Hesabı

A ve B noktaları arasında A noktasından B noktasına olan semt açısının hesaplanması işlemi St_azimuth() fonksiyonu ile yapılır. St_azimuth() fonksiyonu içerisine semt açısının başlangıç noktası ile bitiş noktası olacak noktaları alır. Örneğin A noktasından B noktasına olan semt açısının hesaplanması isteniyorsa st_azimuth(a,b) şeklinde fonksiyon yazılır. St_azimuth() fonksiyonu sonuç değeri radyan olarak döndürür, sonucun grad açı biriminde görünmesi isteniyorsa fonksiyon 200 ile çarpılıp pi() fonksiyonuna bölünür. Pi() fonksiyonu postgresql veri tabanı yönetim sisteminde π sabit değeri için kullanılır. ifade 59 “nokta_tm” tablosu “nokta_ad” sahasında “P.2” adlı veri kaydı olan noktadan, aynı tablo içindeki “nokta_ad” sahasında “P.49” adlı veri kaydı olan noktaya olan semt açısı için gerekli SQL kodu bulunmaktadır.

ifade 59

```
select distinct st_azimuth(
(select geom_n from nokta_tm where nokta_ad = 'P.2'),
(select geom_n from nokta_tm where nokta_ad = 'P.49')
) * 200/pi() from nokta_tm;
```

Şekil 127 sorgu sonucudur.

Data Output		Messages	Notifications
	?column? double precision		
1	42.46178215270721		

Şekil 127

ifade 60 sorgusu bir önceki sorgunun ters semt açısıdır. Sorguda “P.49” adlı noktadan “P.2” adlı noktaya olan semt açısının hesabı bulunmaktadır.

ifade 60

```
select distinct st_azimuth(
(select geom_n from nokta_tm where nokta_ad = 'P.49'),
(select geom_n from nokta_tm where nokta_ad = 'P.2')
) * 200/pi() from nokta_tm;
```

Şekil 128 sorgunun sonucudur

Data Output		Messages	Notifications
	?column? double precision		
1	242.4617821527072		

Şekil 128

$$(P.49 - P.2) = (P.2 - P.49) + 200^g$$

Coğrafi Objeler Arasında Konumsal İlişkilerin Sorgulanması

Konu başlığı altında, coğrafi objeleri temsil eden grafik objelerin birbiriyle olan ilişkileri sorgulanacaktır. Birbirleriyle olan ilişkilerden kastedilen, objelerin teması, kesişimi, tamamen içinde kalması, birbirlerine belirli mesafede yakınlığı gibi konumsal ilişkilerdir. Bu ilişkilerin sorgulanması iki temel amaca dayanmaktadır. Birinci amaç, gerçekte olan konumsal ilişkidir. İkincisi ise, tersim (çizim) aşamasında yapılan yanlışın tespit edilmesidir.

Sorgulamaları yapmak için kullanılacak (postgis eklenti kütüphanesinde var olan) hazır fonksiyonlar geriye, doğru (TRUE ya da true) ya da yanlış (FALSE ya da false) olarak, mantıksal cevaplar döndürülecektir.

Objenin diğerini Tamamen Kapsaması (Tamamen İçinde olması)

Parsel, bina, kadastro veya imar adası, göl veya gölet gibi coğrafi objeler harita üzerinde alan (çokgen) grafik objesiyle temsil edilir. Objelerin CAD yazılımlarında tersim işlemi sırasında hata olabilir. Objeye çizimi sırasında gerekli olan obje yakalama modları kullanılmadığı için çizim sırasında komşu obje sınırları içine girerek çizim yapılabilir. Diğer bir durum ise coğrafi objenin bir diğer objenin tamamen içinde olup olmadığının sorgulanması istenebilir. Bu işlemi postgis eklentisinin hazır fonksiyonlarından *St_Contains()* ve *St_Within()* fonksiyonlarıyla yapabiliriz. Bu iki fonksiyonun içine iki parametre alır. Kapsayan obje ve içte kalan obje. Örneğin A objesi B objesini tamamen kapsıyor veya B objesi A objesini tamamen içinde kalıyor. Sorgulaması sırasında bunun doğru olup olmadığını bu fonksiyonlar ile sorgularken:

$$St_contains(A,B)$$

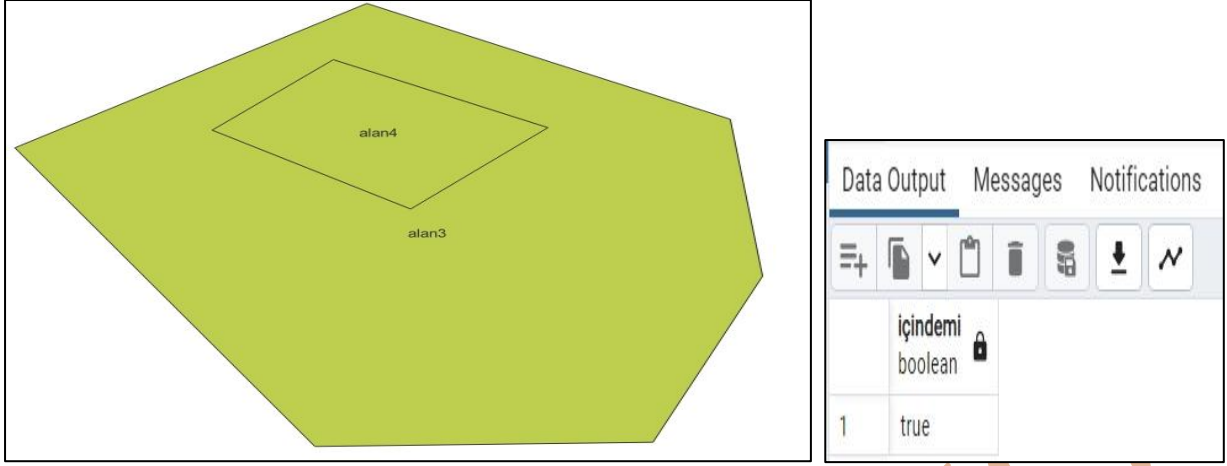
$$St_within(B,A)$$

Şeklinde sorgulayabiliriz.

ifade 61 “alan_tm” tablosu “alan_ad” sahasında *alan3* verisi olan kayıttaki grafik objenin, aynı tabloda “alan_ad” sahasında *alan4* verisi olan kayıttaki grafik objeyi kapsayıp kapsamadığının sorgulandığı sql cümlecığı bulunmaktadır. *Şekil 129* sol resim grafik objelerin birbirleriyle olan ilişkilerini göstermektedir. *Şekil 129* sağ resim ise, alan3 isimli objenin alan4 isimli objeyi kapsadığı sorgu sonucunun *true* (doğru) olarak döndürdüğü görülmektedir.

ifade 61

```
select distinct St_Contains(
(select geom_a from alan_tm where alan_ad = 'alan3'),
(select geom_a from alan_tm where alan_ad = 'alan4'))
from alan_tm;
```

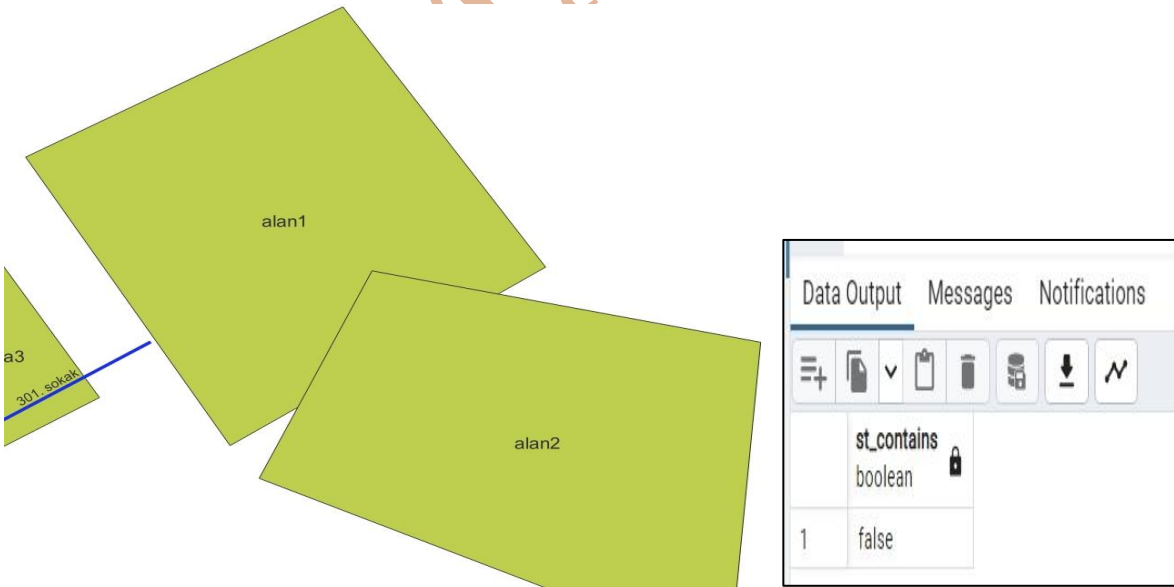


Şekil 129

ifade 62 “alan_tm” tablosu “alan_ad” sahasında *alan1* verisi olan kayıttaki grafik objenin, aynı tablo aynı sahada *alan2* verisi olan kayıttaki grafik objeyi kapsadığı sorgusuna ait sql cümlecği bulunmaktadır. Şekil 130 hem grafik objelerin haritadaki temsilini (sol resim) hem de sorgu sonucunu (sağ resim) göstermektedir. Objeler sadece kesişmektedir.

ifade 62

```
select distinct St_Contains(
(select geom_a from alan_tm where alan_ad = 'alan1'),
(select geom_a from alan_tm where alan_ad = 'alan2'))
from alan_tm;
```



Şekil 130

ifade 63, *ifade 64* ve *ifade 65* sorgu cümlecikleri, “alan_tm” tablosun “alan_ad” sahasında alan6 verisi olan kayıttaki alan grafik objesinin, “çizgi_tm” tablosun “çizgi_ad” sahasında çizgi2, çizgi3 ve çizgi4 verileri olan kayıtlardaki grafik objeleri kapsayıp kapsamadığı sorgularına ait sql cümlecikleri vardır.

Şekil 134 grafik objelerin haritadaki temsiline ait şekil gözükmemektedir. Her bir sorgu cümlecigiinden sonra dönen sonucu temsil eden resim bulunmaktadır. *Şekil 134* incelenirse, St_contains() fonksiyonu alan grafik objesinin kenarında çizilmiş olan ve çizgi3 verisinin ilişkili olduğu grafik obje gibi bir ucu alan objesinin kenarında olan objeyi kapsamadığı sonucu görülmektedir. Sadece çizgi4 verisiyle ilişkili çizgi grafik objesini tamamen kapsadığı görülmektedir.

ifade 63

```
select st_contains(alan_tm.geom_a,cizgi_tm.geom_c) as çizgi2içindemi
from alan_tm,cizgi_tm
where alan_tm.alan_ad = 'alan6' and
cizgi_tm.cizgi_ad = 'çizgi2';
```

Data Output	
	çizgi2içindemi boolean
1	false

Şekil 131

ifade 64

```
select st_contains(alan_tm.geom_a,cizgi_tm.geom_c) as çizgi3içindemi
from alan_tm,cizgi_tm
where alan_tm.alan_ad = 'alan6' and
cizgi_tm.cizgi_ad = 'çizgi3';
```

Data Output	
	çizgi3içindemi boolean
1	false

Şekil 132

ifade 65

```
select st_contains(alan_tm.geom_a,cizgi_tm.geom_c) as çizgi4içindemi
from alan_tm,cizgi_tm
where alan_tm.alan_ad = 'alan6' and
cizgi_tm.cizgi_ad = 'çizgi4';
```

Data Output		Messages	Notifications
	çizgi4içindemi boolean		
1	true		

Şekil 133

St_contains() fonksiyonu, çokgenin sadece kenarları yakalanarak çizilmiş (çizgi2, çizgi1) veya kenarına temas edilerek çokgenin içine doğru çizilmiş (çizgi3) objeler için *false* sonucunu döndürür, yani tamamen kapsamaz sonucunu döndürür. Fonksiyon sadece çokgenin içinde çizilmiş ve içinde kalan obje için *true* sonucunu döndürür.

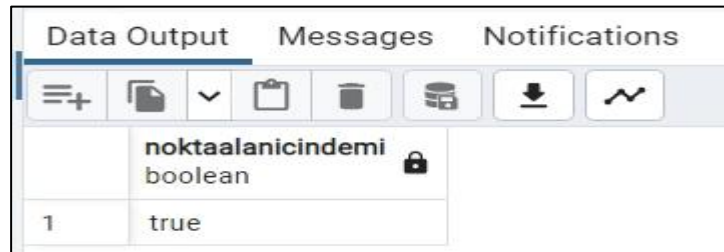


Şekil 134

ifade 66 ve ifade 67, Şekil 137'de görülen alan grafik objesiyle nokta grafik objelerinin konumsal ilişkisini sorgulamak için kullanılan sql cümlecikleridir. Sorgulama işleminde alan objesinin nokta objelerini kapsadığının sorgulanması için St_Contains() fonksiyonu kullanılmıştır. Sonuçlar Şekil 135 ve Şekil 136 içinde görülmektedir.

ifade 66

```
select st_contains(geom_a,geom_n) as noktaalanicindemi  
from nokta_tm, alan_tm  
where nokta_tm.nokta_ad = 'nokta8'  
and alan_tm.alan_ad = 'alan7';
```

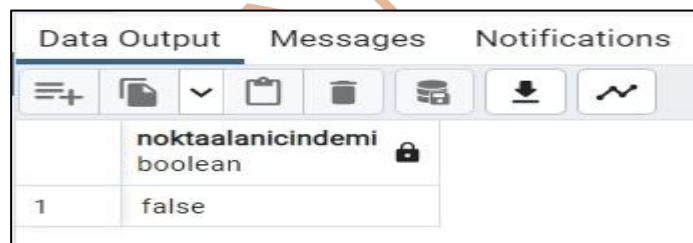


	noktaalanicindemi boolean
1	true

Şekil 135

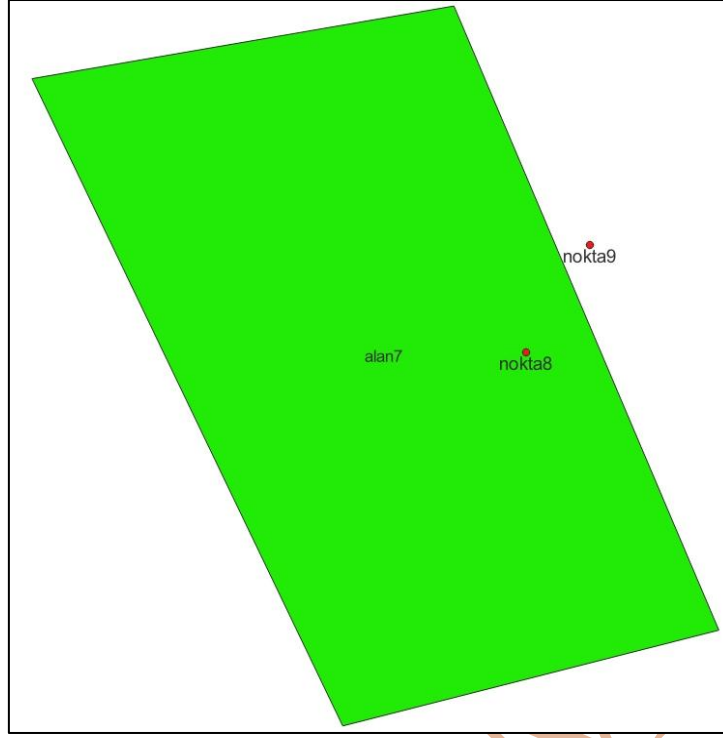
ifade 67

```
select st_contains(geom_a,geom_n) as noktaalanicindemi  
from nokta_tm, alan_tm  
where nokta_tm.nokta_ad = 'nokta9'  
and alan_tm.alan_ad = 'alan7';
```



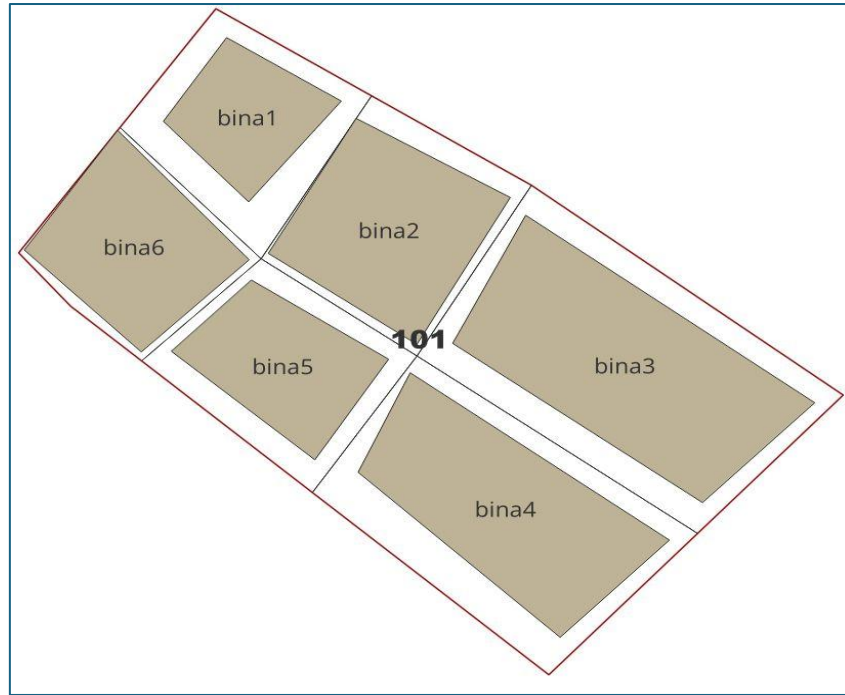
	noktaalanicindemi boolean
1	false

Şekil 136



Şekil 137

Örnek: Şekil 138 101 numaralı ada ve ada içindeki parsellerde bulunan binaların temsilidir. 101 numaralı ada sınırlarının tamamen içinde kalan ve tamamen içinde kalmayan binaların bulunması isteniyor.



Şekil 138

ada	bina
Columns (4)	Columns (4)
gid	gid
objectid	objectid
ada_no	bina_ad
geom	geom

Şekil 139

```

1 select bina.bina_ad
2 from ada,bina
3 where ada.ada_no=101
4 and st_contains(ada.geom,bina.geom);

```

	bina_ad character varying (80)
1	bina4
2	bina3
3	bina5
4	bina2
5	bina1

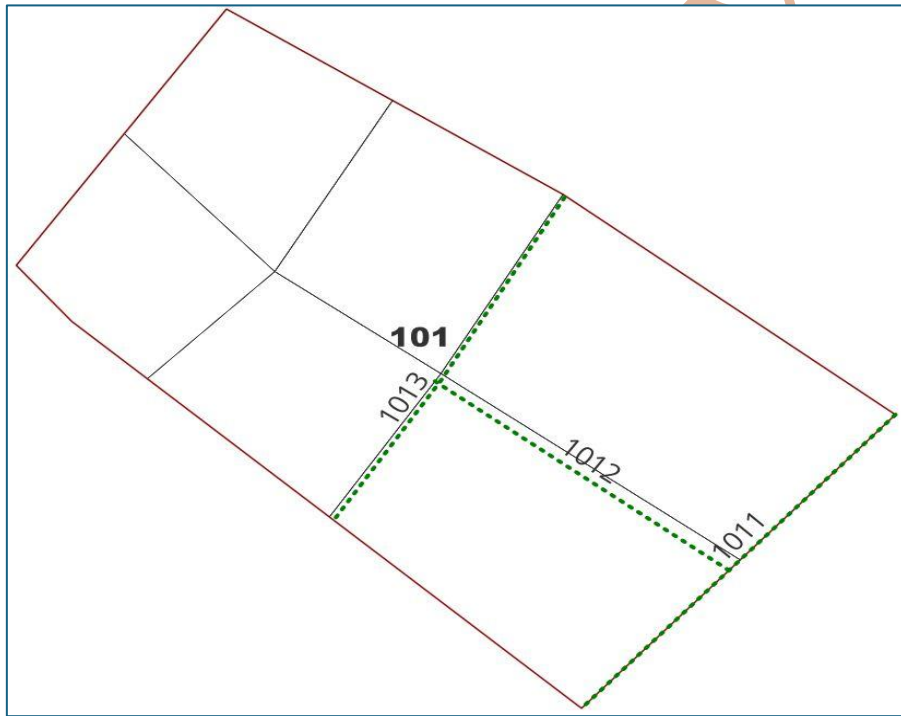

```

1 select bina.bina_ad
2 from ada,bina
3 where ada.ada_no=101
4 and st_contains(ada.geom,bina.geom)=false;

```

	bina_ad
	character varying (80)
1	bina6

Örnek: 101 numaralı imar adası içinde tamamen kalan enerji nakil hatlarının tespiti istenmektedir.



enerji_hat	
Columns (4)	
gid	
id	
nakil_no	
geom	

ada	
Columns (4)	
gid	
objectid	
ada_no	
geom	

```

1 select enerji_hat.nakil_no
2 from enerji_hat,ada
3 where ada.ada_no=101
4 and st_contains(ada.geom,enerji_hat.geom);

```

	nakil_no
1	1013

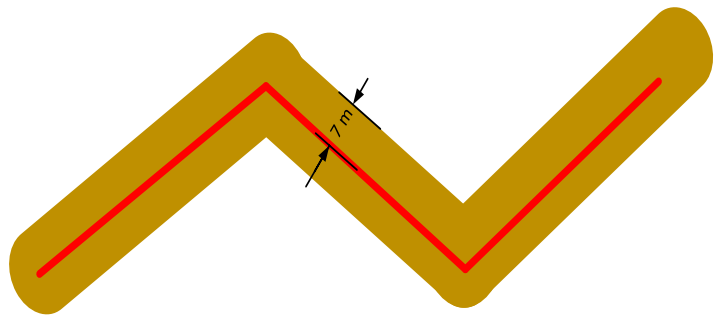
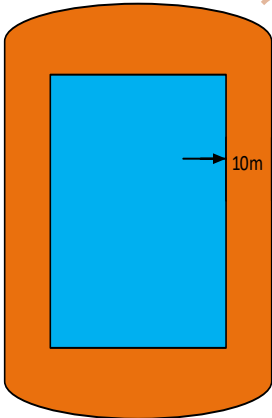
Objeye Belirli Mesafede İçinde Kalan Objelerin Sorgulanması

Coğrafik objenin etrafında kalan diğer bir coğrafik objenin bulunması coğrafi bilgi sistemleri uygulamalarında kullanılan önemli analizlerden biridir. Bu analizi yaparken A objesinin etrafında kalan B objelerini bulurken mesafe de belirtilebilir. Bu mesafe sayesinde sorgu “A objesinin etrafında 80 metre mesafede kalan B objelerinin bulunması” haline dönebilir.

Belirli bir mesafeyle içinde kalan objelerin bulunması için St_Dwithin() fonksiyonu kullanılır. Fonksiyon içine 3 parametre alır.

St_Dwithin(a_objesi, b_objesi, mesafe)

Yazılan fonksiyonda a_objesi'nin mesafe değeri kadar etrafında ve oluşturulan tampon bölge içinde kalan b_objesi'nin bulunması isteniyor.

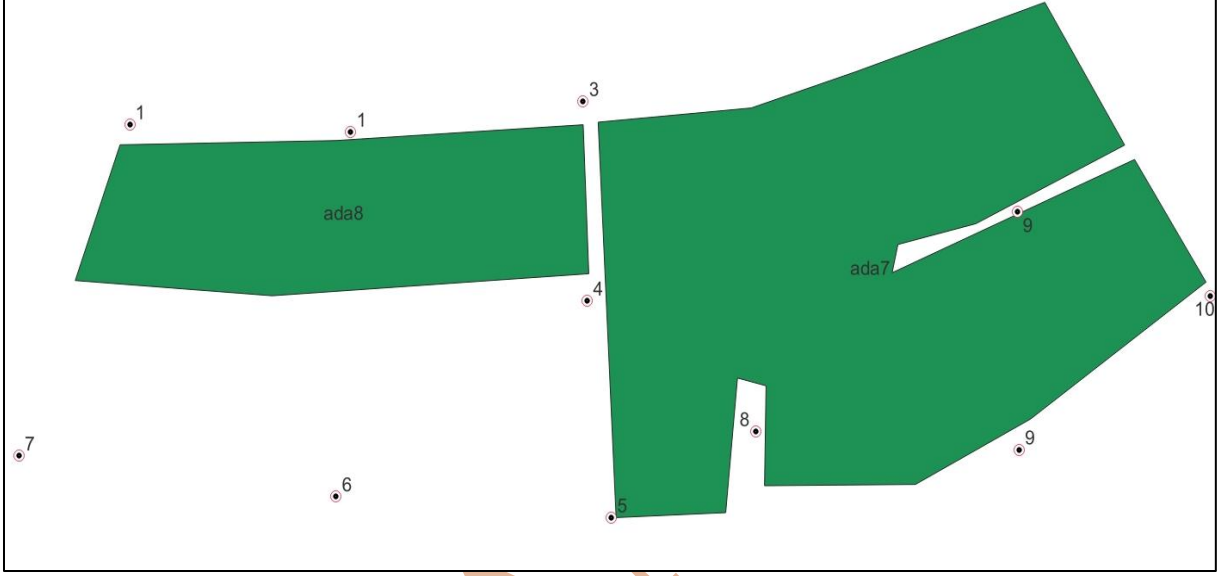


Şekil 140 imar adaları ve adaların etrafındaki elektrik direklerinin olduğu haritayı temsil etmektedir. ada8 isimli imar adasının 10 metre etrafında kalan elektrik direklerinin adları istenmektedir. ifade 68 istenilen sorgunun yazıldığı sql cümlecığıdir. Sorgu incelenirse, where

parametresinde `st_dwithin()` fonksiyonu kullanılmıştır. Fonksiyonda mesafe değeri belirtilirken ondalıklı hane değerleriyle yazılmıştır.

ifade 68

```
select elektrik_tm.nokta_ad
from elektrik_tm, alan_tm
where alan_tm.alan_ad = ' ada8'
and st_dwithin(elektrik_tm.geom_n, alan_tm.geom_a, 10.000);
```



Şekil 140

Şekil 141 sorgunun sonucunu göstermektedir.

Data Output		Messages	Notifications
	nokta_ad text		
1	1		
2	1		
3	3		
4	4		

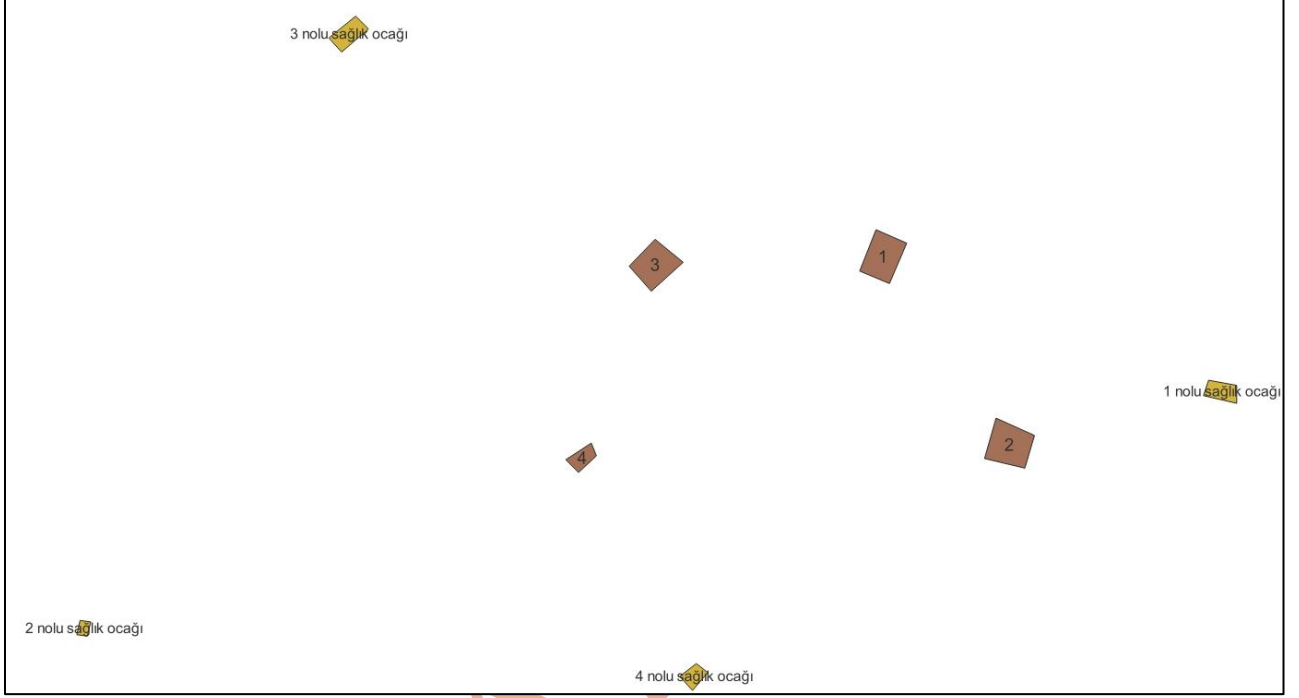
Şekil 141

Şekil 142 mesken tipindeki binaları ve binaların etrafında var olan sağlık ocaklarının olduğu haritayı temsil etmektedir. 1 numaralı binanın etrafında 300 metre veya daha yakında olan sağlık ocaklarının adlarının listelenmesi istenmektedir.

ifade 69 istenilen sorgu sonucunun sql cümlecığıdir.

ifade 69

```
select saglik_ocagi.ocak_adi  
from saglik_ocagi,mesken_tm  
where mesken_tm.bina_no = 1 and  
st_dwithin(saglik_ocagi.geom_s,mesken_tm.geom_b,300.000);
```



Şekil 142

Şekil 143 sorgu sonucunu göstermektedir.

	ocak_adi character varying (80)
1	1 nolu sađlık ocađı
2	4 nolu sađlık ocađı

Şekil 143

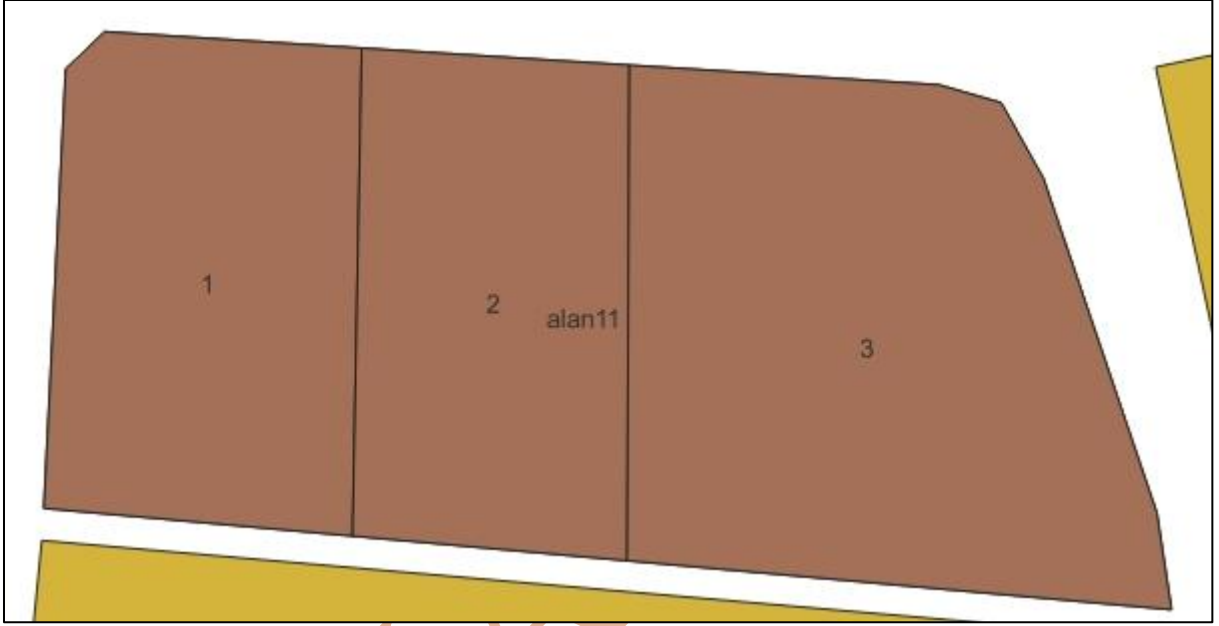


st_dwithin() fonksiyonunda eđer son parametre olan mesafe parametresi 0.0001 gibi milimetreden daha küçük bir hane girildiđi takdirde ierde kalan objenin tamamen kapsayan objenin iinde kalmasını aramaz. İerde kalan obje kapsayan objenin sınır izgilerinden itibaren izilmiş olması yeterli olacaktır.

Şekil 144 alan_tm tablosunda alan_ad bilgisi alan11 olan imar adası içinde 1, 2 ve 3 numaralı parsellere ait objeler gözükmemektedir. alan11 alan adlı ada içinde kalan imar parsellerini tespit etmek için illa kesişim sorgulaması yapmamız gerekmez.

ifade 70

```
select alan_tm.alan_ad, imar_parsel.parsel_no  
from alan_tm, imar_parsel  
where st_dwithin(imar_parsel.geom_a, alan_tm.geom_a, 0.0001)  
and alan_tm.alan_ad = 'alan11';
```



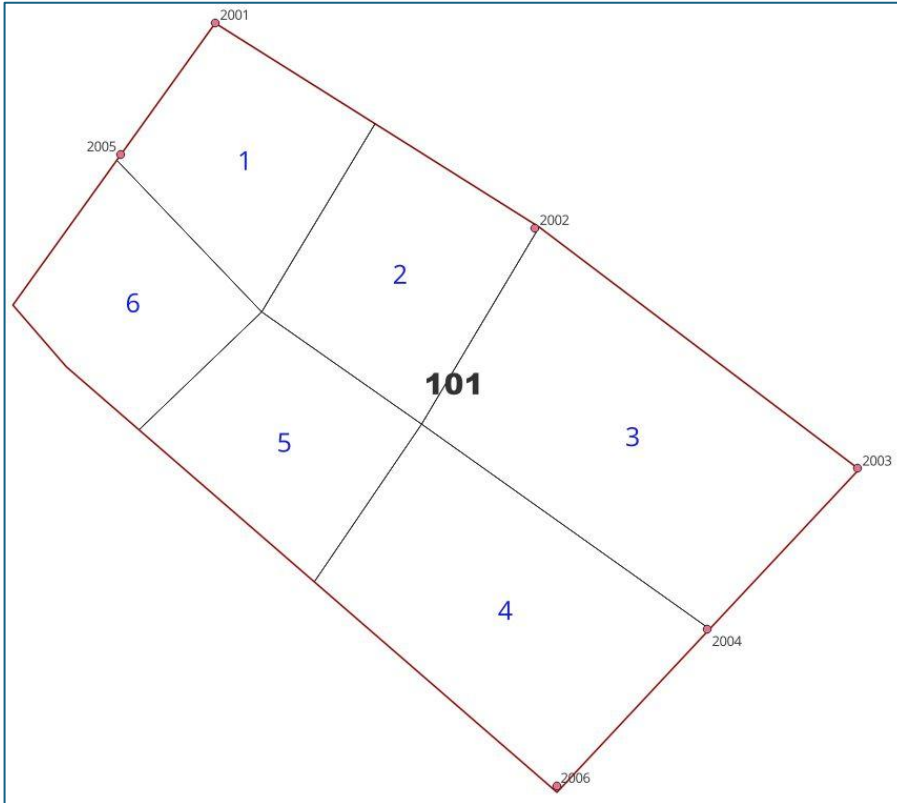
Şekil 144

	alan_ad	parsel_no
1	alan11	1
2	alan11	2
3	alan11	3

Şekil 145

Örnek: 2 numaralı parselin 50 metre yakınında kalan telefon direklerinin numaralarının listelenmesi (Şekil 146).

telefon_direk	parsel
Columns (4)	Columns (4)
gid	gid
id	objectid
direk_no	parsel_no
geom	geom



```

1 select telefon_direk.direk_no
2 from parsel,telefon_direk
3 where parsel.parsel_no=2
4 and st_dwithin(telefon_direk.geom,parsel.geom,50.000);

```

direk_no	
double precision	
1	2002
2	2004
3	2005
4	2001

Şekil 146

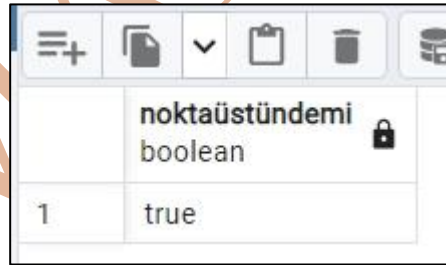
Nokta Grafik Objesinin Çizgi Grafik Objesi Üzerinde Olmasının Sorgulanması

St_Contains() fonksiyonu (veya St_Within()), alan objesinin kapsadığı alan grafik objesi, çizgi grafik objesi veya nokta grafik objesinin sorgulanmasında kullanılır. Fakat çizgi grafik objesinin üzerinde olan nokta grafik objelerini sorgulamak için St_Contains() fonksiyonu kullanılamaz. Çünkü çizginin tam üzerinde olan objeleri kapsama özelliği yoktur. Nokta grafik objesinin çizgi üzerinde olmasını sorgulamak için St_Covers() fonksiyonu kullanılır.

ifade 71 ve ifade 72, Şekil 149'de tasviri yapılmış olan “çizgi_tm” tablosu “çizgi_ad” sahasında *çizgi3* adlı verinin olduğu kaydın “geom_c” sahasındaki grafik obje kaydındaki, çizgi grafik objesi üzerindeki nokta objelerinin olup olmadığını sorgulayan sql cümlecikleridir. İşlemin yapılması için *St_Covers()* fonksiyonu kullanılmaktadır. Fonksiyon içindeki parametrelerin yazım sıralamasında önce çizgi sonra nokta objesi yazılmıştır. *Şekil 147 ve Şekil 148* sorgu sonuçları gözükmemektedir.

ifade 71

```
select st_covers(geom_c, geom_n) as noktaüstündemi
from çizgi_tm, nokta_tm
where nokta_tm.nokta_ad = 'nokta1'
and çizgi_tm.cizgi_ad = 'çizgi3';
```



	noktaüstündemi
1	true

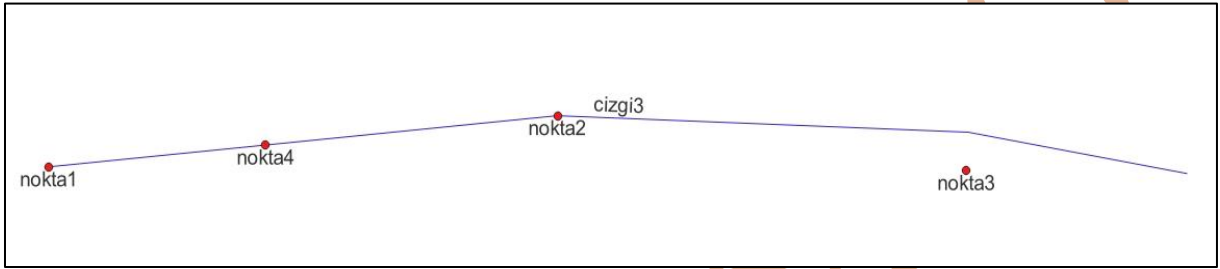
Şekil 147

ifade 72

```
select st_covers(geom_c, geom_n) as noktaüstündemi
from çizgi_tm, nokta_tm
where nokta_tm.nokta_ad = 'nokta3'
and çizgi_tm.cizgi_ad = 'çizgi3';
```

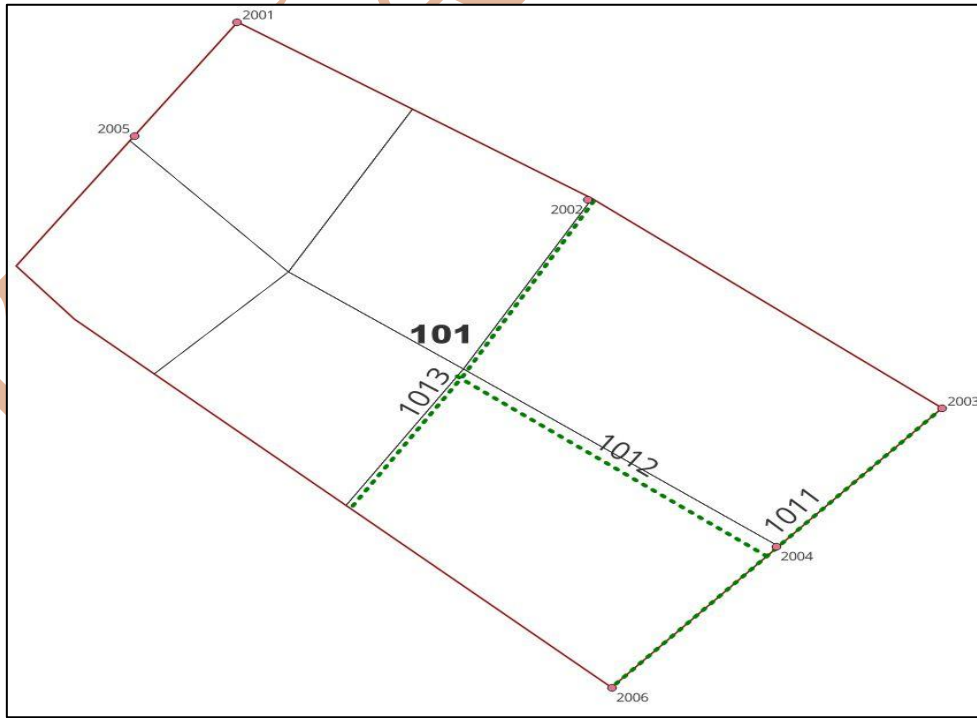

Data Output		Messages	Notifications
noktaüstündemi			
boolean			
1	false		

Şekil 148



Şekil 149

Örnek: 1011 numaralı enerji nakil hattı üzerindeki telefon direklerinin direk numaralarının listelenmesi.



telefon_direk	enerji_hat
Columns (4)	Columns (4)
gid	gid
id	id
direk_no	nakil_no
geom	geom

```

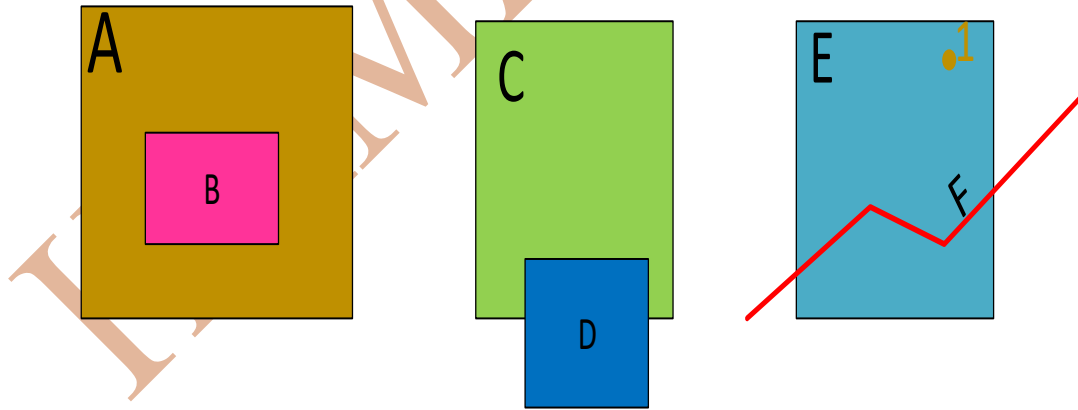
1 select telefon_direk.direk_no
2 from telefon_direk, enerji_hat
3 where enerji_hat.nakil_no=1011
4 and st_covers(enerji_hat.geom,telefon_direk.geom);

```

	direk_no
	double precision
1	2003
2	2006

Kesişen Grafik Objelerin Tespiti

Grafik objelerin kesişmesi, sadece obje kesişimini içermez, aynı zamanda bir önceki konu olan kapsamayı da içerir. Şekil 150 incelendiğinde A çokgeni B çokgenini kapsar aynı zamanda A ile B kesişir. C ile D çokgenleri kesişen iki objedir. E çokgeni ile F çokgeni doğru objesi kesişir. E çokgeni 1 nokta objesini kapsar aynı zamanda E ile 1 kesişir.



Şekil 150

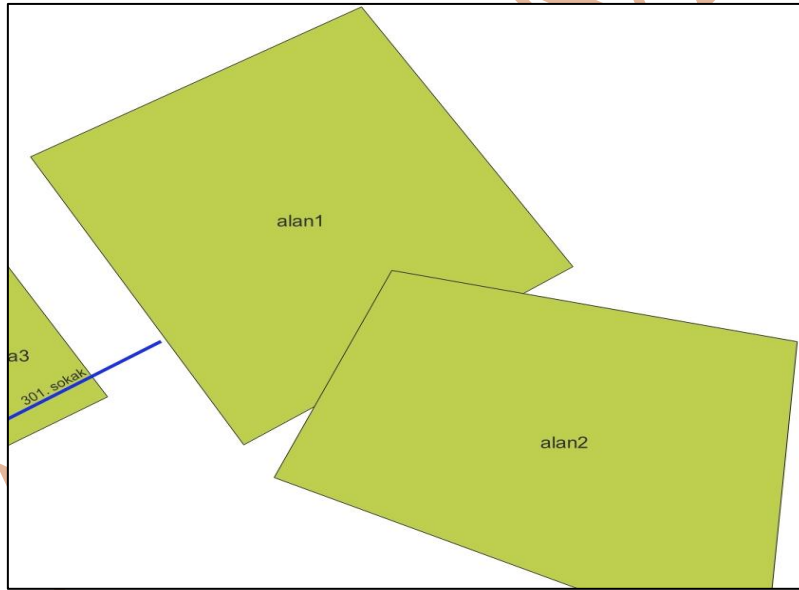
Grafik objelerin kesişimlerini kontrol etmek için St_Intersects() fonksiyonu kullanılır. Daha önce sorgulanan iki objenin kesişimlerini sorgulayalım. Şekil 152 daha önce objenin bir diğerini kapsamasını için sorgulanmıştı. ifade 73 kesişim sorgulamasına ait sql cümlecığıdir. Şekil 151 sorgu sonucunu göstermektedir. Sorgu sonucu true kesişimin olduğunu söyler.

ifade 73

```
select distinct st_intersects(
  (select geom_a from alan_tm where alan_ad = 'alan1'),
  (select geom_a from alan_tm where alan_ad = 'alan2'))
from alan_tm;
```

Data Output		Messages	Notifications
<div style="display: flex; justify-content: space-between; align-items: center;"> st_intersects boolean 🔒 </div>			
1	true		

Şekil 151



Şekil 152

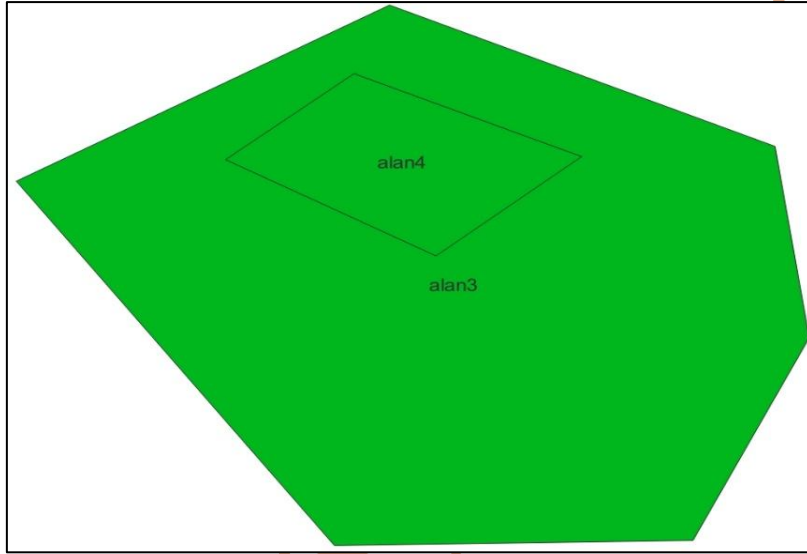
Şekil 154 alan4 isimli çokgen objenin alan3 isimli çokgen obje ile kesişimi sorgulanmaktadır. Bu objeler daha önce birbirlerini kapsayıp kapsamadıkları sorgulanmıştı. ifade 74 kesişim sorgusunun sorgu cümlecğini içerir. Şekil 153 yapılan sorgu sonucu st_intersects() fonksiyonun seçimini geri döndürür.

ifade 74

```
select distinct st_intersects(
  (select geom_a from alan_tm where alan_ad = 'alan3'),
  (select geom_a from alan_tm where alan_ad = 'alan4')) as alan3alan4kesişim
```

Data Output		Messages	Notifications
	alan3alan4kesişim boolean		
1	true		

Şekil 153



Şekil 154

Örnek: 1013 numaralı enerji nakil hattının kestiği parsellerin parsel numaralarının listesi istenmektedir.

parsel	
Columns (4)	
gid	
objectid	
parsel_no	
geom	

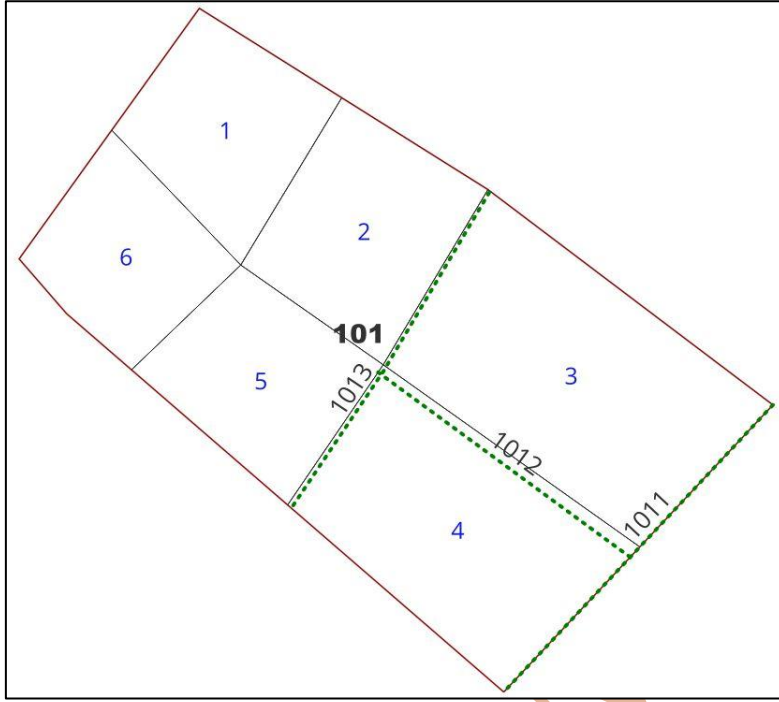
enerji_hat	
Columns (4)	
gid	
id	
nakil_no	
geom	

```

1 select parsel.parsel_no
2 from parsel,enerji_hat
3 where enerji_hat.nakil_no=1013
4 and st_intersects(enerji_hat.geom,parsel.geom)

```

parsel_no		
double precision		
1	4	
2	3	



İki Objenin Geometrik İlişkilerinin Olup olmadıklarını Sorgulanması

Tek bir tabaka içindeki (veya iki ayrı tabakanın grafik objeleri arasındaki) grafik objelerin aralarında grafik ilişkilerinin olduğu ya da olmadığını sorgulamak için `St_Disjoint()` fonksiyonu kullanılır. Fonksiyon içerisine iki adet *geometry* veri tipinde parametre alır.

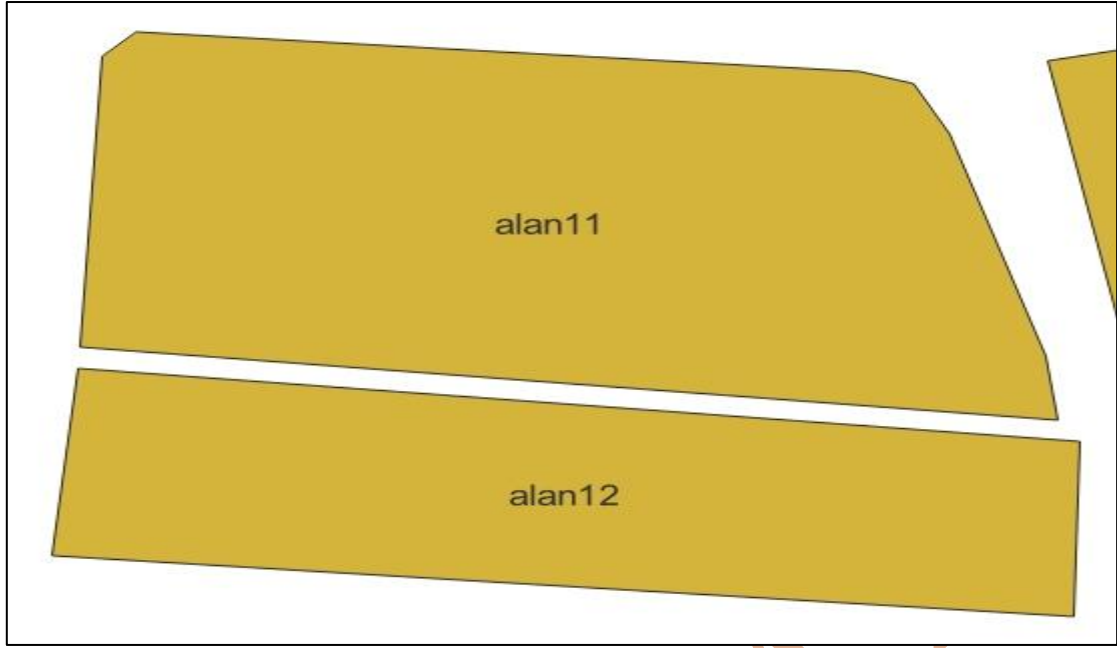
St_disjoint(geom1, geom2)

Eğer objeleri arasında bir geometrik ilişki yoksa (yeni kesişim yoksa, biri diğerinin içinde değilse, bir noktada birbirlerine dokunmuyorsa) *true* (doğru) cevabı geri döner. Eğer fonksiyon içindeki iki objenin birbirleriyle ilişkileri varsa fonksiyon geriye *false* (yanlış) cevabı döndürür.

Şekil 155 alan11 ve alan12 adlı, alan_tm tabakasındaki iki grafik objedir. Bu iki grafik obje arasında hiçbir geometrik ilişki yoktur. ifade 75 bu iki grafik objenin geometrik ilişkisinin olup olmadığını `st_disjoint()` fonksiyonu ile yapılan sorgunun sql cümlecığıdir. Şekil 156 sorgu sonucunun *true* olduğu görülmektedir.

ifade 75

```
select distinct st_disjoint((select geom_a from alan_tm where alan_ad = 'alan11'),
(select geom_a from alan_tm where alan_ad = 'alan12'))
from alan_tm;
```



Şekil 155

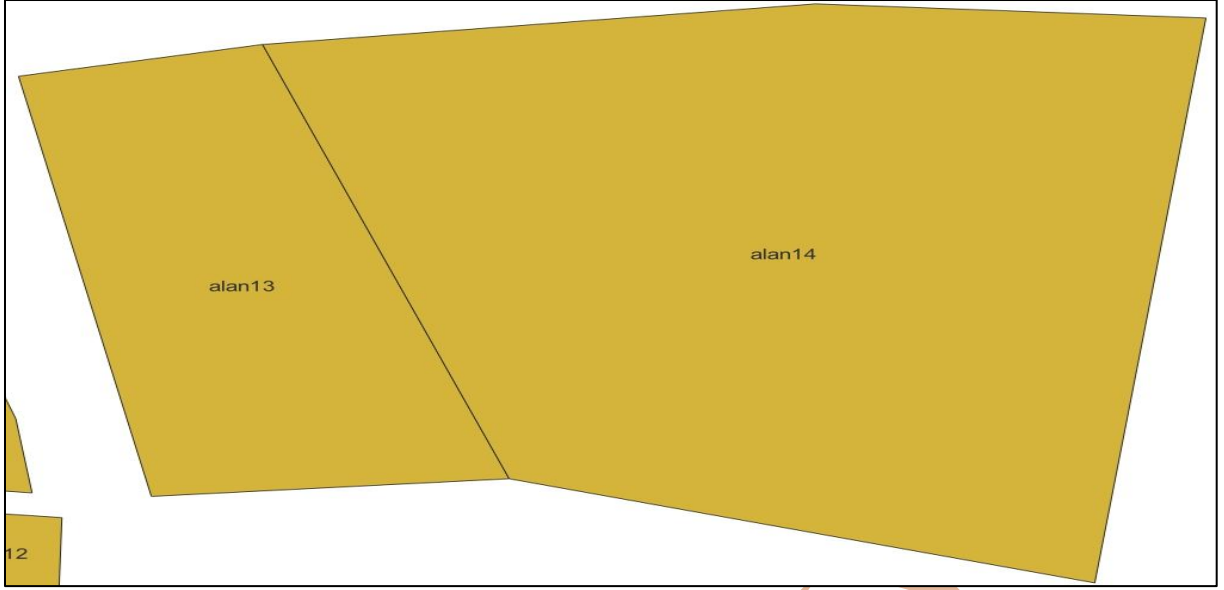
Data Output		Messages	Notifications
	st_disjoint boolean		
1	true		

Şekil 156

Şekil 157 alan_tm tablosundaki alan13 ve alan14 grafik objelerinin haritadaki temsildir. İki grafik obje çizimi yapılırken ortak iki kırılma noktası (veya ortak çizgi) kullanılmıştır. Aralarında sınır komşuluğu vardır yani aralarında grafik ilişki vardır. ifade 76 iki grafik obje arasındaki geometrik ilişkisinin olup olmadığına dair sorgu cümlecığıdir. Sorgu sonucu Şekil 158'da görülmektedir. Geriye false sonucu dönmüştür.

ifade 76

```
select st_disjoint((select geom_a from alan_tm where alan_ad = 'alan13'),
(select geom_a from alan_tm where alan_ad = 'alan14'))
```



Şekil 157

Data Output	Messages	Notifications
<div style="display: flex; justify-content: space-between;"> ☰+ 📄 ▼ 📋 🗑️ 🗄️ 📥 📈 </div>		
	st_disjoint boolean	🔒
1	false	

Şekil 158

Aynı Tablo İçindeki Grafik Objelerin Grafik İlişkilerinin Tespiti

Aynı tablo içinde var olan grafik objelerin birbirleriyle olan ilişkilerinin sorgulanması için tablo adını iki farklı değişkene aktarmamız gerekli. Bu işlemi *from* parametresinde yapacağız. ifade 77 tablo adının bir değişkene aktarılması işlemine bir örnektir.

ifade 77

from tablo_ad a veya *from* tablo_ad as a

from tablo_ad a, tablo_ad b veya *from* tablo_ad as a, tablo_ad as b

Tablo içindeki her bir objeyi tek tek birbirleriyle karşılaştırabilmek için id (kimlik) sahaları kullanılmalıdır. Kimlik sahaları genelde her bir kayıt için tekil ve tam sayı değeriyle temsil edilecek şekilde oluşturulur. Bu sahaların sırasıyla bir sonraki kayıttan bir önceki kayıt ile karşılaştırılması için *where* parametresinde

$a.id < b.id$ or $a.id > b.id$ or $a.id \neq b.id$

Kısıtlaması yapılır.

Sadece birbiri içinde kalan alan objelerinin bulunması ve içerenin adı ile içinde kalanın adı olacak şekilde listelenmesi (ifade 78):

ifade 78

```
select a.alan_ad,b.alan_ad
from alan_tm a,alan_tm b
where ST_contains(a.geom_a,b.geom_a)
AND a.id < b.id
```

Birbirleriyle kesişen objelerin bulunması: (ifade 79)

ifade 79

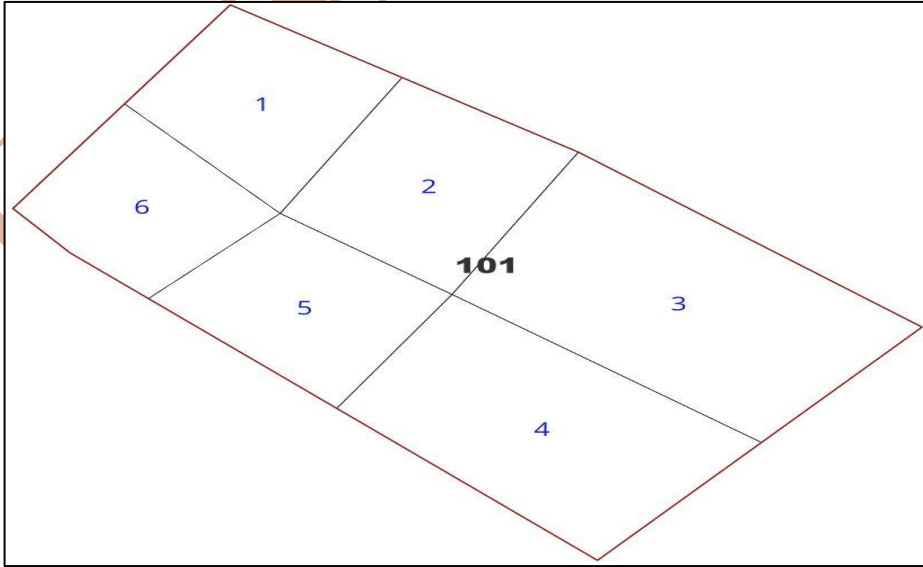
```
select a.alan_ad,b.alan_ad
from alan_tm a,alan_tm b
where ST_Intersects(a.geom_a,b.geom_a)
AND a.id < b.id
```

Birbirleriyle kesişen çizgilerin tespiti (Kesişen yol objelerinin tespiti): (ifade 80)

ifade 80

```
select a.cizgi_ad,b.cizgi_ad
from cizgi_tm as a,cizgi_tm as b
where st_crosses(a.geom_c,b.geom_c)
and st_intersects(a.geom_c,b.geom_c) and a.id < b.id;
```

Örnek: 3 numaralı parselde komşu olan parsellerin parsel numaralarının listelenmesi istenmektedir.



▼	📄	parsel
▼	📄	Columns (4)
	📄	gid
	📄	objectid
	📄	parsel_no
	📄	geom

```

1 select b.parsel_no
2 from parsel a, parsel b
3 where a.parsel_no=3
4 and st_intersects(a.geom,b.geom)
5 and a.gid!=b.gid;

```

	parsel_no	
	double precision	🔒
1		4
2		5
3		2

Birbirini kapsayan iki objeden birindeki saha verisi ile diğerinin sahasının güncellenmesi

“mezarlik” tablosundaki alan objesinin içinde kalan, “ada” tablosundaki alan objelerinin *kimlik_mez* sahasının “mezarlik” tablosundaki *kimlik_mez* sahasıyla doldurulması.

ifade 81

```

update ada
set kimlik_mez = mezarlik.kimlik_mez
from mezarlik
where st_dwithin(ada.geom_ada, mezarlik.geom_mez, 0.00001);

```

Tablodaki Grafik Objeyi Güncelleme (Düzenleme)

```
update   nokta2   set   geom=St_geomfromtext('POINT(33.697203725661474
39.35609534526361)',4326) where nokta_ad='KamanMyo';
```

PostgreSQL Veri Tabanı Yönetim Sistemi İçinde PL/pgSQL Dili ile Program Oluşturma

PostgreSQL programlama dili içinde PL/pgSQL procedural language (PL – yöntemsel dil) programlama dili kullanılır. Bu dil içinde sınaama yapıları (if - then), döngüler (for - loop), SQL cümlecikleri kullanılabilir, değişken tanımları yapılabilir. Programlama dili anlatımı yapılırken (PostgreSQLTutorial, 2023) internet kaynağından yararlanılmıştır. İş bu doküman içinde detaylı bir programlama anlatımından ziyade PostgreSQL veri tabanı yönetim sisteminde oluşturulmuş bir veri tabanı için kullanıcı tanımlı tetikleyici mekanizmasıyla çalışacak bir fonksiyon oluşturulması için gerekli gelecek programlama bilgiler üzerinde durulmuştur.

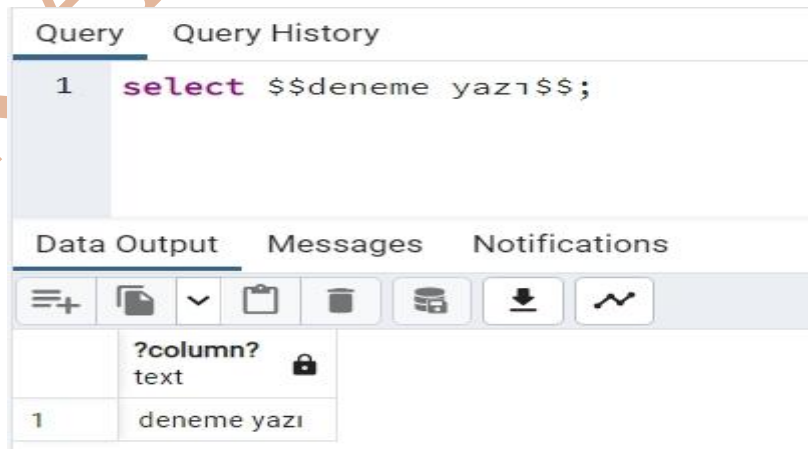
PL/pgSQL Dili ile Programlamaya Giriş

Başlık altında programlama için gereken yazım dilinde kullanılan ifadeler, sınaama yapıları, döngü yapıları anlatılacaktır.

\$\$ Blok Yapıları

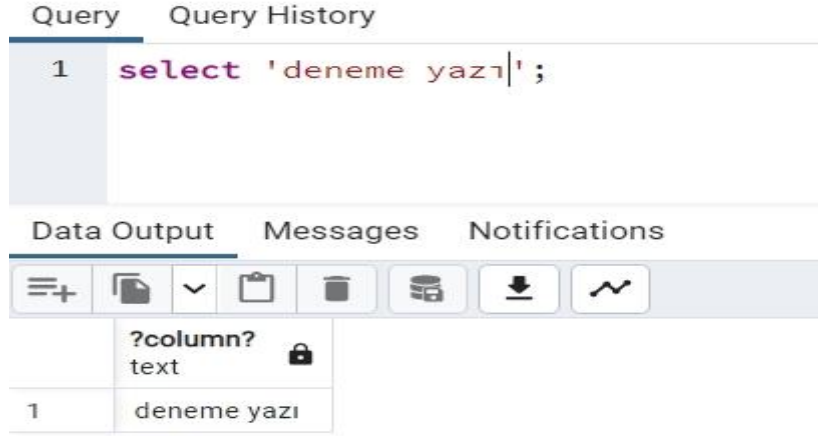
Blok ifadesinden kastedilen \$\$ ile başlayıp \$\$ ile biten kısımdan oluşan blok yapısıdır. Bu yapı arasında kalan kısımda program kodu yazılır. \$\$ bloğu içinde metin yazı yazıldığını belirtmiş oluyoruz.

Şekil 159 \$\$ kullanım örneği vardır. Örnekte program kodu oluşturulmamış, \$\$ kod bloğunun kullanımını amaçlanmıştır. \$\$ bloğu içindeki ifadeyi bir hata olarak görmüyor. Blok içindeki yapıyı bir metin yazısı olarak görüyor.



Şekil 159

\$\$ bloğu yerine ' bloğu kullanılabilir (Şekil 160).



Şekil 160

PostgreSQL veri tabanı yönetim sisteminde SQL kullanılarak tablo ekleme, tablo içindeki verilerin saha oluşturma – veri ekleme – veri güncelleme – kayıt silme gibi işlemler yapılabilir. SQL kullanılarak veri tabanı yönetim işlemleri dışında yapılacak olanları PostgreSQL programına anlatılması için metin blokları kullanılmalıdır. PL/pgSQL programlama dilinde hem program kodunun okunabilirliğini arttırmak için hem de kullanıcıya uyarı mesajları verilmesi için \$\$ bloğu kullanımı yapılarak program algoritması yazılacaktır.

PL/pgSQL Programlama Dilinde Veri Tipleri

Kod bloğu içinde alınan sonuçları tutması veya veri ataması yapmak için değişken tanımları yapılır. Değişkenler belirli tipteki verileri tutmak için tanımlanır. Bu bağlamda tutulacak veriye göre, PL/pgSQL kullanımında var olan, veri tipleri bilinmelidir. PL/pgSQL kullanımında veri tipleri PostgreSQL veri tabanı yönetim sistemi için belirlenmiş veri tipleriyle aynıdır.

Veri Tipi	Veri Tipi İçeriği
integer	Tam sayı verileri tutmak için kullanılır
text veya varchar()	Hem numerik hem de alfabetik veri içeren metin ifadeler için kullanılır.
Numeric(p,s)	Ondalıklı sayıları tutmak için kullanılır. p harfi, ondalık ve tam kısmın toplam hane sayısını belirtir. s ondalık hane sayısını belirtir.
double precision	Ondalıklı sayıları (reel sayı) kayıt altına almak için kullanılır. Yaklaşık 15 haneli sayıları tutmak için kullanılabilir.
real	Ondalıklı sayıları (reel sayı) kayıt altına almak için kullanılır. Yaklaşık 6 haneli sayıları tutmak için kullanılabilir.
date	Tarih verisini kayıt altına almak için kullanılır.

timestamp	Zaman verisini kayıt altına almak için kullanılır.
boolean	Mantıksal ifadeleri (doğru/yanlış) kayıt altına almak için kullanılır.

Değişken tanımları *declare* bloğu içinde yapılır. Declare bloğu içinde değişkenlere değer ataması yapmak için := ifadesi kullanılır.

Örnek:

```
do$$
declare
sayi1 integer;
sayi2 double precision;
sayi3 integer := 0;
begin
    sayi2 := 12.46;
end$$;
```

PL/pgSQL Program Kod Bloğu Oluşturulması

PL/pgSQL blok yapıda programlama dilidir (postgresql tutorial, 2023). \$\$ blok yapısı içinde kalan kısımda program kodu oluşturulacak. Program kodu içinde kullanılacak değişkenler ayrı bir blok yapısı içinde bildirilecek, program kod kısmı ayrı bir blok yapısı içinde bildirilecek. ifade 82 örnek blok yapısıdır. Başlangıç ve bitişte \$\$ bloğu gözükmektedir. *declare* ile başlayan blok değişkenlerin bildirildiği bloktur. Her bir değişken tanım satırı ; ile biter. Program kodunun yazıldığı blok *begin* ile başlayıp *end* ile biter. Dikkat edilirse *end* ifadesi ; ile bitmektedir.

ifade 82

```
do
$$
declare
-- -- değişkenlerin tanımlandığı (bildirildiği) blok
begin
-- -- program kodunun yazıldığı blok
end $$;
```

Örnek: Ada tablosu içindeki kayıt sayısını bir mesaj ile sonucu yazan program kodunu PL/pgSQL program dili ile oluşturunuz.

Cevap: ifade 83 istenilen cevaba ait PL/pgSQL sonucudur. Kod incelendiğinde ilk ifade do padmin editöründe program kodunun çalışmasını sağlayacak ifadedir. *Declare* bloğunda **ada_sayisi** adında *integer* veri tipinde bir değişken tanımlanmış ve değişkenin ilk değeri olarak da 0 değeri aktarılmıştır. := yerine direkt = operatörü de kullanılabilir. Değişkene değer aktarımı yapılırken := ifadesi kullanılmasına dikkat edilmelidir. *begin – end* bloğu arasında ada tablosu içindeki kayıt sayısı *count()* hazır fonksiyonu ile bulunmuş, sonuç **ada_sayisi** değişkenine aktarılırken *into* ifadesi kullanılmıştır. *Raise notice* ifadeleri sonucu bir mesaj olarak gösterilmesinde kullanılmıştır. Sonuç mesaj '' aralığında belirtilmiştir. Mesaj içinde değişkenin kullanılması için % parametresi kullanılmıştır. *Şekil 161* program kodunun padmin editöründe çalıştırılması ve sonuç çıktının oluşturulması örneğidir.

ifade 83

```
do $$
  declare
    ada_sayisi integer := 0;
  begin
    select count(*) into ada_sayisi from ada;
    raise notice 'Ada tablosu içindeki ada sayısı = %', ada_sayisi;
  end $$;
```



Query	Query History
1	do \$\$
2	declare
3	ada_sayisi integer :=0;
4	begin
5	select count(*) into ada_sayisi from ada;
6	raise notice 'Ada tablosu içindeki ada sayısı=%',ada_sayisi;
7	end \$\$;
8	

Data Output	Messages	Notifications
	NOTICE: Ada tablosu içindeki ada sayısı=3	DO
Query returned successfully in 43 msec.		

Şekil 161

Örnek: Öğrencinin adı, soyadı ve sınavdan aldığı not değerinin değişkenlere aktarıldığı ve ekrana yazıldığı PL/pgSQL programlama dili kodunu yazınız.

Cevap: Şekil 162 cevabın program kodunun ve sonucunun olduğu resimdir. 7. ve 8. Satırlara dikkat edilirse, mesaj kısmı yazılırken satırın bittiği ; operatörü ile bittiği belirtilmiştir. 7 ve 8. Satırlar tek bir satır gibi işlem görmüştür.

Query	Query History
1	do \$\$
2	declare
3	ogrenci_ad varchar(20) = 'Emre';
4	ogrenci_soyad varchar(50)= 'İNCE';
5	ogrenci_notu numeric(5,2)=78.65;
6	begin
7	raise notice '% ad % soyadlı öğrencinin notu= %',
8	ogrenci_ad, ogrenci_soyad, ogrenci_notu;
9	end \$\$;

Data Output	Messages	Notifications
	NOTICE: Emre ad İNCE soyadlı öğrencinin notu= 78.65	

Şekil 162

Örnek: Her bir mezarlık için kimlik numarası (id) belirlenmiştir. 101 kimlik numarasına sahip mezarlığın ad bilgisini mesaj olarak yazılmasını sağlayan PL/pgSQL program kodunu yazınız.

Cevap:

Şekil 163 istenilen cevabın pgsadmin editöründe oluşturulması ve sonucudur. 3 ve 4. Satırlarda daha önce kullanılmayan *%type* ifadesi bulunmaktadır. Dikkat edilirse *%type* ifadesi, değişken tanımında değişken veri tipinde kullanılmıştır. *%type* ifadesi, değişkenin veri tipinin veri aktarımında kaynak olacak tablo içindeki sahanın veri tipi ile aynı olması için kullanılmıştır. ifade 84 incelendiğinde, *%type* ifadesiyle **mezarlik** adlı tablonun **mezarlik_ad** sahasının veri tipinin aynısı **mez_adi** adlı değişken içinde uygulanması sağlanmıştır.

ifade 84

mez_adi mezarlik.mezarlik_ad%type;

%type ifadesinin iki kullanım gerekçesi vardır:

- İlk gerekçe, tablonun içindeki sahanın veri tipi bilinmek zorunda değildir,
- İkinci gerekçe, tablonun içindeki sahanın veri tipi daha sonra değiştirilebilir.

4. satırda **mez_adi2** isimli farklı bir değişken tanımlanmıştır. Bu değişkenin veri tipi **mez_adi** adlı değişkenin veri tipi ile aynı olması sağlanmıştır.

Query	Query History	
1	<code>do \$\$</code>	
2	<code> declare</code>	
3	<code> mez_adi mezarlik.mezarlik_ad%type;</code>	
4	<code> mez_adi2 mez_adi%type;</code>	
5	<code> begin</code>	
6	<code> select mezarlik_ad into mez_adi from mezarlik where kimlik_mez=101;</code>	
7	<code> raise notice '101 kimlik numaralı mezarlığın adı= %',mez_adi;</code>	
8	<code> end \$\$;</code>	
Data Output	Messages	Notifications
	NOTICE: 101 kimlik numaralı mezarlığın adı= Hacıpınar Mahallesi Mezarlığı	
	DO	

Şekil 163

`%type` ifadesi ile belirtilen sahanın veri tipini elde etmiş olduk. Eğer sorgu sonucunda tek bir sahanın değil tüm sahaların verilerini bir değişkene aktarmak istersek `%rowtype` ifadesi kullanılır.

Örnek: Ada tablosu içinde 101102 kimlik numaralı adanın tüm bilgilerinin listelenmesi istenmektedir. İstenilene verecek PL/pgSQL dili ile yazılmış programı oluşturun.

Cevap:

Şekil 164 istenilen soruya ait cevabın pgsadmin editöründe hazırlanmış program kodu ve cevabına ait şekildir. `declare` bloğunda `ada_tum_sahalar` adlı değişken tanımlanırken tablonun adı ile `%rowtype` ifadesi `ada%rowtype` şeklinde kullanılmıştır. Bu sayede `ada_tum_sahalar` ismindeki değişken içine, seçim sonucu ulaşılan satıra ait kayıttın tümü aktarılmış olur. Şekil 164 satır 10'da sonuç saha değerlerine ulaşım metodu gösterimi vardır.

Query	Query History	
1	<code>do</code>	
2	<code> \$\$</code>	
3	<code> declare</code>	
4	<code> ada_tum_sahalar ada%rowtype;</code>	
5	<code> begin</code>	
6	<code> select * into ada_tum_sahalar</code>	
7	<code> from ada</code>	
8	<code> where kimlik_ada=101102;</code>	
9	<code> raise notice '101102 numaralı adanın ada numarası %, içinde bulunduğu mezarlık kimlik numarası %',</code>	
10	<code> ada_tum_sahalar.ada_no,ada_tum_sahalar.kimlik_mez;</code>	
11	<code> end \$\$;</code>	
Data Output	Messages	Notifications
	NOTICE: 101102 numaralı adanın ada numarası 102, içinde bulunduğu mezarlık kimlik numarası 101	

Şekil 164

%rowtype ifadesinin benzeri *record* ifadesidir. *Şekil 165* record ifadesi kullanım örneği bulunmaktadır.

Query	Query History
1	do
2	\$\$
3	declare
4	ada_tum_sahalar record;
5	begin
6	select * into ada_tum_sahalar
7	from ada
8	where kimlik_ada=101102;
9	raise notice '101102 kimlik numaralı adanın ada numarası= %, içinde bulunduğu mezarlık kimlik numarası=%',
10	ada_tum_sahalar.ada_no,ada_tum_sahalar.kimlik_mez;
11	end \$\$;

Data Output	Messages	Notifications
NOTICE: 101102 kimlik numaralı adanın ada numarası= 102, içinde bulunduğu mezarlık kimlik numarası=101		
DO		

Şekil 165

Seçim işlemi sonucu birden fazla kayıt dönebilir. Tüm kayıtlar sırasıyla yine tek bir değişkene aktarılabilir. Seçim sonucu eğer birden fazla kayıt dönüyor ve her bir satıra ayrı ayrı ulaşılmak isteniyorsa yine *record* ifadesi kullanılabilir. *Şekil 166* belirtilene dair bir kod örneğidir. İşlemin yapılabilmesi için for – end loop döngüsü kullanılmıştır. Döngü yapıları daha sonra kendi konu başlığı altında anlatılacaktır.

Query	Query History
1	do
2	\$\$
3	declare
4	ada_secim record;
5	begin
6	for ada_secim in select * from ada loop
7	raise notice 'ada numarası= %, ada kimlik numarası= %, mezarlık kimlik numarası= %',
8	ada_secim.ada_no,ada_secim.kimlik_ada,ada_secim.kimlik_mez;
9	end loop;
10	end \$\$;

Data Output	Messages	Notifications
NOTICE: ada numarası= 101, ada kimlik numarası= 101101, mezarlık kimlik numarası= 101		
NOTICE: ada numarası= 102, ada kimlik numarası= 101102, mezarlık kimlik numarası= 101		
NOTICE: ada numarası= 103, ada kimlik numarası= 101103, mezarlık kimlik numarası= 101		

Şekil 166

Sınama Yapıları

Sınama yapıları sayesinde bir veya birden fazla parametrenin sınanması sağlanabilir. En bilinen sınama yapıları if – then, case when yapılarıdır. Bölüm içinde sırasıyla anlatılacaktır.

If – Then yapısı

If – Then sınama yapısı 3 farklı durumda kullanılabilir:

(1. durum)

```
If (sınama) then
--sınama olumlu
End if;
```

(2. durum)

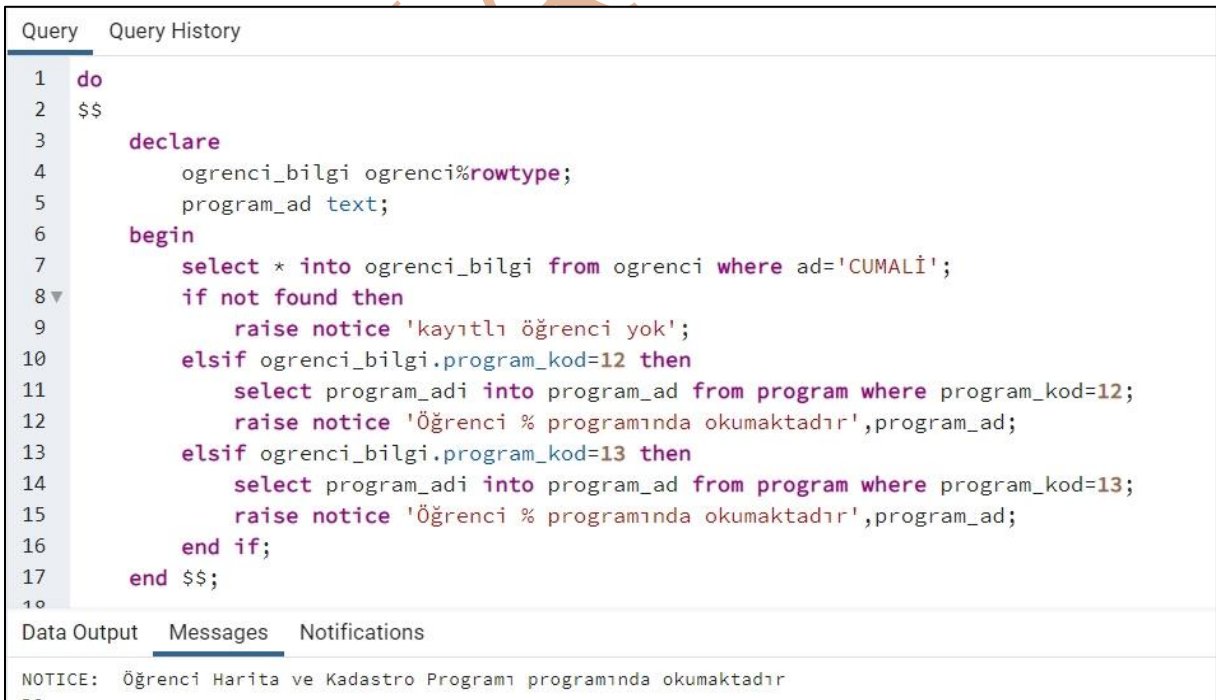
```
If (sınama) then
--sınama olumlu
Else
--sınama olumsuz
End if;
```

(3. durum)

```
If (sınama) then
--sınama olumlu
Elsif (yenisınama) then
--sınama olumlu
else
--sınamalar olumsuz
End if;
```

Örnek: Öğrenci tablosu içinde ad bilgisi CUMALİ olan öğrencinin okuduğu programı bulan program kodunu PL/pgSQL dili yazınız

Cevap:



```
Query Query History
1 do
2 $$
3 declare
4     ogrenci_bilgi ogrenci%rowtype;
5     program_ad text;
6 begin
7     select * into ogrenci_bilgi from ogrenci where ad='CUMALİ';
8     if not found then
9         raise notice 'kayıtlı öğrenci yok';
10    elsif ogrenci_bilgi.program_kod=12 then
11        select program_adi into program_ad from program where program_kod=12;
12        raise notice 'Öğrenci % programında okumaktadır',program_ad;
13    elsif ogrenci_bilgi.program_kod=13 then
14        select program_adi into program_ad from program where program_kod=13;
15        raise notice 'Öğrenci % programında okumaktadır',program_ad;
16    end if;
17 end $$;
18
```

Data Output Messages Notifications

NOTICE: Öğrenci Harita ve Kadastro Programı programında okumaktadır

Şekil 167

Şekil 167 satır 8'de *found* ifadesi bulunmaktadır. *found* tanımlı bir ifadedir. Sadece *found* olarak bırakılırsa kayıt bulundu anlamındadır. *not found* şeklinde kullanılırsa kayıt bulunamadı anlamındadır.

Döngü Yapıları

Aynı işlemin birden fazla defa yapılması gerektiğinde, tabloların verilerine tek tek ulaşmak gerektiğinde döngü yapılarına ihtiyaç duyarız. PL/pgSQL dili içinde döngü yapıları belge içinde işlenecek.

Loop Döngüsü:

Loop döngüsü, kendi başına uygulandığında herhangi bir kısıtlama olmadığı için sonsuz defa dönebilen bir döngüdür. *Exit* veya *return* ifadeleri kullanılarak döngü içinden çıkılabilir. *ifade 85* kısıtlama olmadan oluşturulan loop döngüsü örneğidir. Kısıtlama olmadığı için sonsuz bir döngü içindedir. Döngü programın içine konulduğunda, algoritmanın tasarlanmasına göre, kod satırı işlem sırasında direkt döngü içine girilmektedir. Döngüye girildiği anda döngüden çıkılmamaktadır.

```

ifade 85
Loop
-- Kod satırları
End loop;

```

ifade 86, *ifade 85*'den farklı olarak döngü içine bir *if – end if* yapısıyla sınıma imkanı sağlamıştır. Algoritma tasarımına göre kod satırı döngüye geldiğinde hiçbir şart aranmadan döngü içine girilmektedir. Eğer sınıma sağlanırsa *exit* ifadesiyle döngüden çıkılacaktır.

```

ifade 86
Loop
-- kod satırları
if (sınama) then
exit;
end if;
end loop;

```

While – loop Döngüsü:

While – loop döngüsü, döngüye giriş sırasında bir sınıma ile döngüye giriş sağlar. Sınıma sağlandığı sürece döngü yapısı dönecektir. Eğer sınıma sağlanmıyorsa döngüden çıkılır *ifade 87*.

ifade 87

```

While (sınama) Loop
  -- kod satırları
end loop;

```

For – Loop Döngüsü:

Kısıtlama olmadan belirli artış değerleriyle belirli sayıda döngünün oluşturulması isteniyorsa kullanılacak döngü yapısıdır. Özellikle tablo içindeki verilerin okunması için kullanılabilir döngü yapısıdır.

ifade 88 for döngüsü yazılış örneğidir. *tam_sayi_degisken* ifadesi daha önce declare bloğunda belirtilmek zorunda olmayan tam sayı bir değişkendir. [*artış/azalış*] ifadesi yazılmak zorunda değildir. Eğer *reverse* ifadesi yazılırsa azalacak şekilde *tam_sayi_degisken* değeri azalır. *başlangıç..bitiş* ifadesi yazılmak zorundadır. Döngünün başlangıç ve bitiş tam sayı değerleri yazılmalıdır. Bu kısımda bir sorgu ifadesi kullanılabilir. [*artış miktarı*] ifadesi yazılmak zorunda değildir. Döngüde belirtilen başlangıç değerinden itibaren bitiş değerine kadar tam sayı artış miktarının belirtildiği kısımdır. Eğer [*artış miktarı*] yazılmazsa ve pozitif artış yapılıyorsa artış miktarı birer artış miktarıyla artacaktır.

ifade 88

```

for tam_sayi_degisken in [artış/azalış] başlangıç..bitiş [artış miktarı] Loop
  -- kod satırları
end loop;

```

Şekil 168 1'den 5'e kadar aralıkta birer artışla sayı değerini döngü içinde yazılmasını sağlayan program kodu bulunmaktadır. Kod satırlarına dikkat edilirse, say değişkeni declare bloğu içinde tanımlanmamıştır. Döngünün artış/azalış olduğu belirtilmemiş ve artış miktarı belirtilmemiştir.

Query	Query History
1	do
2	\$\$
3	begin
4	for say in 1..5 loop
5	raise notice 'say değeri%=',say;
6	end loop ;
7	end \$\$;

Data Output	Messages	Notifications
NOTICE:	say değeri1=	
NOTICE:	say değeri2=	
NOTICE:	say değeri3=	
NOTICE:	say değeri4=	
NOTICE:	say değeri5=	
DO		
Query returned successfully in 29 msec.		

Şekil 168

Şekil 169 farklı bir for – loop döngü örneğidir. Örnekte for ifadesinden sonra record veri tipinde bir değişken tanımlanmıştır. Record veri tipi %rowtype veri tipinin daha kolay kullanımınıdır. For – loop döngü tanımında in ifadesinden sonra Select seçim sorgusu vardır. Tablonun tüm kayıtları seçilmiştir. For – loop döngüsünde tüm tablo kayıtları sırasıyla okunmuştur.

Query	Query History
1	do
2	\$\$
3	declare
4	ada_bilgi record;
5	ada_kayit integer:=0;
6	begin
7	select count(*) into ada_kayit from ada;
8	if ada_kayit>0 then
9	raise notice 'ada kaydı var';
10	for ada_bilgi in select * from ada loop
11	raise notice 'ada no=%, ada kimlik=%',ada_bilgi.ada_no,ada_bilgi.kimlik_ada;
12	end loop ;
13	end if ;
14	end \$\$;

Data Output	Messages	Notifications
NOTICE:	ada kaydı var	
NOTICE:	ada no=101, ada kimlik=101101	
NOTICE:	ada no=102, ada kimlik=101102	
NOTICE:	ada no=103, ada kimlik=101103	
DO		

Şekil 169

PostgreSQL Veri Tabanı Yönetim Sisteminde Kullanıcı Tanımlı Fonksiyon Oluşturulması

Fonksiyon içerisine aldığı parametreyi işleme katıp geriye döndüren araçtır. Veri tabanı yönetim sistemlerinde tanımlanan her bir veri tabanı için ayrı fonksiyonlar tanımlanabilir. PostgreSQL veri tabanı yönetim sisteminde kullanıcı tanımlı fonksiyon oluşturmak için *ifade 89*'da belirtilen yapı kullanılır. PostgreSQL veri tabanı yönetim sisteminde veri tabanı veya tablo oluşturmak için *Create* ifadesi kullanılır. Tanımlı veri tabanı içinde fonksiyon tanımlamak için de *Create* ifadesi kullanılır. Fonksiyon içindeki program kodlaması için PL/pgSQL dili kullanılır.

Fonksiyon tanımı *create [or replace]function* cümleciği kullanılır. *[or replace]* ifadesi eğer var olan fonksiyon değiştirilmek isteniyorsa kullanılır.

Fonksiyon adı verildikten sonra parantez içinde, fonksiyona çağırıldığı yerden eklenecek, parametreler eklenir. Fonksiyona parametre ekleme zorunluluğu yoktur.

Parametre parantezi kapatıldıktan sonra, fonksiyonun geriye döndüreceği değerini veri tipi *returns* ifadesinden sonra belirtilir.

Fonksiyon tanımında en son kullanılacak olan programlama dili *language* ifadesinden sonra belirtilir. PostgreSQL veri tabanı yönetim sisteminde programlama dili olarak PL/pgSQL programlama dili kullanılmaktadır. *ifade 89* fonksiyon tanım örneğinde görüldüğü üzere *language plpgsql* kullanılmıştır. *as* ifadesinden sonra \$\$ bloğu içinde fonksiyon için gerekli olan program kodu yazılır.

Begin – end bloğu içinde *return* ifadesiyle fonksiyonun çağırıldığı yere fonksiyondan istenilen değer döndürülür.

ifade 89

create [or replace]function *fonksiyon_adi*(*parametre_adi1* *veri_tipi*, ...)

returns *donus_veritipi*

language plpgsql

as

\$\$

declare

-- *kullanıcının_tanımladığı değişkenler*

begin

-- *fonksiyon yazıldığı satırlar*

return *deger*;

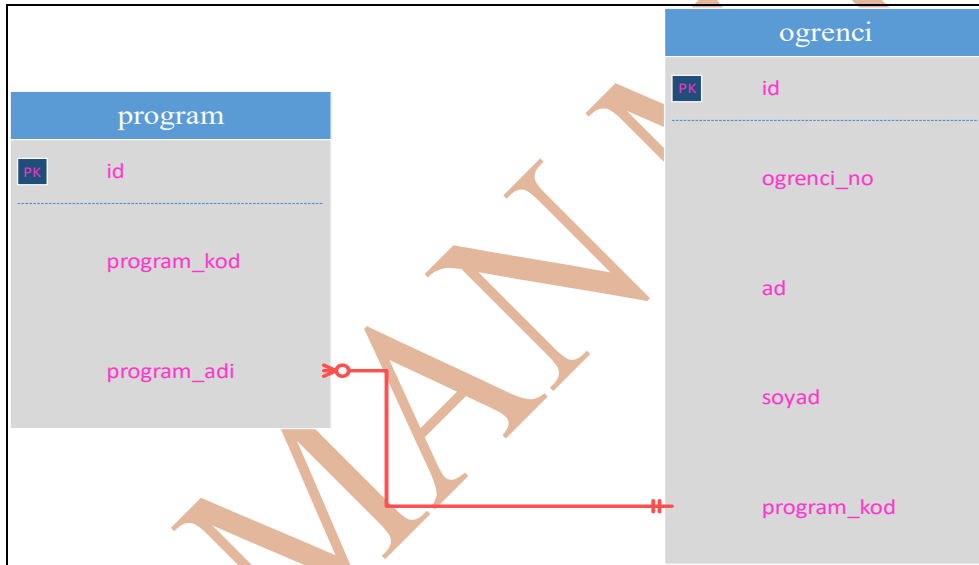
end;

\$\$

Örnek: myo adlı veri tabanı içinde program adı girildiğinde programda kayıtlı öğrenci sayısını veren fonksiyonun oluşturulması.

Cevap:

Programa ait bilgiler **program** adlı tablo içinde tutulmakta. Öğrencilere ait bilgiler öğrenci adlı tabloda tutulmaktadır. Fonksiyona parametre olarak aktarılacak **program** adı program tablosundadır. İlk olarak **program** tablosundan program_kod bilgisi içindeki veri çekilecek, sonrasında öğrenci tablosu içinde belirtilen program_kod bilgisiyle kayıt edilmiş öğrenci sayısı elde edilecek. *Şekil 170* tablolar ve tablolar arasındaki bağlantı gösterilmiştir.



Şekil 170

Şekil 171 istenilen fonksiyonun pgadmin editöründe Query Tool penceresinde yazılmış ve çalıştırılmış halini gösterir.


```

Query  Query History
1  create function prog_ogr_sayisi(prog_adi text) returns integer language plpgsql
2  as
3  $$
4      declare
5          sayi integer :=0;
6          prog_kod integer :=0;
7      begin
8          select program_kod into prog_kod from program where program_adi=prog_adi;
9          select cont(*) into sayi from ogrenci where program_kodu=prog_kod;
10         return sayi;
11     end $$;

```

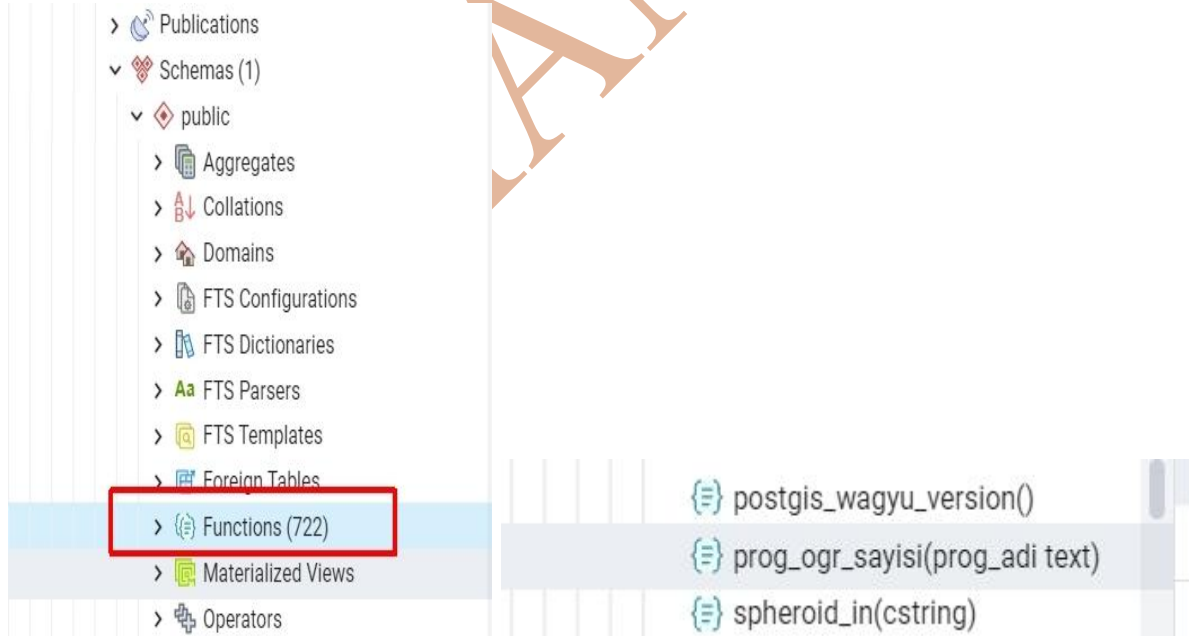
Data Output Messages Notifications

CREATE FUNCTION

Query returned successfully in 31 msec.

Şekil 171

Yazılan kod çalıştırıldığında Şekil 171’de görüldüğü gibi Create Function (Fonksiyon oluşturuldu) mesajı oluşturulur. Fonksiyonun oluşturulup oluşturulmadığını öğrenmek için veri tabanı içinde Schema içindeki functions (fonksiyonlar - Şekil 172 sol resim) ağaç yapısına bakılmalıdır. Fonksiyon verilen isimle oluşturulmuştur (Şekil 172 sağ resim).



Şekil 172

Fonksiyonun çalıştırılması için seçim sql cümlecği kullanılabilir. Şekil 173 fonksiyonun kullanımına bir örneğin temsilidir. Şekil 174 fonksiyonun doğru çalıştığının kanıtlarını gösteren sorgu cümleciklerin temsilidir.

Query		Query History	
1	<code>select prog_ogr_say('Makine Programı') as Öğrenci_Sayı;</code>		

Data Output		Messages		Notifications	
	Öğrenci_sayısı				
	integer				
1					30

Şekil 173

Query		Query History	
1	<code>select count(*) from ogrenci where program_kod=14;</code>		

Data Output		Messages		Notifications	
	count				
	bigint				
1					30

Şekil 174

Query		Query History	
1	<code>select count(*) from ogrenci,program</code>		
2	<code>where program.program_adi='Makine Programı'</code>		
3	<code>and ogrenci.program_kod=program.program_kod;</code>		

Data Output		Messages		Notifications	
	count				
	bigint				
1					30

PostgreSQL Veri Tabanı Yönetim Sisteminde Tetikleyici Fonksiyon ve Tetikleyici Kullanımı

Kullanıcı tanımlı fonksiyon oluşturmayı ve kullanmayı bir önceki konu başlığında anlatılmıştır. Tetikleyici fonksiyon ise veri tabanı içinde oluşacak tabloya kayıt ekleme (insert), tablodaki kaydın güncellenmesi (update), kaydın silinmesi (delete) gibi işlemler olduğu anda çalışacak fonksiyondur.

Belirtilecek işlem olduğunda fonksiyonun çalışabilmesi için, hem fonksiyonun tetiklenen fonksiyon olarak oluşturulduğu belirtilmeli, hem de fonksiyonun çalışmasını

sağlayacak tetikleyici oluşturulmalıdır. Tetikleyici kelimesinin İngilizce karşılığı trigger kelimesidir.

Tetiklenen fonksiyon oluşturulması, kullanıcı tanımlı fonksiyon oluşturulmasına benzer. Geriye dönüş tipi bir veri tipi değildir. Geriye dönüş veri tipi yerine *trigger* ifadesi kullanılacaktır.

ifade 90

Create function *fonksiyon_adi*(parametre1 veritipi) *return trigger language plpgsql*

As

\$\$

declare

– – *değişken tanımları*

begin

– – *program kodlarının olduğu kısım*

return new;

end \$\$;

Oluşturulan fonksiyonu çalıştırmak için fonksiyonun tetikçisi oluşturulmalıdır. *ifade 91* fonksiyon tetikçisinin oluşturulma yapısını göstermektedir. *Create trigger* İfadesinden sonra tetikleyicinin adı yazılacak. *Before/after* ifadeleri tetikleme işleminin insert/update/delete olaylarından önce (*Before*) ya da sonra (*after*) çalışacağını belirtmek için kullanılır. *On* İfadesinden sonra olayın (insert/update/delete olayları) gerçekleşeceği tablonun adı belirtilmelidir. *For each* ifadesinden sonra row gelirse işlemin yapılacağı her satıra ayrı ayrı işlemin uygulanmasını, statement gelirse işlemin tümü için bir kerede yapılacağı anlamındadır. *Execute procedure* ifadesinden sonra tetiklenen fonksiyonun adı yazılır.

ifade 91

Create trigger *tetikleyici_adi*

Before/after *OLAY*

On *olayın_oldugu_tablo_adi*

For each row/statement

Execute procedure *tetiklenen_fonksiyon_adi*();



For each row/statement yazılmak zorunda değil. Eğer yazılırsa, trigger her çalışmasında tablodaki her olay için işlem yapılacaktır. Örneğin bir kayıt ekleme (insert) işleminden sonra tetikle dediğimizde, tetiklenen fonksiyondaki işlem her satır için yapılacaktır. Eğer sadece tek bir satıra uygulanmasını istiyorsak bu satırı yazmak zorunda değiliz.

Örnek: Mezarlık bilgi sistemi için hazırlanmış tablolar vardır. Mezarlıkları coğrafi objeleri tutan tabloda her bir mezarlık eklenmesi işlemiyle beraber, mezarlık objesi için bir kimlik değerinin otomatik olarak oluşturulması istenmektedir. Bunun için tetikleyiciye ekleme işleminden sonra (after insert) mezarlık kimlik numarasının güncellenmesi isteni sağlayan fonksiyonu çalıştırması istenmiştir. Bu işlemi sağlayacak fonksiyon ve tetikleyiciyi program kodunu yazınız.

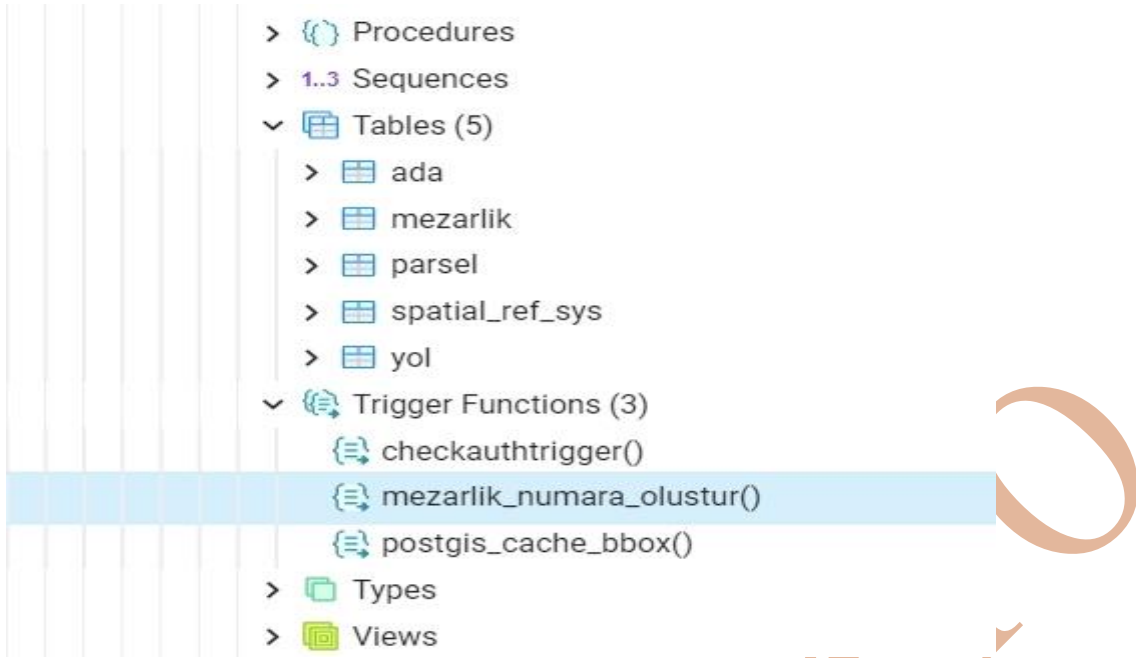
Cevap: Şekil 175 istenilen fonksiyon program kodunu temsil etmektedir. Fonksiyon çalıştırılınca veri tabanı ağaç yapısı altındaki Trigger Function yapısı içinde oluşur (Şekil 176). Şekil 177 fonksiyonu çalıştıracak tetikleyici tanımını yapan program kodunu temsil etmektedir. Program kodu çalıştığında, tetikleyici **mezarlik** tablosu için çalışacağı için, tablonun olduğu ağaç yapısı altında **Triggers** altında görülecektir (Şekil 178).

```

Query Query History
1 create or replace function mezarlik_numara_olustur() returns trigger language plpgsql
2 as
3 $$
4     declare
5         sayi integer:=0;
6         maks_id integer :=0;
7         maks_mez_kimlik integer=0;
8     begin
9         select count(*) into sayi from mezarlik;
10        raise notice 'sayi=%',sayi;
11        if sayi=1 then
12            update mezarlik set kimlik_mez=101;
13        elsif sayi>1 then
14            select max(kimlik_mez) into maks_mez_kimlik from mezarlik;
15            select max(id) into maks_id from mezarlik;
16            raise notice 'maks mez kimlik=%',maks_mez_kimlik;
17            update mezarlik set kimlik_mez=(maks_mez_kimlik+1) where id=maks_id;
18        end if;
19        return new;
20    end $$;
Data Output Messages Notifications
CREATE FUNCTION

```

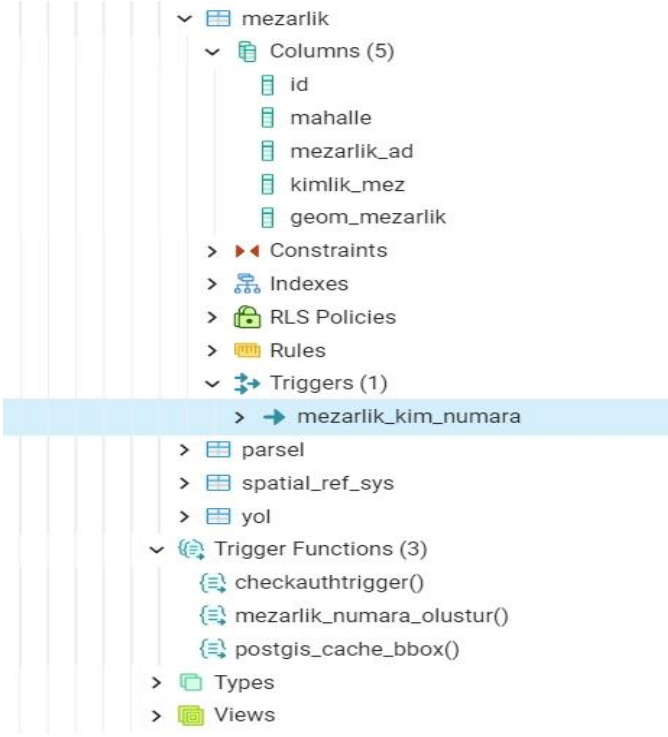
Şekil 175



Şekil 176

```
Query  Query History
1  create trigger mezarlik_kim_numara
2  after insert
3  on mezarlik
4  execute procedure mezarlik_numara_olustur();
```

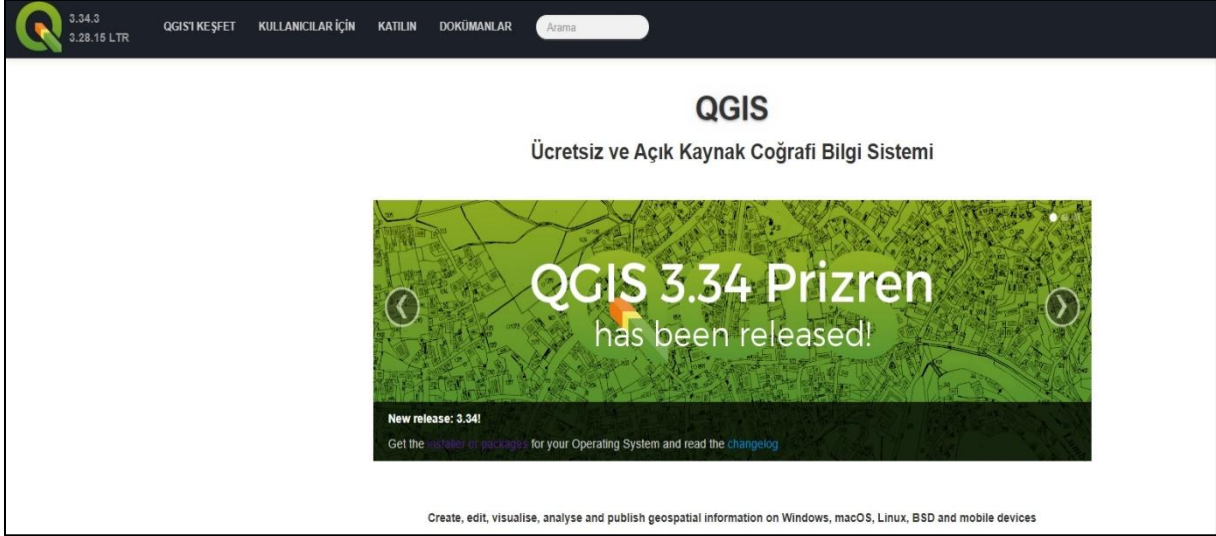
Şekil 177



Şekil 178

Quantum Geographic Information System (QGIS) Coğrafi Bilgi Sistemleri Yazılımından PostgreSQL Yazılımındaki Veri Tabanına Bağlantı Kurma

Quantum Geographic Information System (QGIS) yazılımı açık kaynaklı coğrafi bilgi sistemi kurulması, yönetilmesi, konum tabanlı analizlerin yapılmasını sağlayan yazılımdır.



Şekil 179 (QGIS, 2024)

QGIS CBS yazılımı, diğer bilinen CBS yazılımlarına göre PostgreSQL veri tabanı yönetim sistemi yazılımına daha kolay bağlantı kurar. QGIS yazılımı kolay bir şekilde PostgreSQL içindeki veri tabanında kayıtlı grafik veya sözel tipteki verilerin olduğu tablolara ulaşım veri çeker veya PostgreSQL içindeki veri tabanında kayıtlı tablolara veri gönderir. Bu işlemi birden farklı şekilde yapabilir. Konu başlığı içinde hem PostgreSQL veri tabanı yönetim sistemindeki bir veri tabanına veri gönderimi hem de veri tabanından veri çekimi işlemleri farklı yöntemler ile gösterilecektir.

QGIS Yazılımı Ara Yüzüyle Postgresql Veri Tabanı Yönetim Sitemindeki Veri Tabanına Bağlantı Kurulması

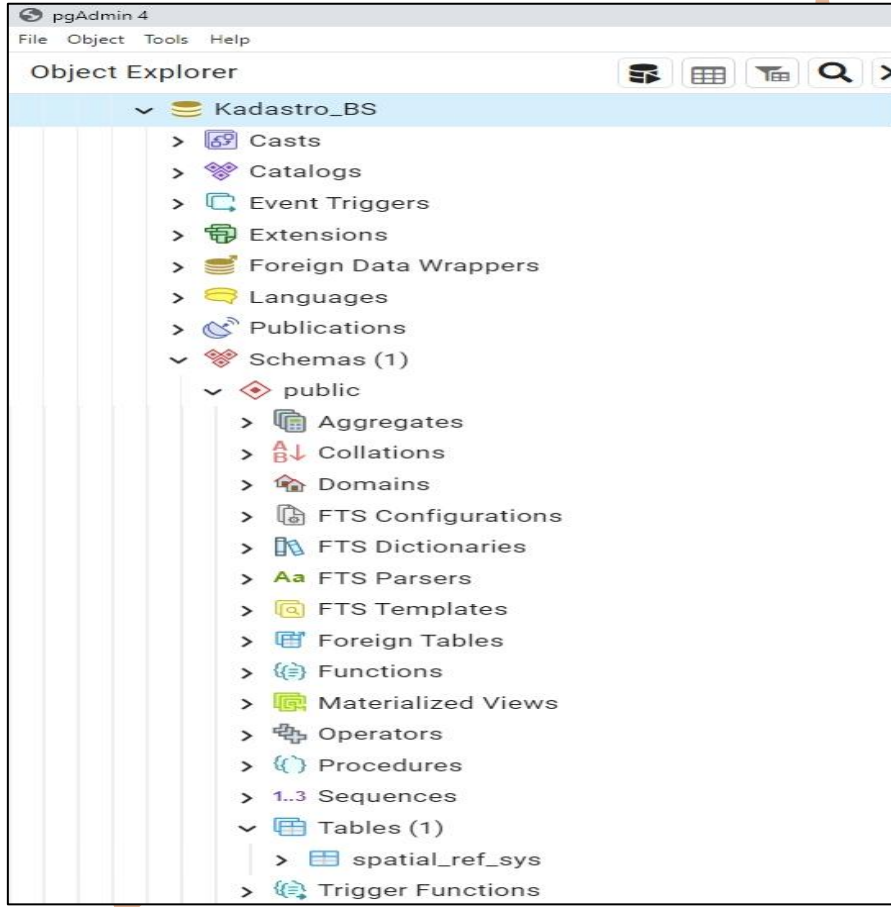
QGIS yazılımındaki Tabakaların PostgreSQL VTYS Tanımlanmış Veri Tabanı İçine Aktarılması

QGIS yazılımında oluşturulmuş tabakaların PostgreSQL yazılımı içinde tanımlanmış bir veri tabanına aktarılması için ilk önce QGIS yazılımında ilgili veri tabanına bağlantı sağlanmalıdır.

Yapılan örnek için PostgreSQL yazılımında **Kadastro_BS** adlı bir veri tabanı oluşturuldu (Şekil 180). Şekil 180 incelendiğinde **Kadastro_BS** veritabanı içine, postgres eklentisiyle gelen tablo dışında eklenmiş tablo yok.



Spatial_ref_sys (Spatial reference systems – konumsal referans sistemleri) tablosu PostgreSQL veri tabanı yönetim sistemi için konumsal bilginin kaydolmasını sağlayan eklenti kurulduğu anda otomatik oluşur. Tablo içinde tanımlı referans sistemleri, koordinat sistemleri, projeksiyonlar, elipsoitler, datumlar gibi bilgiler bulunmaktadır. Tanımlı olan çok fazla bilgi olduğu için Spatial_ref_sys tablodundaki kayıt sayısı çok fazladır ve tablonun boyutu çok büyüktür. Eğer Spatial_ref_sys tablosu ücretli bir sunucu içinde barınacaksa bu tablo içinde sadece kullanılacak olan ülke koordinat sistemlerini barındırmak mantıklı olacaktır.

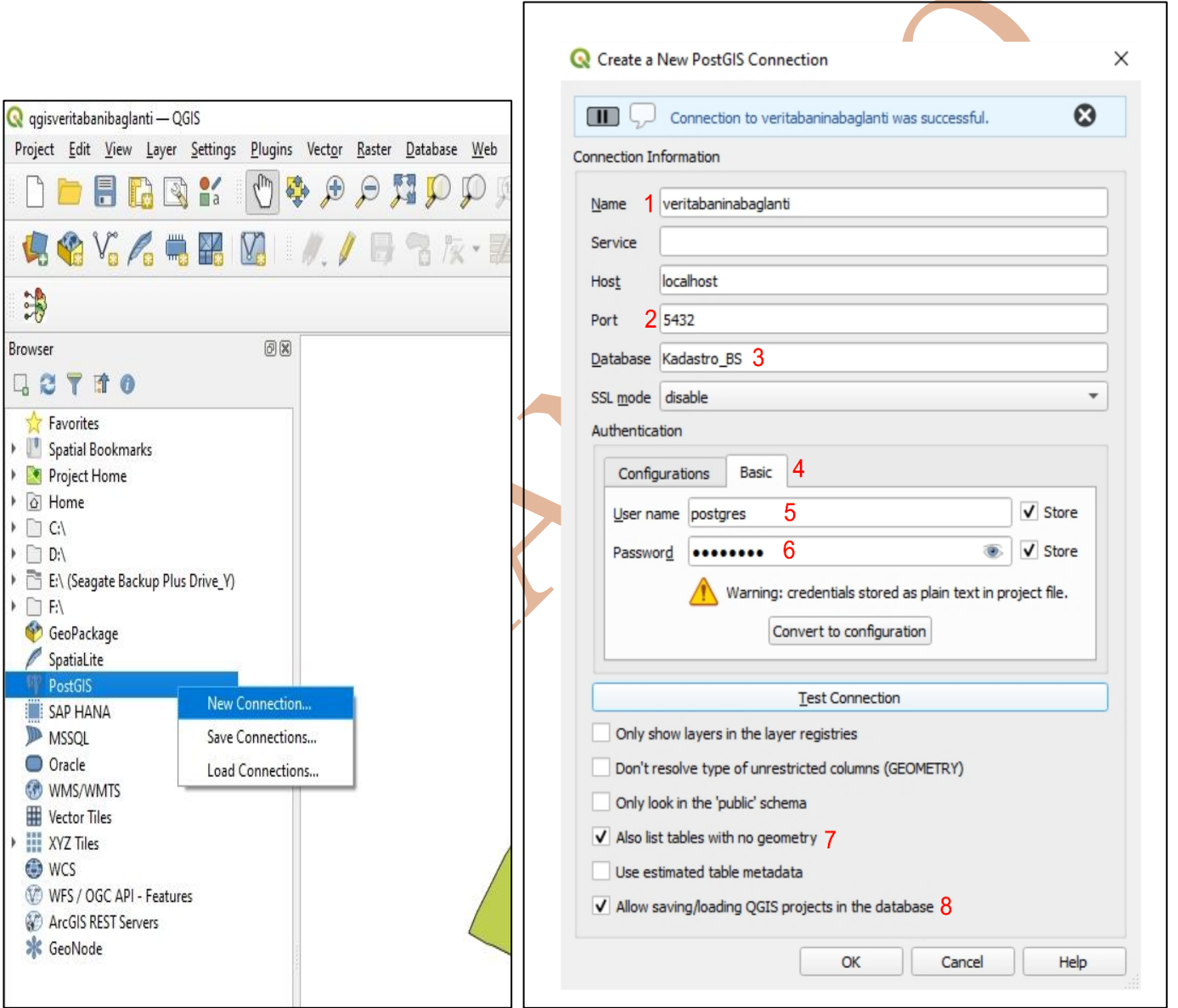


Şekil 180

QGIS yazılımından PostgreSQL içinde tanımlanmış veri tabanına ulaşabilmek için bağlantı kurulmalıdır. Bağlantı kurulma işlemi *Browser (tarayıcı)* Penceresi içinde PostGIS (veya PostgreSQL) alt menüsüne sağ tuşla tıklanıldığında açılan pencerede *New Connection (yeni bağlantı)* alt menüsü seçilir (Şekil 181 sol resim).

Açılan *Create a New PostGIS Connection* (Postgis ile yeni bir bağlantı oluştur) penceresinde bağlantı için gerekli parametreler girilir. Şekil 181 sağ resimde örnek için gerekli bağlantılar görülmektedir. Şekil 181 sağ resimde 1 numaralı kısımda oluşturulacak bağlantı için bir isim isteniyor. Bu isim kullanıcının kendisinin belirleyeceği bir isimdir. 2 numaralı kısımda

veri tabanının bulunduğu bilgisayarda bağlantı için kullanılacak port numarasının girilmesi isteniyor. Bu port numarası veri tabanının bulunduğu bilgisayarda PostgreSQL kurulumu esnasında belirlenmiş olan port numarasıdır. 3 numaralı kısımda PostgreSQL içinde oluşturulmuş ve iletişim kurulacak olan veri tabanının ismi girilecek. 4 numaralı sekme altında ulaşılabilecek veri tabanının kullanıcı adı ve şifre bilgisi girilecek. Bu kısımda *Store (kaydet)* onay kutuları işaretlenirse, QGIS yazılımından veri tabanına daha sonraki erişimlerde tekrar kullanıcı adı ve şifre girişlerine gerek kalmayacaktır.



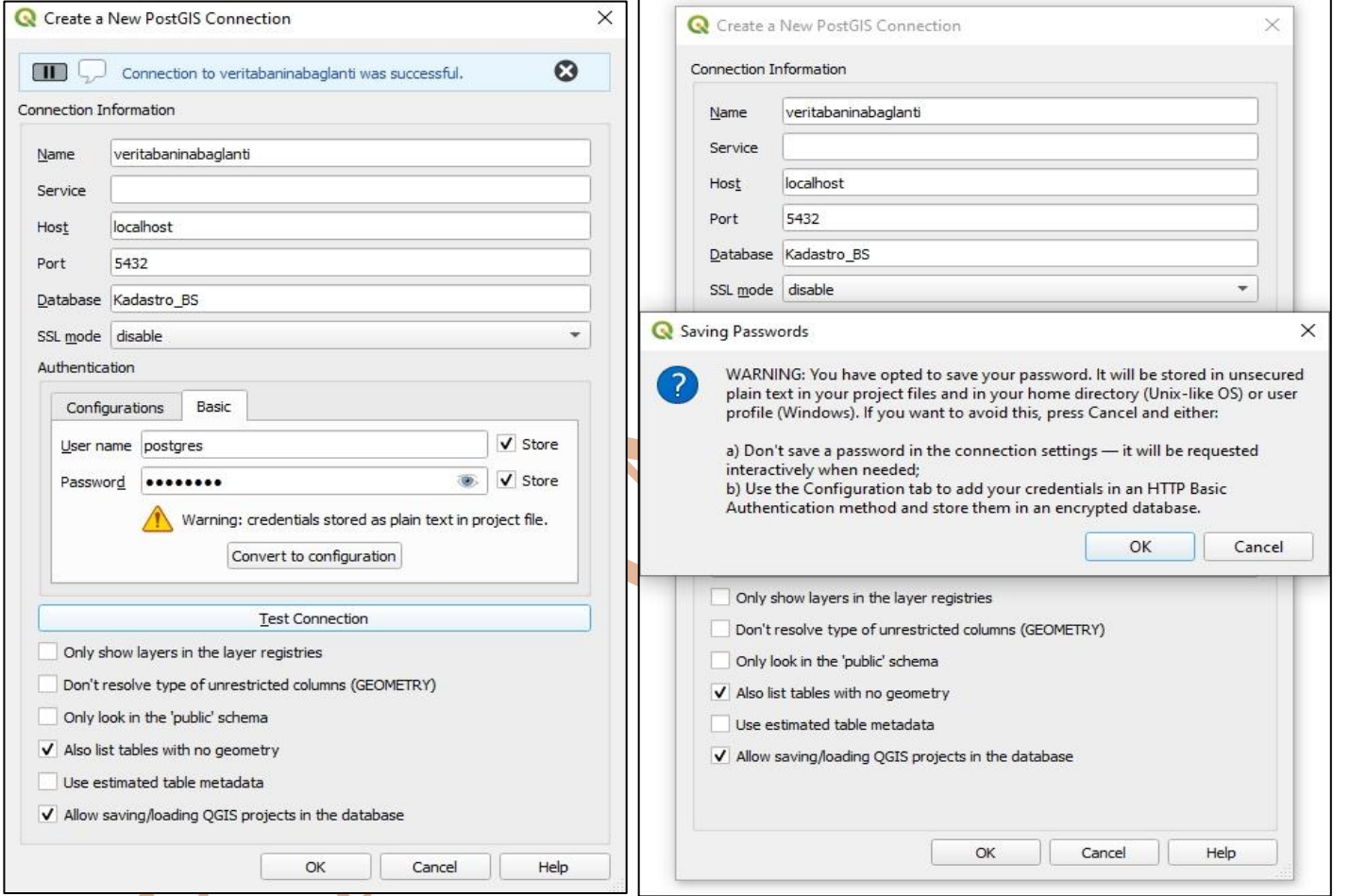
Şekil 181

Şekil 181 sağ resimde alttaki iki onay kutusu da seçilmiştir. Onay kutularından (7 numaralı kısım) Also list tables with no geometry (Geometrik bilgileri olmayan tabloları da listele) ilişki kurulan veri tabanındaki sözel tabloların da QGIS yazılımında görülmesi ulaşılabileceğini sağlar. Diğer onay kutusu (8 numaralı kısım) Allow saving/loading QGIS projects

in the database (QGIS yazılımından veri tabanına veri kaydetme ve veriyi çekme işlemlerine izin ver). Bu iki onay kutusu çift taraflı işlemin yapılmasında

Şekil 182 sol resimde bağlantı bilgilerinin kontrol aşamasının temsili vardır. Bağlantının kontrol edilmesi için *Test Connection (Bağlantıyı test et)* düğmesine basılması sonucu bağlantının olumlu olduğuna dair görüntü bulunmaktadır.

Şekil 182 sağdaki resimde yapılan işlemde verilere ulaşımda olabilecek tehlikeli durumları belirtiyor. Tamam tuşuna basıldığında bağlantı kurulmuş olacak.



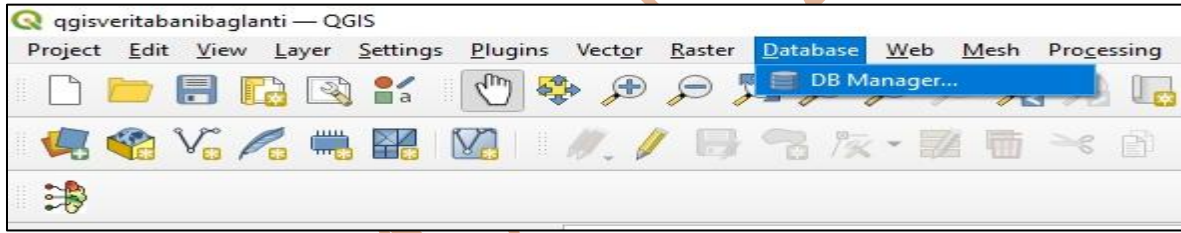
Şekil 182

Şekil 183 QGIS yazılımında oluşturulmuş kadastro adaları ve kadastro parsellerinin durumunu göstermektedir. Bu tabakaların oluşturulan bağlantı üzerinden PostgreSQL de var olan veri tabanına aktarmak için QGIS yazılımı içindeki aktarım aracı kullanılabilir.



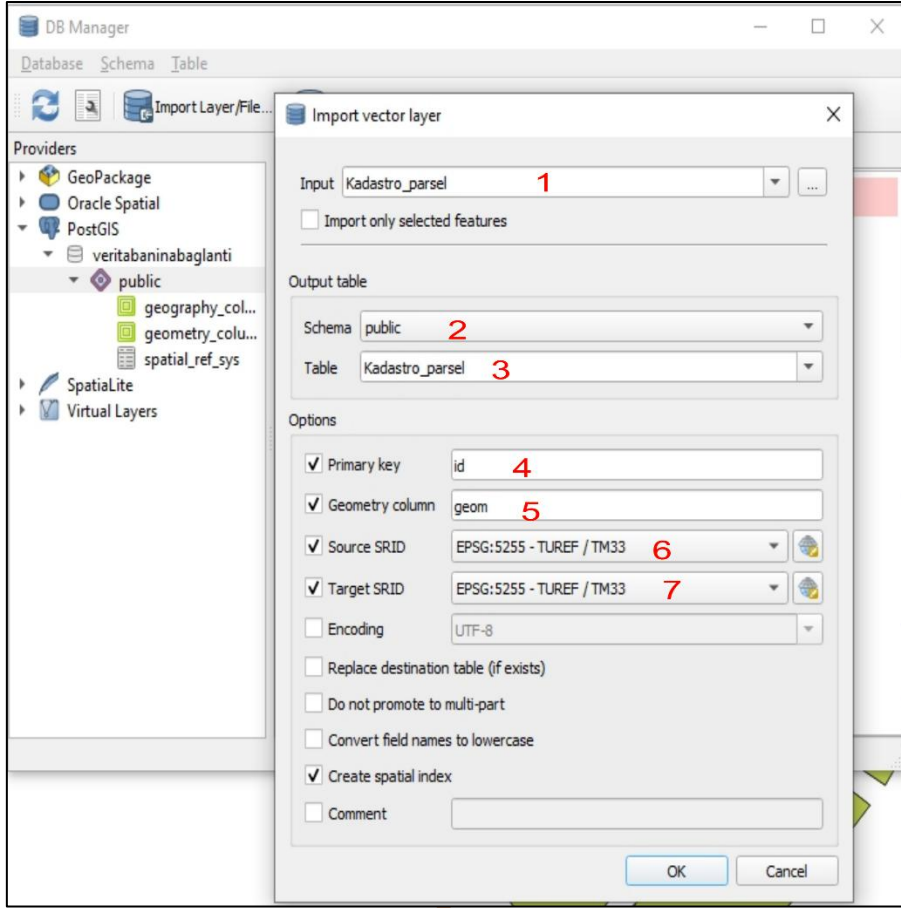
Şekil 183

QGIS yazılımından veri aktarımı yapabilmek için QGIS yazılımından *Database* menüsü alt menüsü olan *DB Manager* kullanılır (Şekil 184).



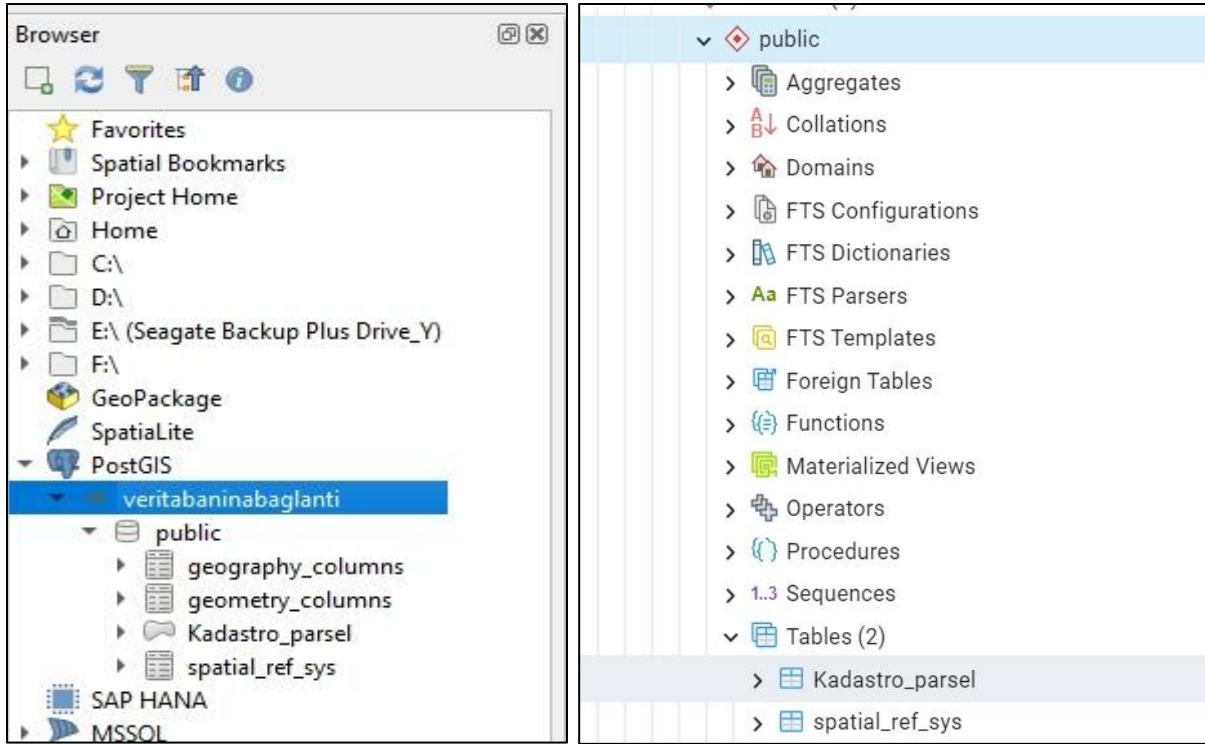
Şekil 184

Şekil 185 açılan *DB Manager* penceresinde yapılan işleme örnektir. Penceredeki *Import Layer/File* aracı seçildiğinde *Import Vector Layer* penceresi açılır. Şekil 185 1 numaralı PostgreSQL içinde oluşturulan *Kadastro_BS* veri tabanına aktarılacak tablo seçiliyor. 2 numaralı kısımda veri tabanı içindeki şema seçiliyor. Örnekte *public* şeması seçilmiş. 3 numaralı kısımda tablonun veri tabanı içindeki ismi belirtiliyor. 4, 5, 6 ve 7 numaralı sahalara isteğe bağlı sahalara. Fakat kullanıcının seçim yapması gerekli sahalara. 4 numaralı kısımda anahtar sahanın hangi saha olacağı belirtiliyor. Örnekte *id* ismi verilmiş. 5 numaralı kısımda geometrik veriyi tutacak olan sahanın ismi belirtiliyor. Örnekte *geom* adı verilmiş. 6 numaralı kısımda eklenen tablonun koordinat sistemi, 7 numaralı kısımda ise veri tabanında tutulacak tablonun koordinat sisteminin seçimleri yapılmakta. Örnekte her iki koordinat sistemi de TUREF – TM33 olarak belirtilmiş. Create Spatial index onayı işaretlenmiş. Bu onay sayesinde verilere ulaşmak daha kolay olacak.



Şekil 185

Şekil 186 sol resimde QGIS yazılımı Browser penceresinde, PostgreSQL yazılımındaki veri tabanına, tabakanın aktarıldığı görülmektedir. Şekil 186 sağ resimde ise Pgdmin 4 yazılımında PostgreSQL yazılımı içindeki veri tabanına tabakanın aktarıldığı görülmektedir.



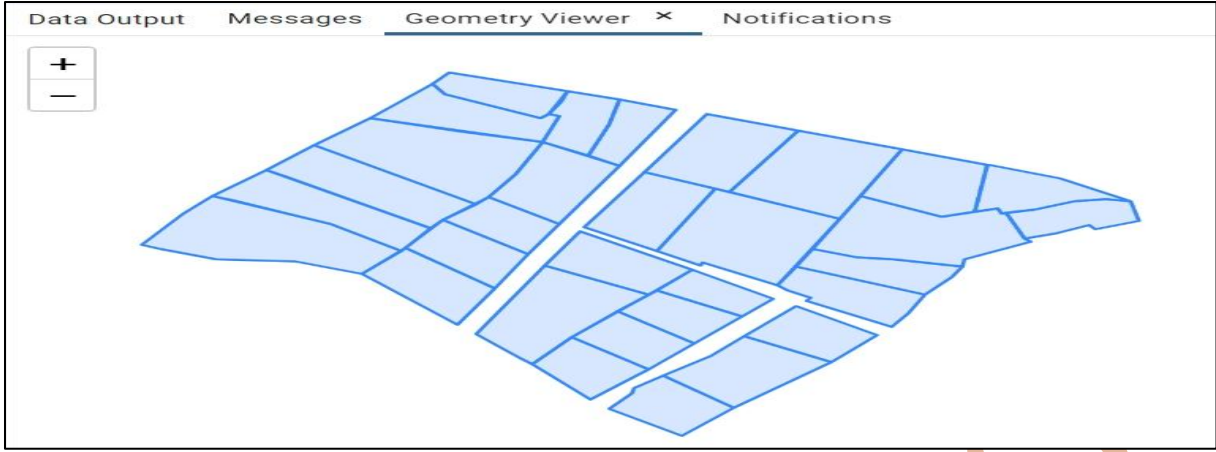
Şekil 186

Şekil 187 *Kadastro_BS* veri tabanındaki objelerin kayıtlarının listesi gözükmektedir.

Data Output		Messages	Notifications
id	[PK] integer	geom	geometry
1	1	010600002087140000010000000103000000010000000F0000001F85EBD1D8272141B81E	
2	2	0106000020871400000100000001030000000100000011000000AE47E1FA1B292141D7A3	
3	3	010600002087140000010000000103000000010000000A0000001F85EBD179282141EC51	
4	4	0106000020871400000100000001030000000100000008000000666666E614282141713D0	
5	5	010600002087140000010000000103000000010000000B000000D7A3703DFB2721411F85	
6	6	010600002087140000010000000103000000010000000900000052B81E85D32821419A99	
7	7	010600002087140000010000000103000000010000000A0000001F85EB51332921416666	
8	8	0106000020871400000100000001030000000100000009000000CDCCCCC462821413D0	
9	9	0106000020871400000100000001030000000100000006000000295C8E4243282141C3E5	

Şekil 187

Şekil 188 Pgadmin 4 yazılımında, *Kadastro_BS* veri tabanındaki içinde kayıtlı grafik objelerin tabaka olarak görünümüdür.



Şekil 188

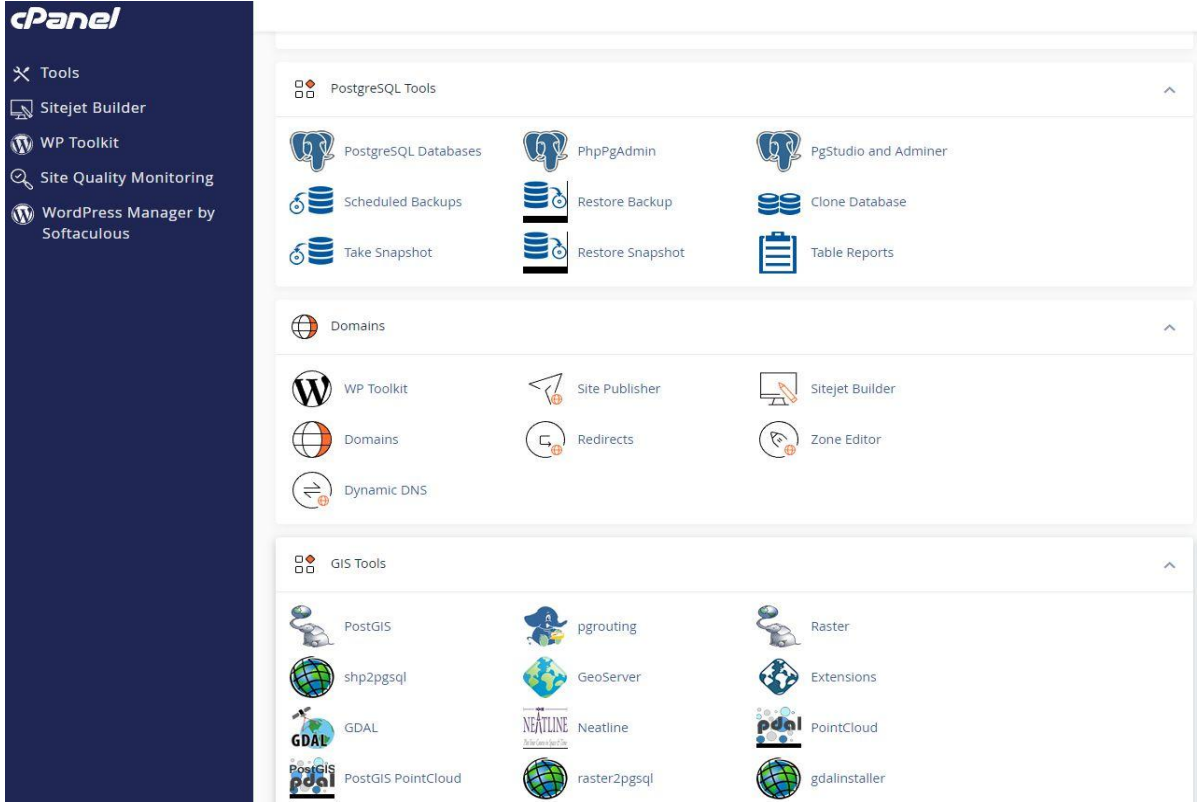
Uzak Sunucuda veya Bulut Ortamında Tabloların Tutulması

Konu başlıkları altında gösterilen uygulamalar PostgreSQL veri tabanı yönetim sisteminin kullanıcı bilgisayarında olduğu uygulamalardır. PostgreSQL veri tabanına PgAdmin yazılımıyla veya QGIS yazılımıyla bağlantı kurulmasında kullanıcının kendi bilgisayarından bağlantı kurulduğu için uzak bilgisayar bağlantısının da bu şekilde kolaylıkla yapılabileceği düşünülüyor.

Uzak Sunucuda Tabloların Tutulması

Uzak bilgisayardan kastedilen, internet sitesi veya veri tabanı için Hosting (barındırma) hizmeti veren firmaların sahip oldukları server olarak ifade edilen 7/24 aralıksız çalışan ve ulaşılabilen bilgisayarlara verilen isimdir. Hosting hizmeti veren firmalar her daim size hizmet verecek bilgisayarlara PostgreSQL veri tabanı yönetim sistemi kurmayabilirler. Ek olarak server üzerine kurdukları PostgreSQL veri tabanı yönetim sistemine Postgis eklentisi kurma desteği vermeyebilirler. Bu durumda elinizdeki grafik objeyi tutabilecek veri tabanları ve tablolar oluşturamazsınız, konumsal sorgulamalar yapacak fonksiyonlara sahip olamazsınız.

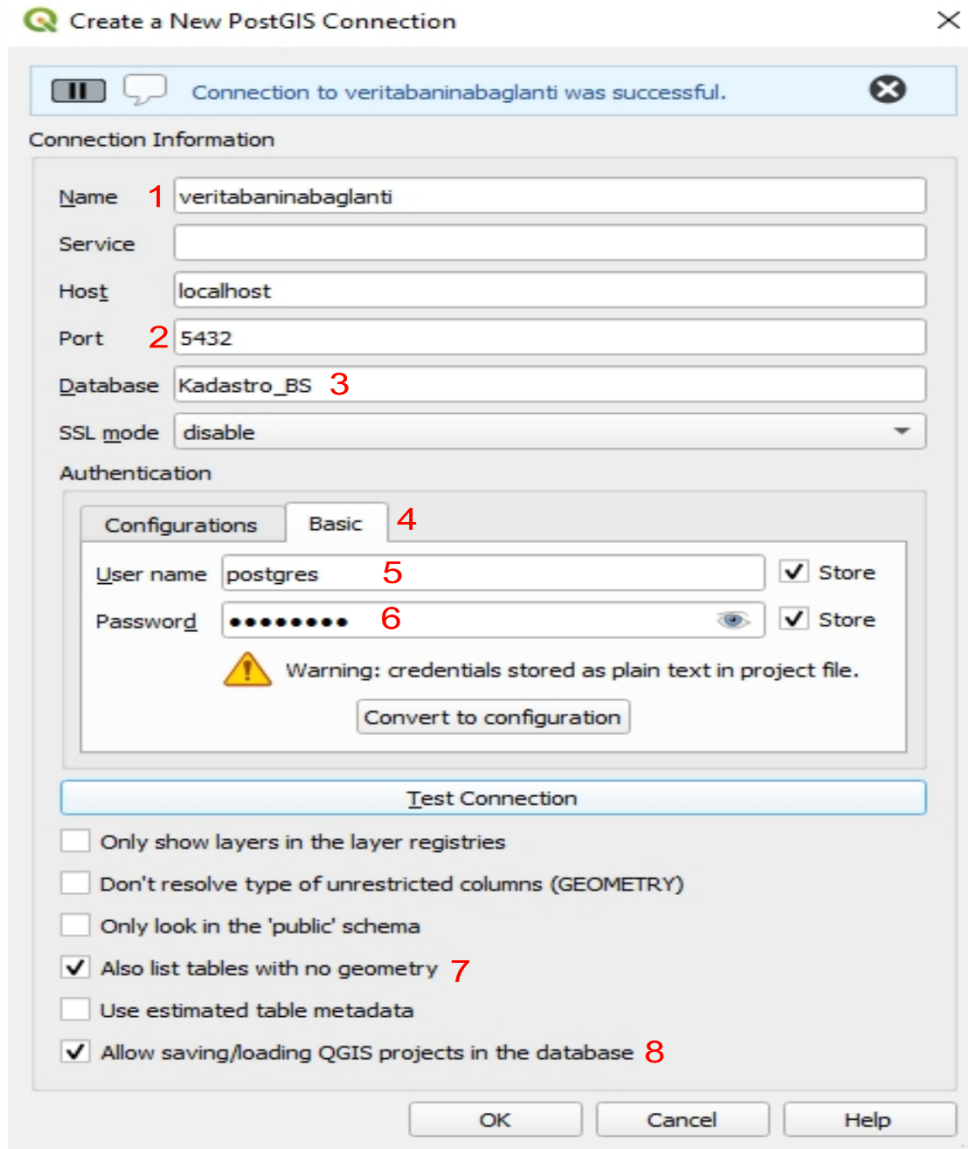
Uygulamaların anlatımını sağlamak için AcuGIS şirketinin hosting hizmetinden faydalanıldı. AcuGIS hosting hizmetinin tercih edilmesinin nedeni hem hizmet verilecek bilgisayar üzerinde veri tabanı yönetim sistemi olarak PostgreSQL kurulumu yapmaları hem de veri tabanı üzerinde Postgis eklentisi kurma imkânı sağlamalarıydı. Şekil 189 AcuGIS firmasının kullanıcıya sunduğu hosting hizmetinin yönetimi için arayüzün belirli bir kısmına ait görüntüdür.



Şekil 189

PgAdmin veya Qgis yazılımlarından uzak bilgisayarlara bağlanmak için barındırma hizmeti için verilen IP adresi veya site için satın aldığınız ad ve site yönetimine giriş için belirlenmiş şifre kullanılacak. Qgis yazılımından uzak bilgisayardaki PostgreSQL veri tabanında tanımlanan veri tabanına bağlantı için Şekil 190 üzerinde görünen kısımlara girilecek bilgiler aşağıda tanımlanmıştır:

- 1 numaralı kısımda kullanıcının bağlantı için vermek istediği herhangi bir isim yazılacak,
- Host kısmında internet sitesinin IP adresi veya site adı,
- 2 numaralı kısımda siteye giriş için kullanılacak port (genelde 5432),
- 3 numaralı kısımda sunucudaki PostgreSQL veri tabanı yönetim sisteminde tanımlanmış olan ve bağlanılacak olan veri tabanının adı,
- 5 ve 6 numaralı kısımlarda sunucunun kullanımı için tanımlanmış ve sunucuya bağlanmak için belirlenmiş kullanıcı adı ve şifre verileri girilecek.



Şekil 190

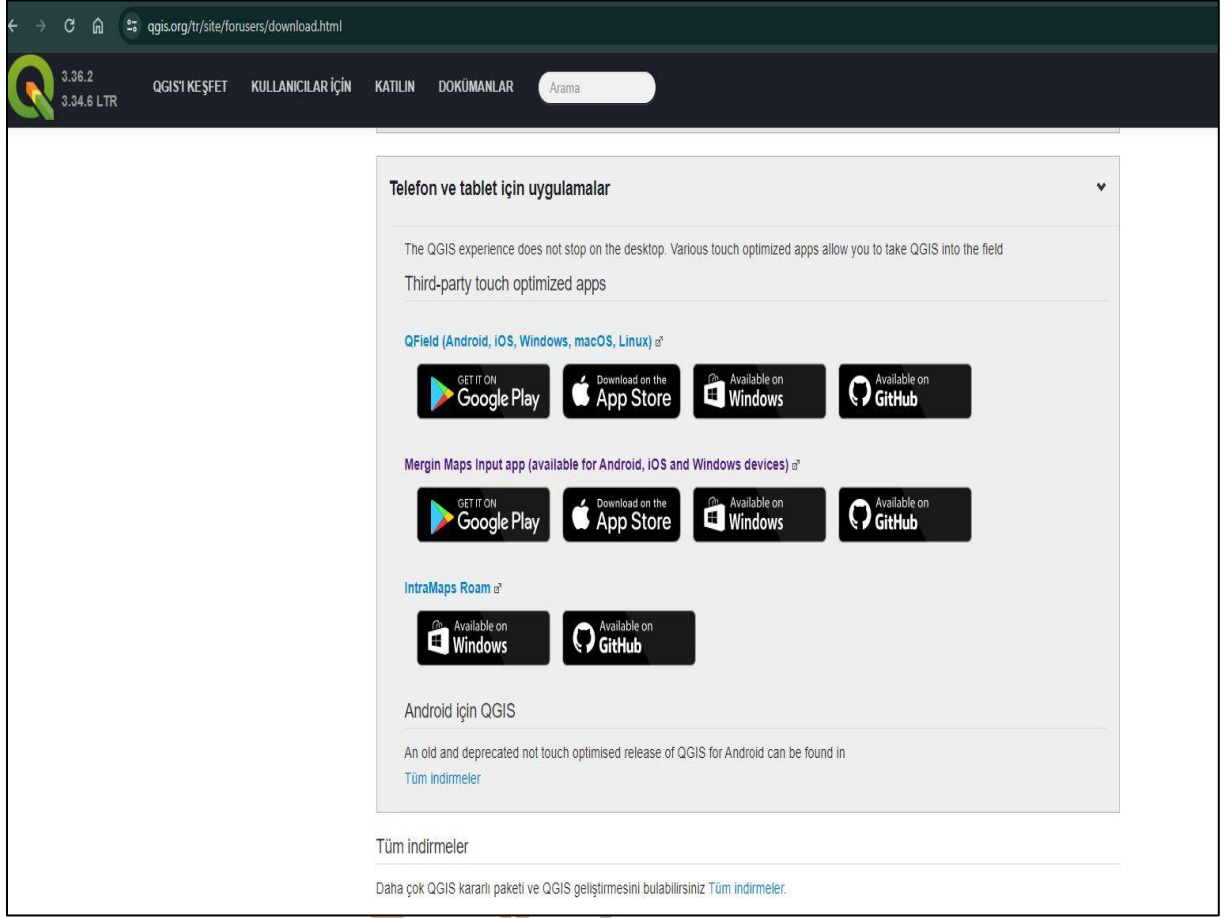
Bağlantılar sağlandıktan sonra konu başlıkları altındaki işlemler aynı şekilde yapılacaktır.

Bulut Ortamında Tabloların Tutulması

Qgis yazılımı açık kaynaklı olması yanı sıra yazılımın kendi bünyesindeki araçlara ek olarak Qgis yazılımcılarının veya kullanıcılarının geliştirdiği eklentilerle gün be gün daha etkin bir coğrafi bilgi sistemi yazılımına dönüşmektedir.

Qgis yazılımını masaüstü bilgisayara kurmak için veya farklı teknoloji platformlarına (tablet veya cep telefonu) kurmak için aşağıdaki linkteki siteyi kullanabilirsiniz (Şekil 191).

<https://qgis.org/tr/site/forusers/download.html> (Quantum Geographic Information System, 2024)



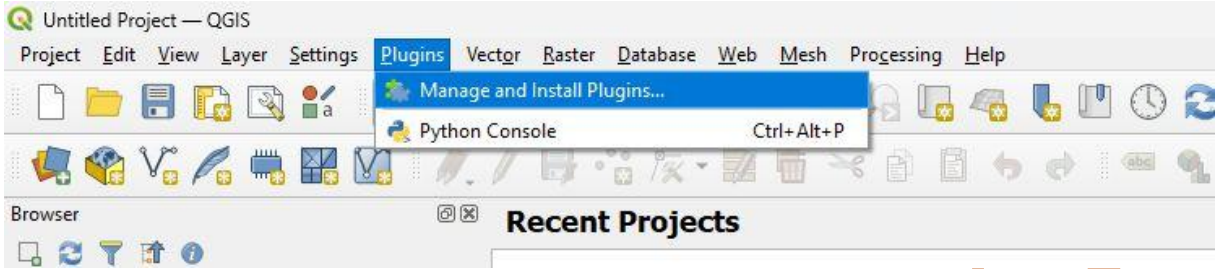
Şekil 191(05.05.2024)

Şekil 191 sitenin temsili görünümüdür. Sitede Android ve IOS işlemcili telefon ve tabletler için kullanılabilir Qfield uygulaması görülmektedir. Qfield uygulaması, QGIS yazılımında oluşturulan haritaların telefon veya tablet ortamında kullanılmasını sağlamaktadır. QGIS ortamında oluşturulmuş haritalar uzak bilgisayardan QGIS yazılımına eklenmiş olabilir. Bu sayede Qgis ortamında oluşturulan harita Qfield yazılımı ile cep telefonu veya tablete gönderildiğinde esasında uzak bilgisayar ile cep telefonu veya tablet iletişime geçmiş olacaktır.

Eğer uzak bilgisayardan barındırma hizmeti almıyorsanız, Qfield bulut ortamını kullanabilirsiniz. Qfield bulut ortamı basit harita uygulamaları oluşturmak amaçlı kullanılabilir. Çok kullanıcının cep telefonuna yüklenecek olan ve sürekli veri alışverişi yapacak uygulamalar için Qfield bulut ortamı için ücret ödenmesi gerekir.

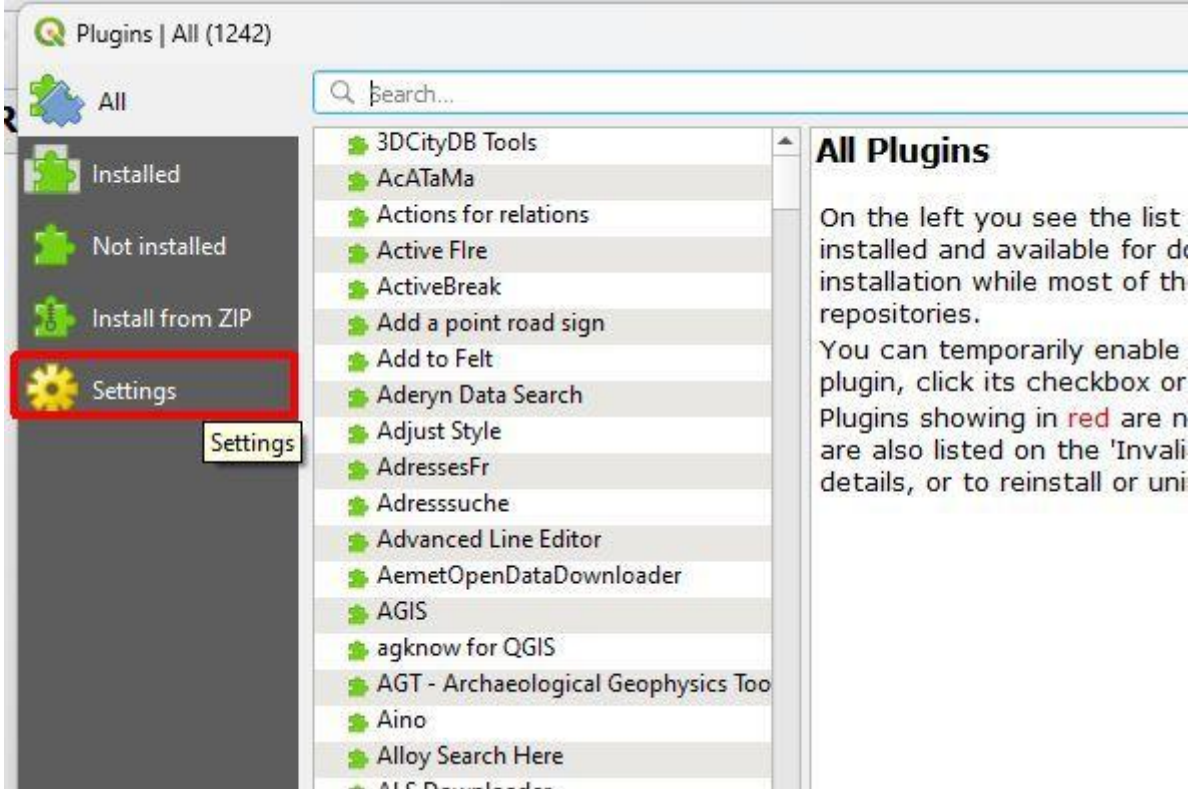
Qfield kullanarak cep telefonu veya tablet teknolojilerinde kullanılabilir harita tabanlı bir uygulama tasarlamak için bilgisayara kurulu olan QGIS yazılımına ve cep telefonuna (ya da tablete) Qfield eklentisi yüklenmelidir.

Qgis yazılımında Qgis yazılımcılarının ya da kullanıcılarının hazırladığı eklentileri eklemek için Plugins (eklentiler) menüsü ve Manage and Install Plugins... (Eklentileri yönet ve Ekle) alt menüsü kullanılmalıdır (Şekil 192).



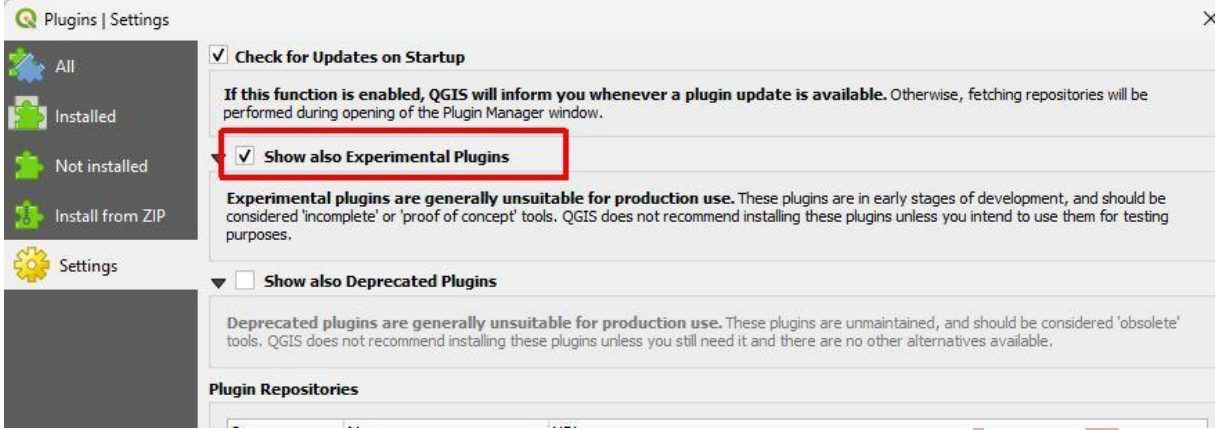
Şekil 192

Plugins penceresi açıldığında kullanıcılarında geliştirdiği eklentilerin yüklenebilmesi için bir ayar yapılmalıdır (Şekil 193).



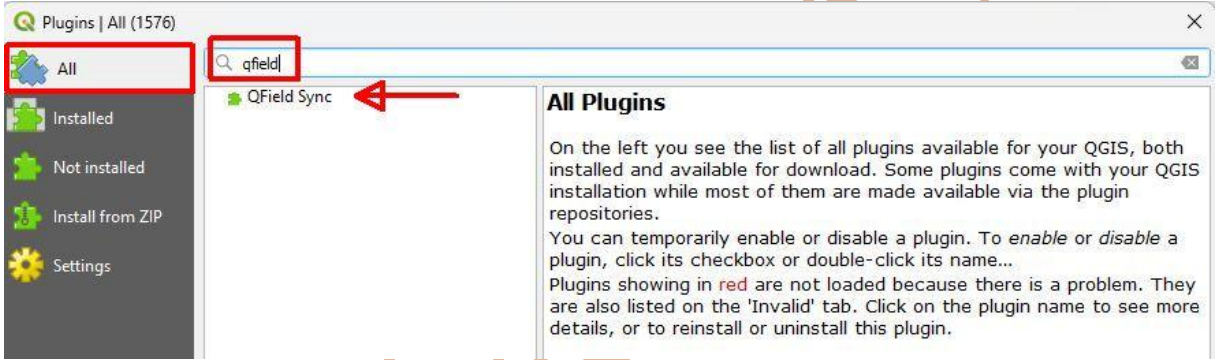
Şekil 193

Pencerede Settings (ayarlar) düğmesiyle açılan pencerede Show also Experimental Plugins (deneysel eklentileri de göster) onay kutusu seçilmelidir (Şekil 194). Bu eklenti Qfield ile ilgili değildir. Fakat Qgis yazılımında geliştireceğimiz projelerde bize yardımcı olacak eklentileri bulmakta fayda sağlayacaktır.



Şekil 194

Qfield eklentisini yüklemek için Plugins penceresinde All (hepsi) düğmesi seçilmeli ve arama çubuğuna qfield yazılması eklenti bulmayı kolaylaştıracaktır (Şekil 195).



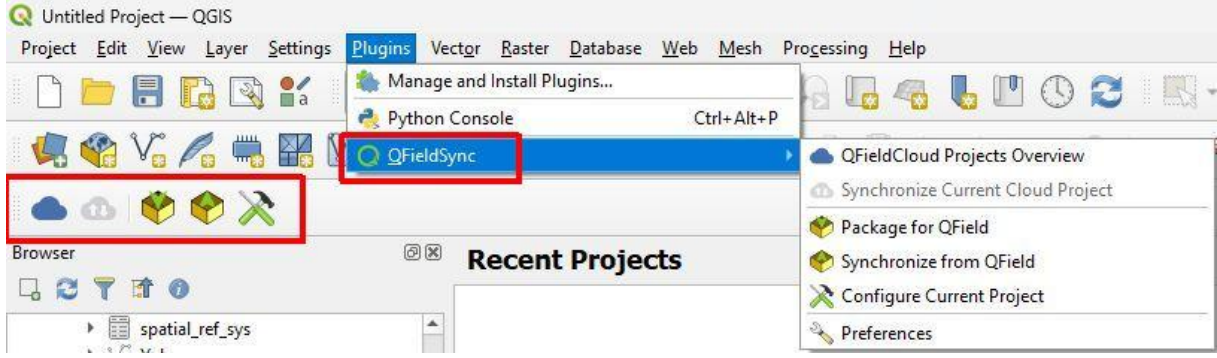
Şekil 195

Arama sonucunda çıkan Qfield Sync adlı eklenti seçilip Install Plugin (eklenti yükley) düğmesi tıklanması sonucu eklenti kullanılır hale gelecektir (Şekil 196).



Şekil 196

Şekil 197 eklenti yüklendikten sonra Plugins menüsünde araç çubuğunda Qfield kullanımı için oluşan menü ve araçlar gözükmemektedir.



Şekil 197

Qfield Eklentisiyle Bulut Ortamında Verilerin Tutulması ve Uygulama Geliştirilmesi

Qfield Qgis yazılımında oluşturulan CBS projesinin tablet veya cep telefonunda sunumu, projeye veri girişi için kullanılacak bir araçtır.

Eğer Qgis yazılımındaki tabakalar uzak bilgisayardan çekiliyorsa, Qfield eklentisiyle cep telefonunda açılacak proje verisi uzak bilgisayardan gelecek, projedeki tabakaların sahalarına veriler eklendiğinde veri direkt olarak uzak bilgisayardaki tabloya kaydolacak. Bilgisayardaki Qgis uzak bilgisayardaki veri tabanına bağlandığında anlık olarak güncellenen veri görünecek.

Eğer Qgis yazılımındaki tabakalar Qfield bulut ortamına yükleniyorsa, Qfield eklentisiyle cep telefonunda açılacak proje verisi Qfield bulut ortamından gelecek, projedeki tabakaların sahalarına veriler eklendiğinde veri direkt olarak Qfield ortamında kayıtlı tabloya kaydolacak. Bilgisayardaki Qgis anlık olarak Qfield ortamına kaydedilen güncel veriyi göremez. Bunun için Qfield için oluşturulmuş projenin güncellenmesi gereklidir.

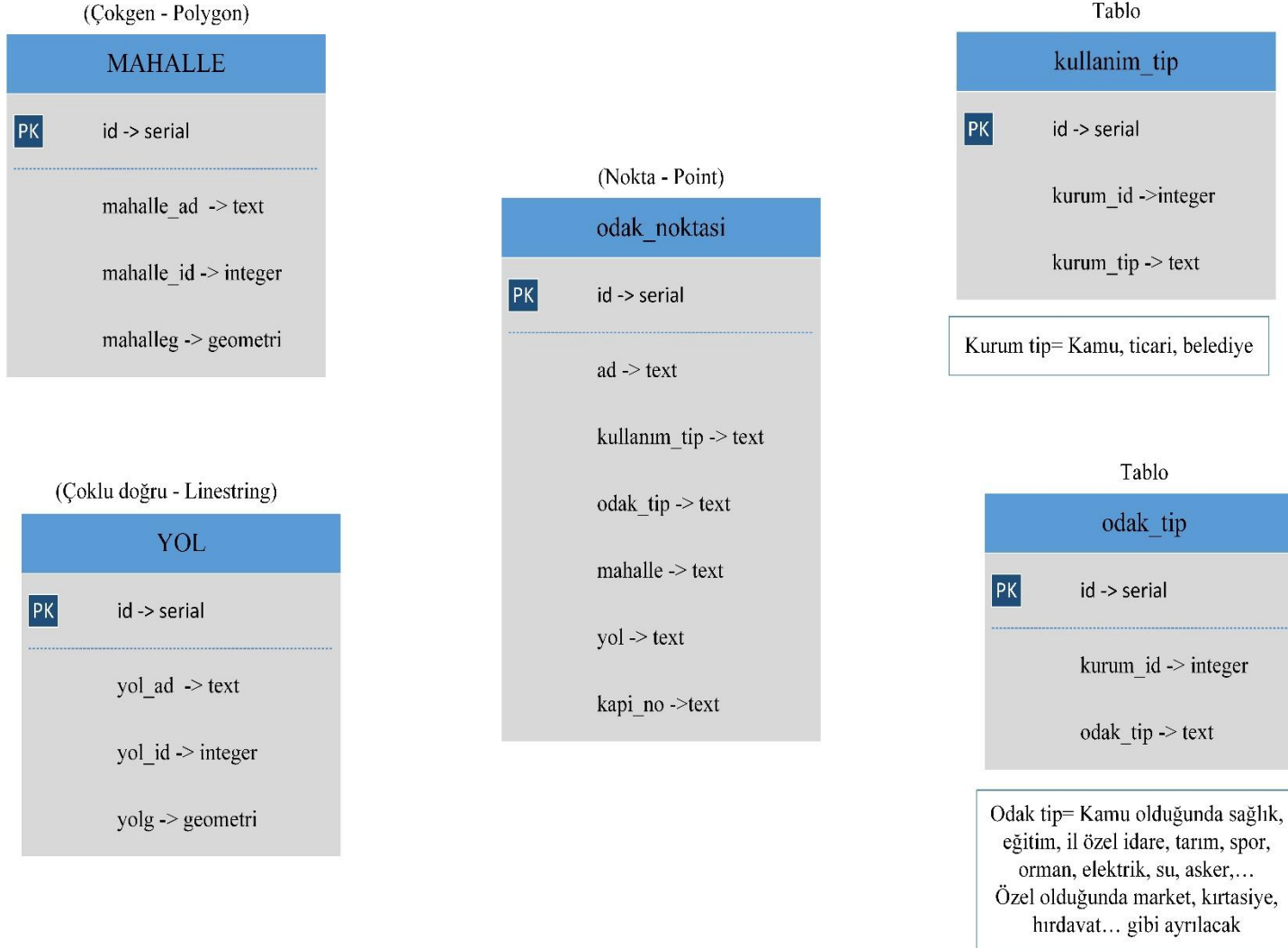
Dokümandaki anlatımda öğrencilerin saha çalışmalarında faydalanmaları için Qfield eklentisinin Bulut ortamının kullanımına dair anlatım yapılacaktır.

QGIS Yazılımında Proje Oluşturma

Qfield eklentisini kullanmadan önce Qgis yazılımında tablolar, tablolarda tutulacak sahaların, sahaların veri girişinin yapılma şekli, yapılacak sorgulamalar belirlenmelidir. Hazırlama sürecinden sonra Qgis yazılımında proje dosyası oluşturulduktan sonra Qfield eklentisi kullanıma başlanabilir.

Anlatımda kullanılacak örnekte, kent içindeki odak noktaları ve odak noktaları hakkında bilgi toplanması amaçlı çalışma planlanmıştır. Şekil 198 veri tabanı içindeki tablo tasarımı mevcuttur.

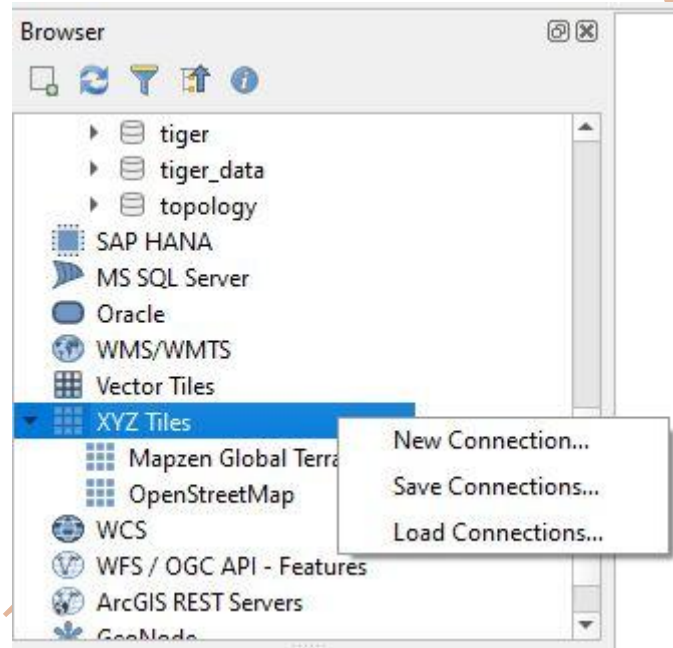
ODAK NOKTALARINA AİT VERİ TOPLAMA



Şekil 198

Qgis Yazılımında Altlık Harita Ekleme

Qfield da oluşturulacak haritaların dışında kullanıcıların harita kullanımı kolaylaştırmak, haritanın görselliğini arttırmak amacıyla altlık haritalar eklenebilir. Bu haritalar internet harita sağlayıcıları (openstreetmap, Google map, bing map, Yandex map,...) tarafından oluşturulmuş haritalar olabilir. Qgis ortamında bu haritaların eklenmesinin birden fazla yöntemi bulunmaktadır. Bu yöntemlerden bir tanesi XYZ tiles kısmında internet harita sağlayıcısına ait internet link adresinin eklenmesidir. İşlemin yapılabilmesi için Browser (tarayıcı) paneli içindeki XYZ Tiles ağaç yapısına sağ tuş ile tıklayarak New Connection (yeni bağlantı) alt menüsü seçilir (Şekil 199).



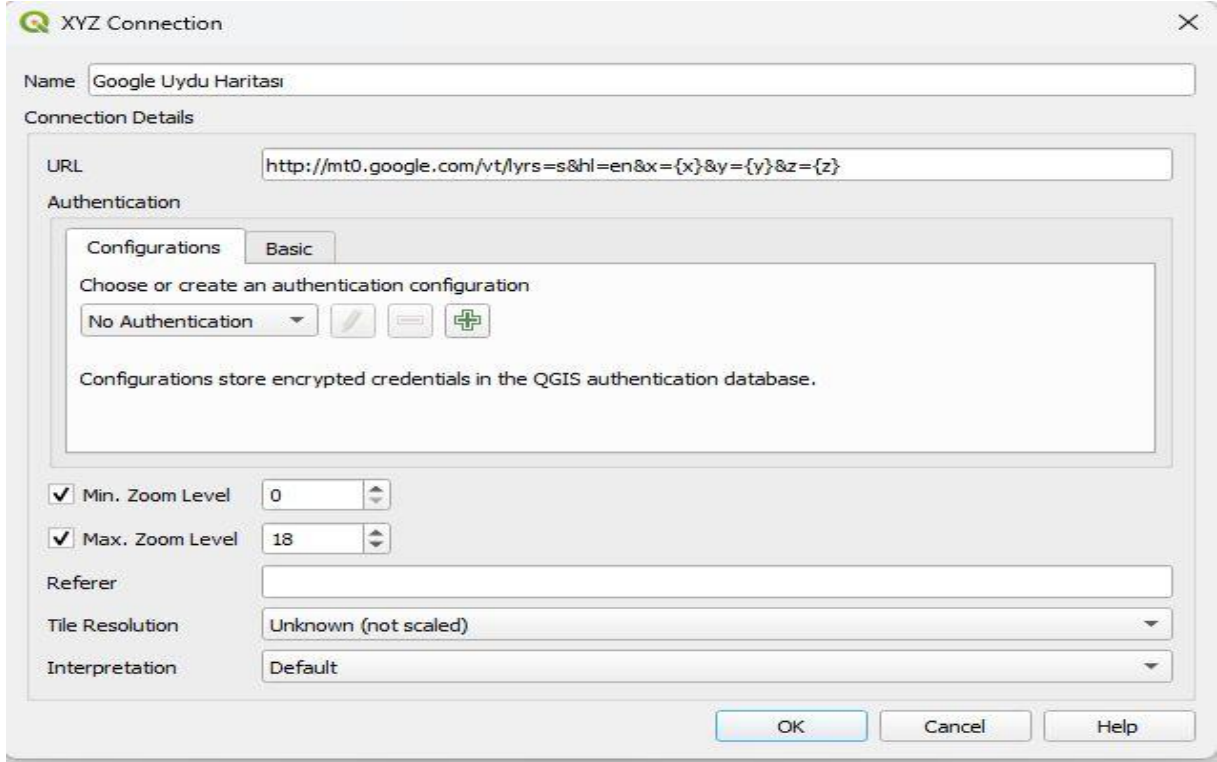
Şekil 199

Açılan XYZ Connection penceresinde

Name kısmına kullanıcının harita amacı için gireceği isim yazılır.

Url kısmında harita sağlayıcısına ait haritanın çağırılması için gereken internet adresi girilir. Aşağıda bazı internet harita sağlayıcılarına ait internet adresleri vardır.

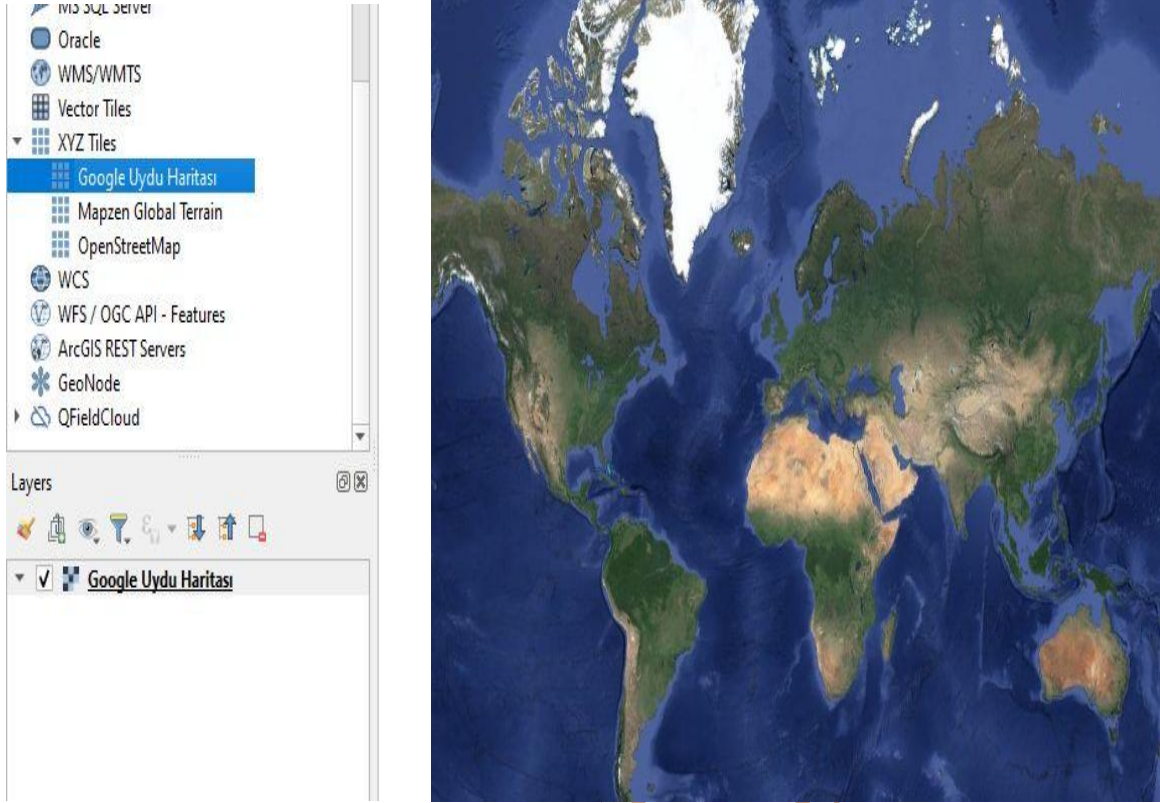
- Google uydu haritaları: <http://mt0.google.com/vt/lyrs=s&hl=en&x={x}&y={y}&z={z}>
- Google arazi haritaları: <http://mt0.google.com/vt/lyrs=p&hl=en&x={x}&y={y}&z={z}>
- Esri uydu haritaları:
https://server.arcgisonline.com/ArcGIS/rest/services/World_Imagery/MapServer/tile/{z}/{y}/{x}
- Bing uydu haritaları: <https://t0.tiles.virtualearth.net/tiles/a{q}.jpeg?g=685&mkt=en-us&n=z>
- Opentopo haritaları: <https://tile.opentopomap.org/{z}/{x}/{y}.png>



Şekil 200



Şekil 201 internet harita sağlayıcısı tarafından haritanın taban harita olarak eklenmesini temsil etmektedir. Harita internet tabanlı çalışır. Hem Qgis yazılımına taban harita olarak eklenebilmesi hem de Qfield eklentisinde çalışabilmesi için internet bağlantısının olması gereklidir.

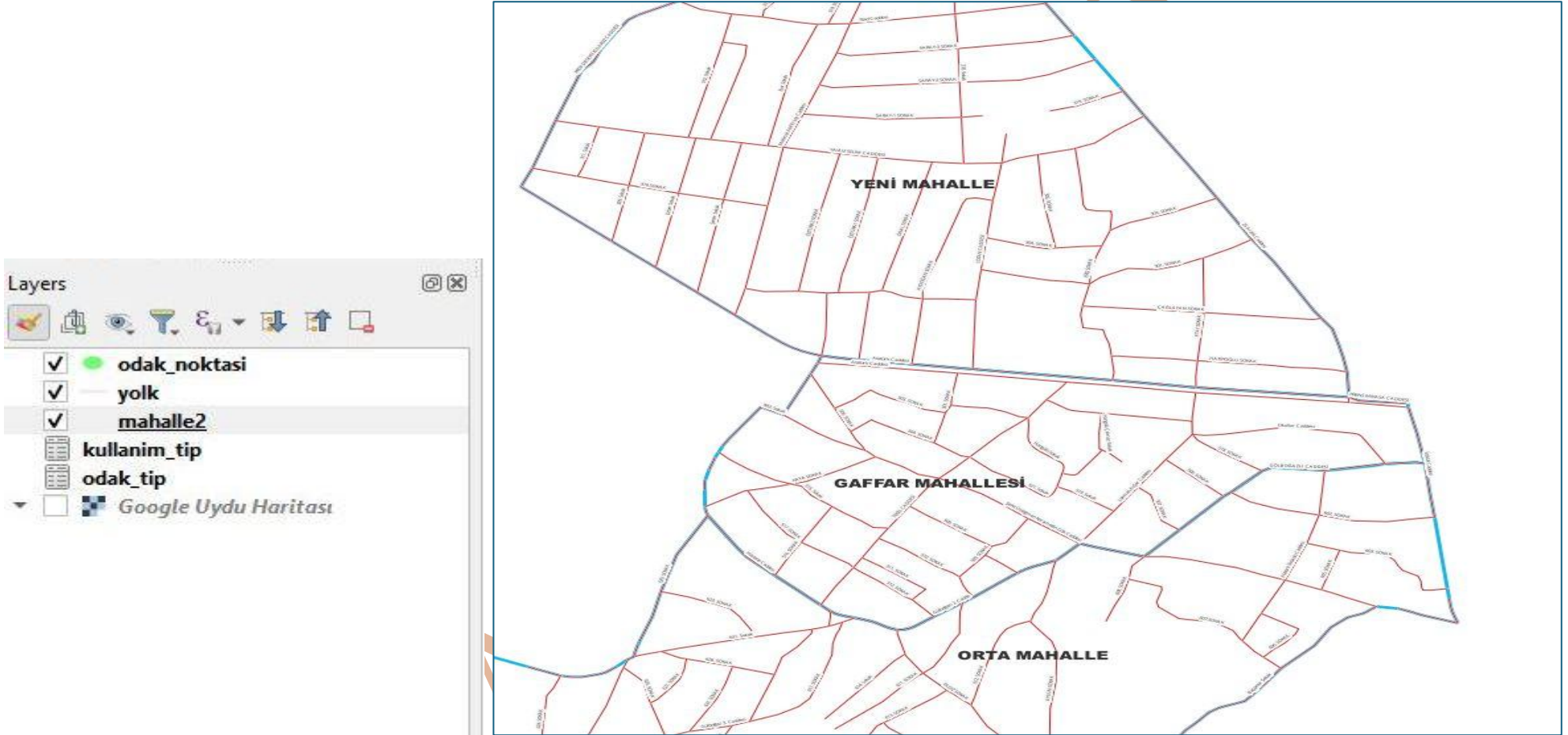


Şekil 201

Örnek proje için Şekil 198’de verilen tasarımda olduğu gibi tabaklar oluşturuldu. Amaç oluşturulan tabakaları proje formatında cep telefonu veya tablet teknolojileri üzerinden görebilmek, tabakaların sahalarına ait verilerin girilebilmesi, hatta verilerin girilmesi işlemlerinde cep telefonu veya tablet teknolojilerinin GNSS sinyal alıcılarının konum değerlerinin de kullanılması sağlamasıdır.

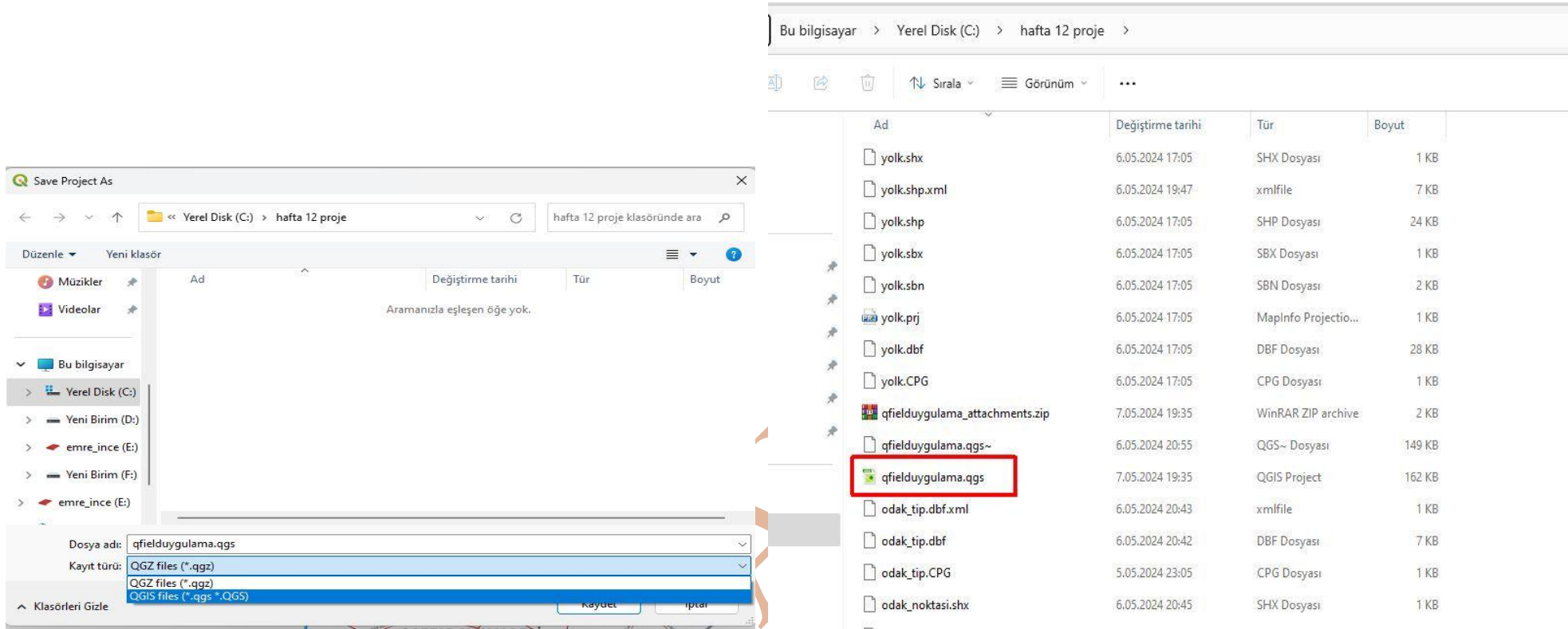
Projede Kullanılacak Tabakaların Oluşturulması ve Proje Dosyasının Oluşturulması

Tasarlanan tabakalar Qgis yazılımında oluşturuldu, tabakaların görünüm etiket stilleri düzenlendi (Şekil 202). Proje dosyası tabakaların saklandığı dosyaya kaydedildi. Şekil 203 proje dosyasının kayıt aşamasıdır. Qgis yazılımının proje dosya formatı qgs veya qgz olacak şekilde iki farklı türdür. Qgz formatında sıkıştırma işlemi uygulanır. Bu sebepten dolayı dosya formatı olarak qgs formatı seçildi. Proje dosyasına *qfielduygulama* adlı isim verildi (Şekil 203).



Şekil 202

Coğrafi Bilgi Sistemleri II



Şekil 203

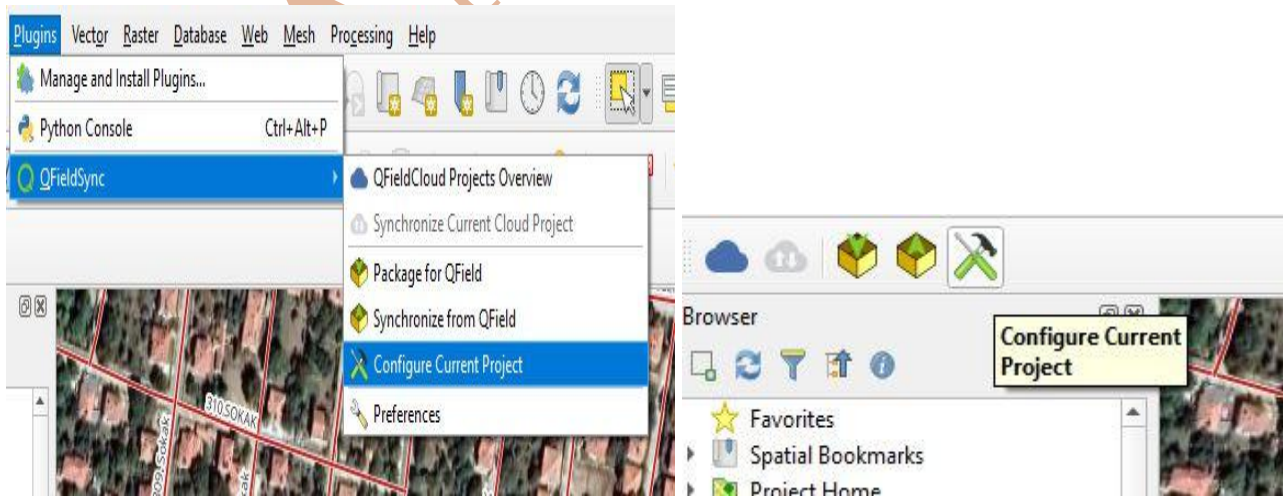
Projede sadece odak_noktasi tabakasına ait sahaların verilerinin girileceği tasarlandı. Diğer mahalle (mahalle2) verisinin olduğu tabaka ile yol (yolk) verisinin olduğu tabaka sadece altlık harita olarak kullanılacak tabakalar olarak tasarlandı. Mahalle ve yol verisinin olduğu tabakaların projeksiyonları 3857 olarak ayarlandı. Odak noktalarına ait verilerin girileceği odak_noktasi tabakasının projeksiyonu 4326 epsg kodu olacak şekilde ayarlandı. İki farklı projeksiyon gibi görülebilir. Bu farklılığın temel sebebi, mahalle ve yol verilerinin olduğu tabakalar Turef – TM33 projeksiyonunda mevcut olan tabakalardır.

Odak noktasının girileceği tabakanın konum bilgileri cep telefonunun GNSS sinyal alıcısının elde ettiği koordinatlar ile sağlanacak. Cep telefonundaki sinyal alıcılar elde ettikleri jeosantrik koordinat değerlerini jeodezik enlem ve jeodezik boylam olarak dönüştürür. Buldukları enlem boylam değerlerini projeksiyon yardımıyla ikinci kere dönüştürmezler. Cep telefonunda yapılacak veri girişleri 4326 epsg kodunda yani WGS -84 elipsoidini kullanarak jeodezik enlem ve jeodezik boylam olarak kayıt altında tutacaklar.

Mahalle ve yol tabakaları 5255 EPSG projeksiyonunda GRS-80 elipsoidinde harita düzlem koordinatlarında (sağa koordinat - Y ve yukarı koordinat - X) tutulurlar. 5255 EPSG projeksiyonundaki X – Y koordinatlarından 4326 EPSG projeksiyonundaki enlem – boylam koordinatlarına dönüştürüldüğünde haritadaki objelerde bozulmalar olacaktır. 5255 EPSG projeksiyonundaki Mahalle ve yol tabakaları, 3857 EPSG projeksiyonuna dönüştürülerek projeye eklenmiştir. 3857 EPSG projeksiyonunda nokta objelerinin koordinatları WGS-84 elipsoidinde X – Y koordinatlarında tutulur. Yani GRS – 80 elipsoidindeki X – Y düzlem koordinatları WGS – 84 elipsoidinde X – Y düzlem koordinatlara dönüştürüldüler.

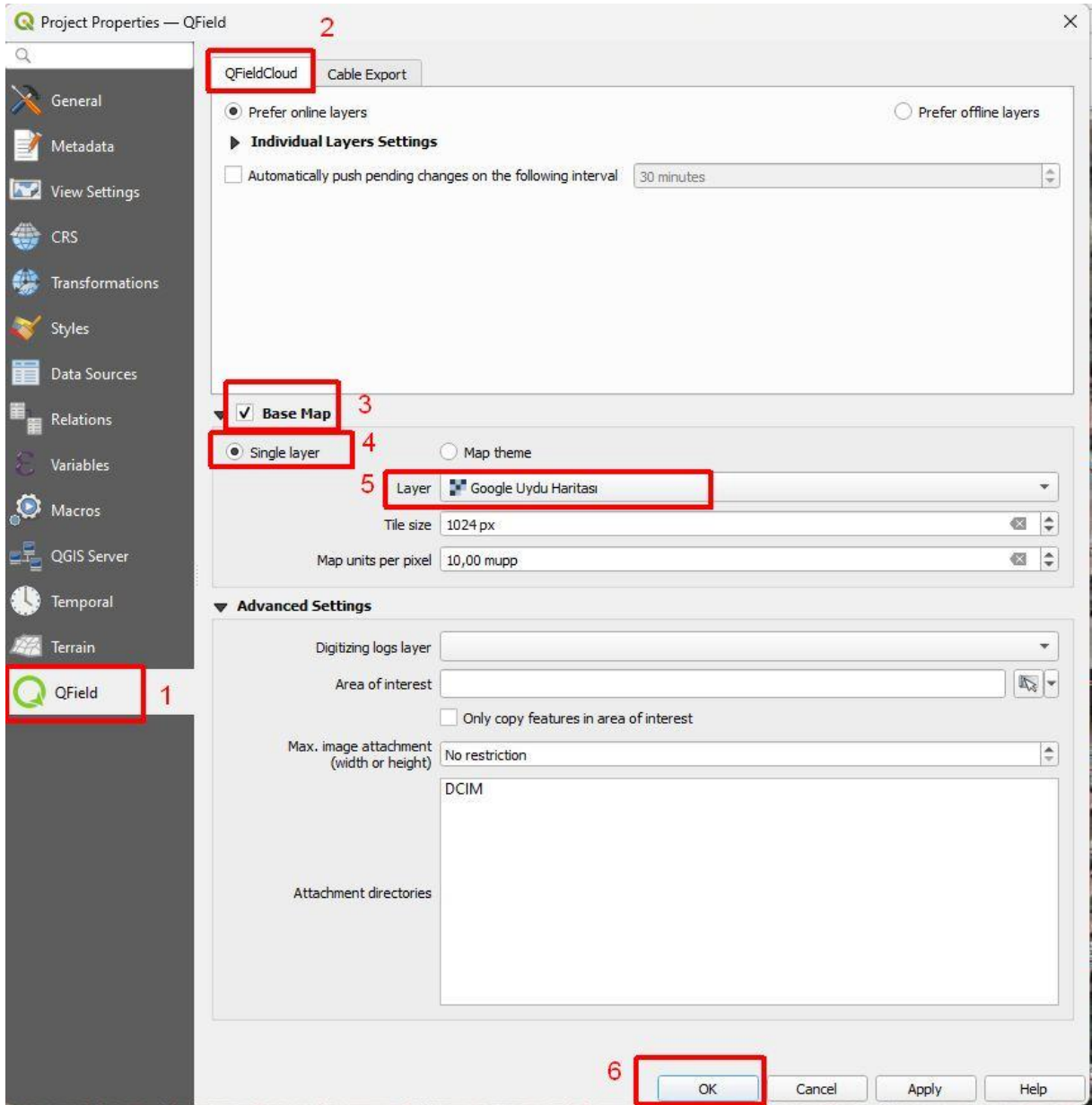
Projenin Qfield Bulut Ortamına Aktarılması İşlem Adımları

Hazırlanan tabakalar ve kaydedilen proje dosyası sonrasında Qfield proje ayarlarına girilip gereken ayarlamalar varsa yapılır. Örneğin projede temel harita olarak Google uydu haritaları temel harita olarak kullanılacak. Temel haritanın ayarlanması için Şekil 204 de görülen araç çubuğu veya alt menü kullanılabilir. Geçici olarak sadece temel haritanın ayarlanması aşaması gösterilecek onun dışında yapılacak ayarlamalar konu için de detaylı anlatılacaktır.



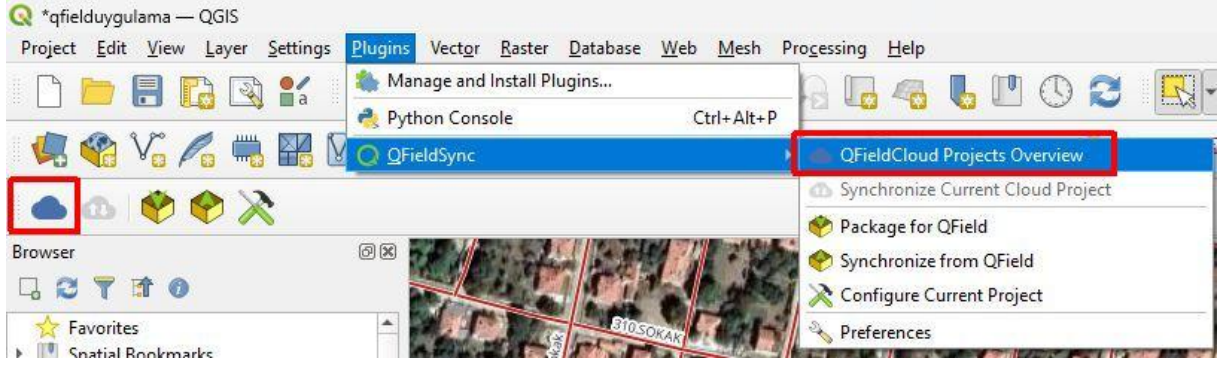
Şekil 204

Şekil 205 temel kullanılacak haritanın ayarlanması işlem adımlarının tasviridir.



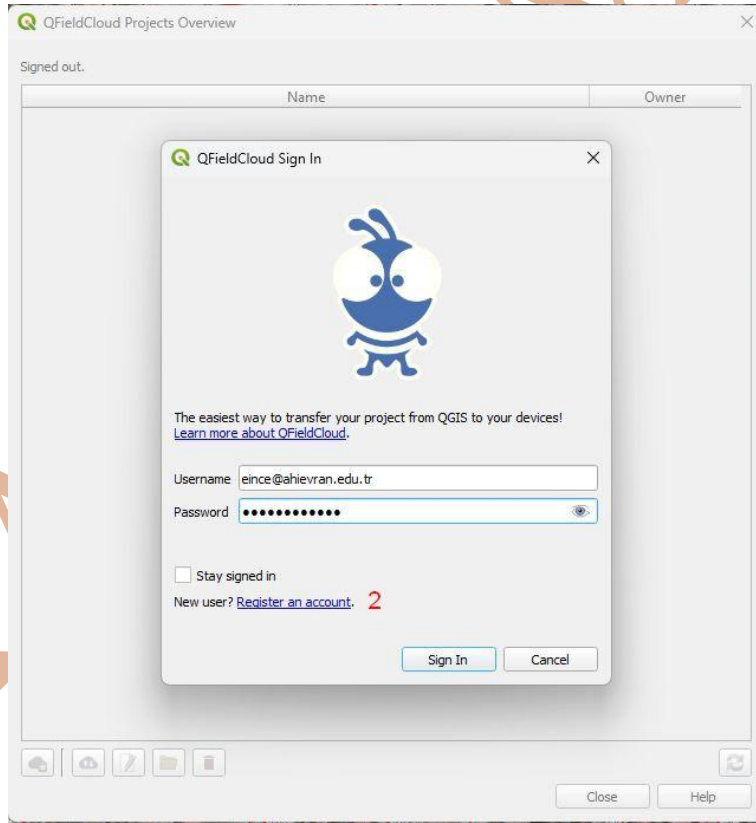
Şekil 205

Projenin Qfield Cloud ortamına aktarılması için QfieldCloud Projects Overview alt menüsü veya araç çubuğundaki ilgili düğme seçilir (Şekil 206).



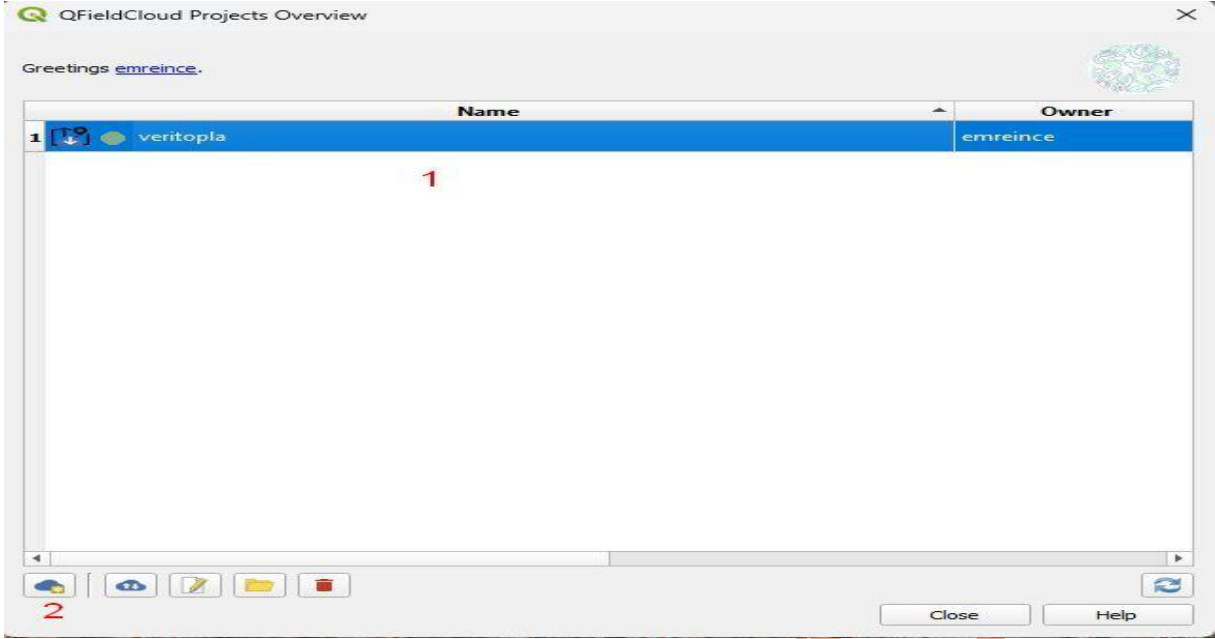
Şekil 206

Qfield Cloud ortamında proje oluşturabilmek için ilk önce siteme kayıtlı bir kullanıcı olmalısınız. İşlemlerin yapılması için internet bağlantınızın olması gereklidir. Qgis yazılımından projeyi göndereceğimiz için ilk kayıt işlemi masaüstü veya dizüstü bilgisayarınızda kullandığınız Qgis yazılımında Şekil 206'da gösterilen alt menüden ulaştığımız Şekil 207'de görünen penceredeki 2 numaralı kısımdan kayıt işlemlerini yapmalısınız.



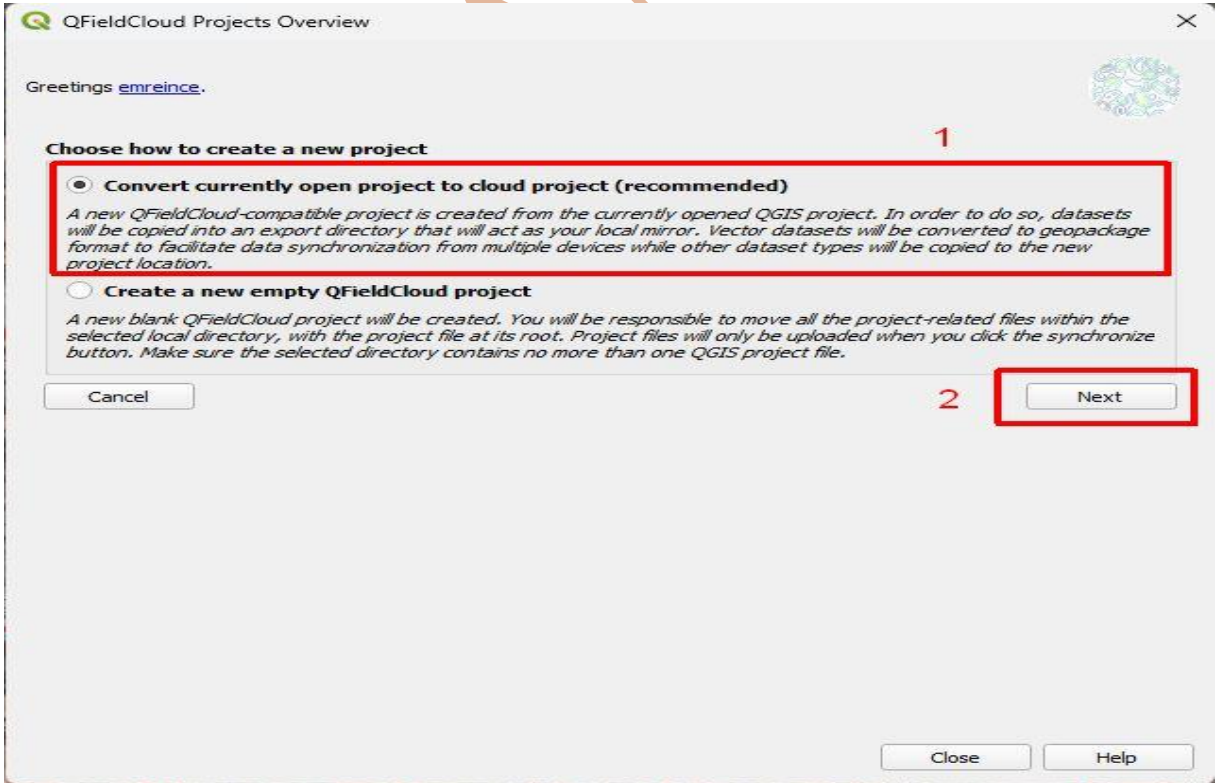
Şekil 207

Bağlantı sağlandıktan sonra Şekil 208'de görünen pencere gelir. Penceredeki 1 numaralı alan daha önce eklenmiş projeleri gösterir. 2 numaralı düğme yeni proje eklemek için kullanacaktır.



Şekil 208

Yeni proje eklemek için Şekil 209’da görülen pencere açılır. 1 numaralı seçenekte Qgis yazılımında tabakaların olduğu projenin Qfield Cloud a aktarılmasında işleminde kullanılır. Diğer radyo düğmesi boş bir projeye bir dosyadan aktarım için kullanılır. 1 numaralı seçenek seçildikten sonra Next tuşuna basılır.



Şekil 209

Coğrafi Bilgi Sistemleri II

Açılan son projede Name kısmı (1 numaralı alan) Qgis yazılımında proje ismi ile aynı olacak şekilde otomatik oluşur. 2 numaralı kısımda ise kullanıcının proje için ek gireceği tanım kısmıdır. 3 numaralı kısım bilgisayarınızda oluşturulacak ve dosyanızın kaydedileceği dizindir. 4 numaralı düğme ile projeniz Qfield bulut ortamında oluşur. Artık projenizi cep telefonunuz veya tabletinize yükleyeceğiniz Qfield uygulamasında açabilirsiniz.

QFieldCloud Projects Overview

Greetings [emreince](#).

Project Details

Name: qfielduygulama **1**

Description: Kaman ilçesinde ilgi odaklarına ait verilerin toplanması için yapılan Qfield projesi. **2**

Owner: emreince

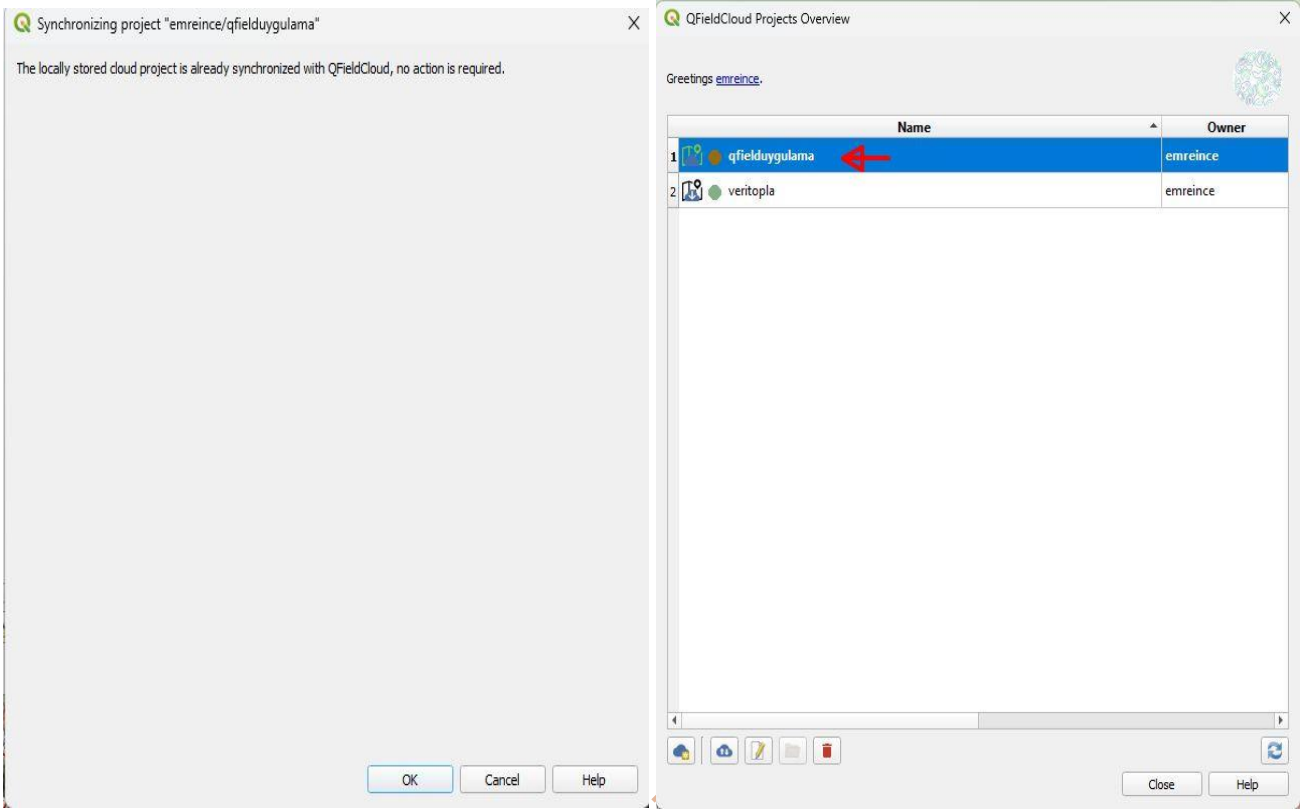
Local Project Settings

Local Directory: C:\Users\incee\QField\cloud\qfielduygulama **3**

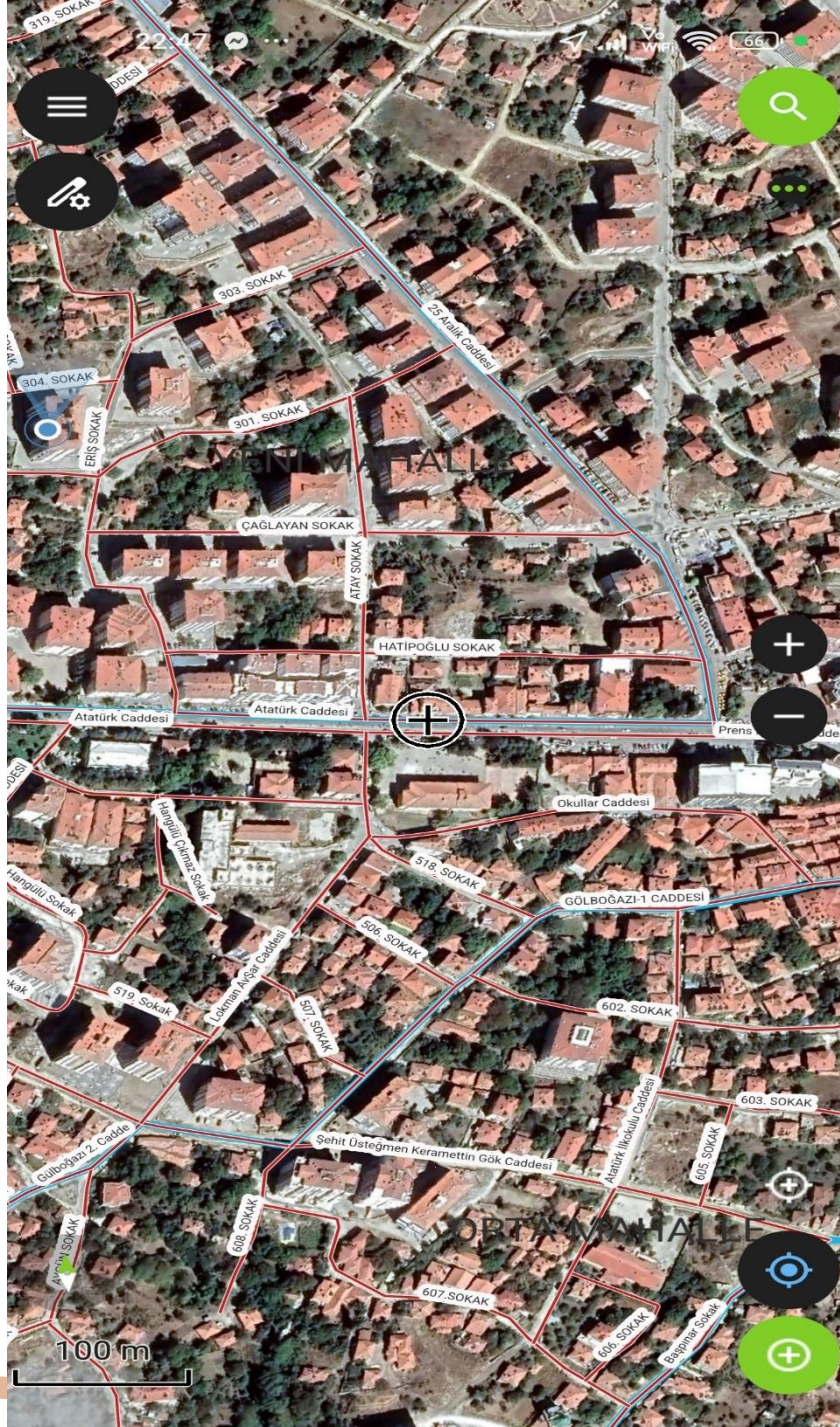
Back **4** Create

Close Help

Coğrafi Bilgi Sistemleri II

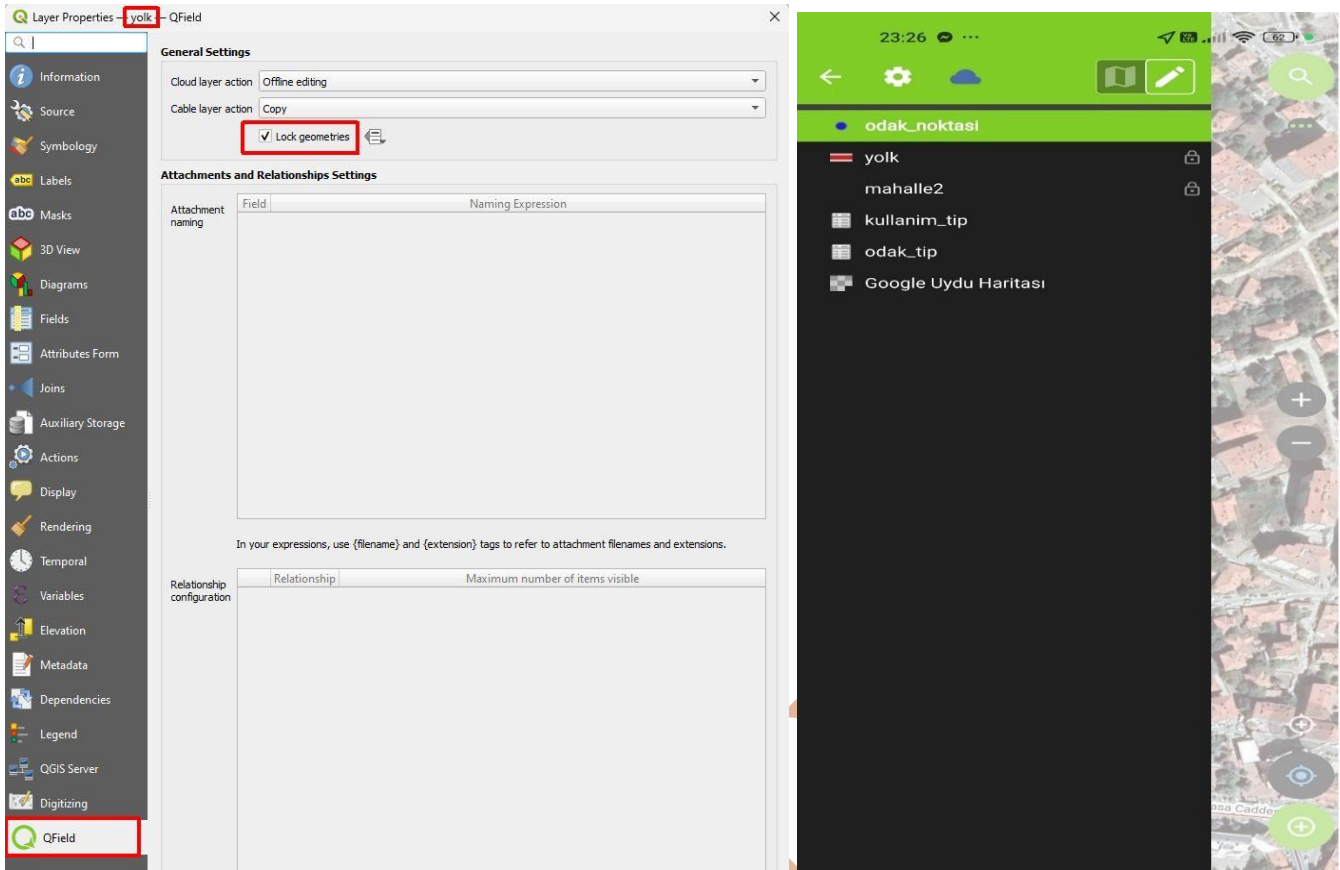


KAMAN



Qgis Katmanların Düzenlenebilir Olma Ya Da Kilitli Olma Ayarları

Qgis içinde hazırlanmış Proje içindeki her bir katman düzenlenebilir olmayabilir. Eğer projeye eklenecek katman sadece altlık harita olarak kullanılacak düzenlenmeyecek ise tabakanın özelliklerine girdikten sonra Qfield sekmesinde Lock Geometries seçeneği seçildiğinde Qfield projesinde tabakaların yanında kilit işaretleri oluşur (Şekil 210).

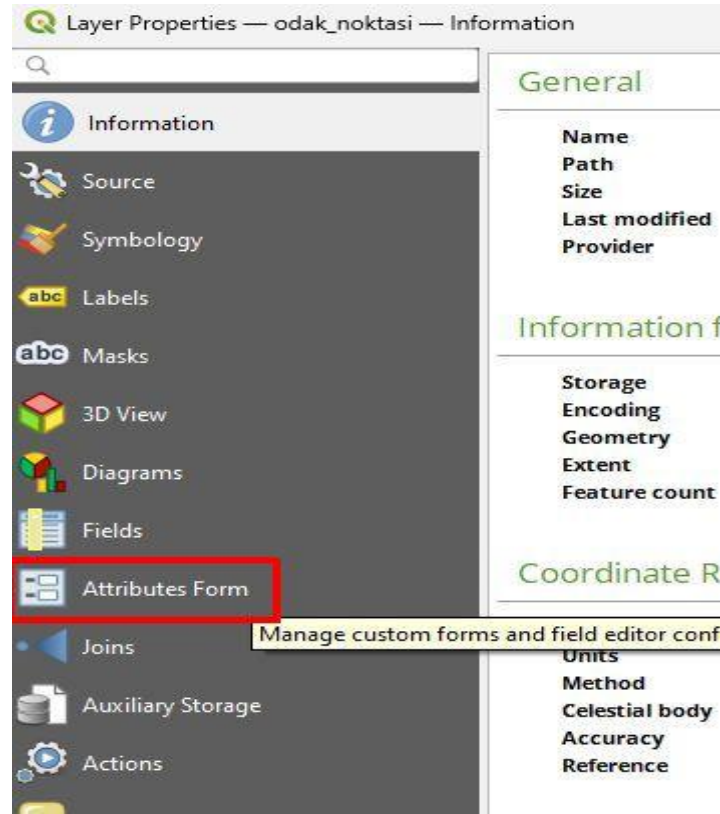


Şekil 210

Qfield Yazılımında Kullanılacak Tabakanın Veri Girişlerini Düzenleme

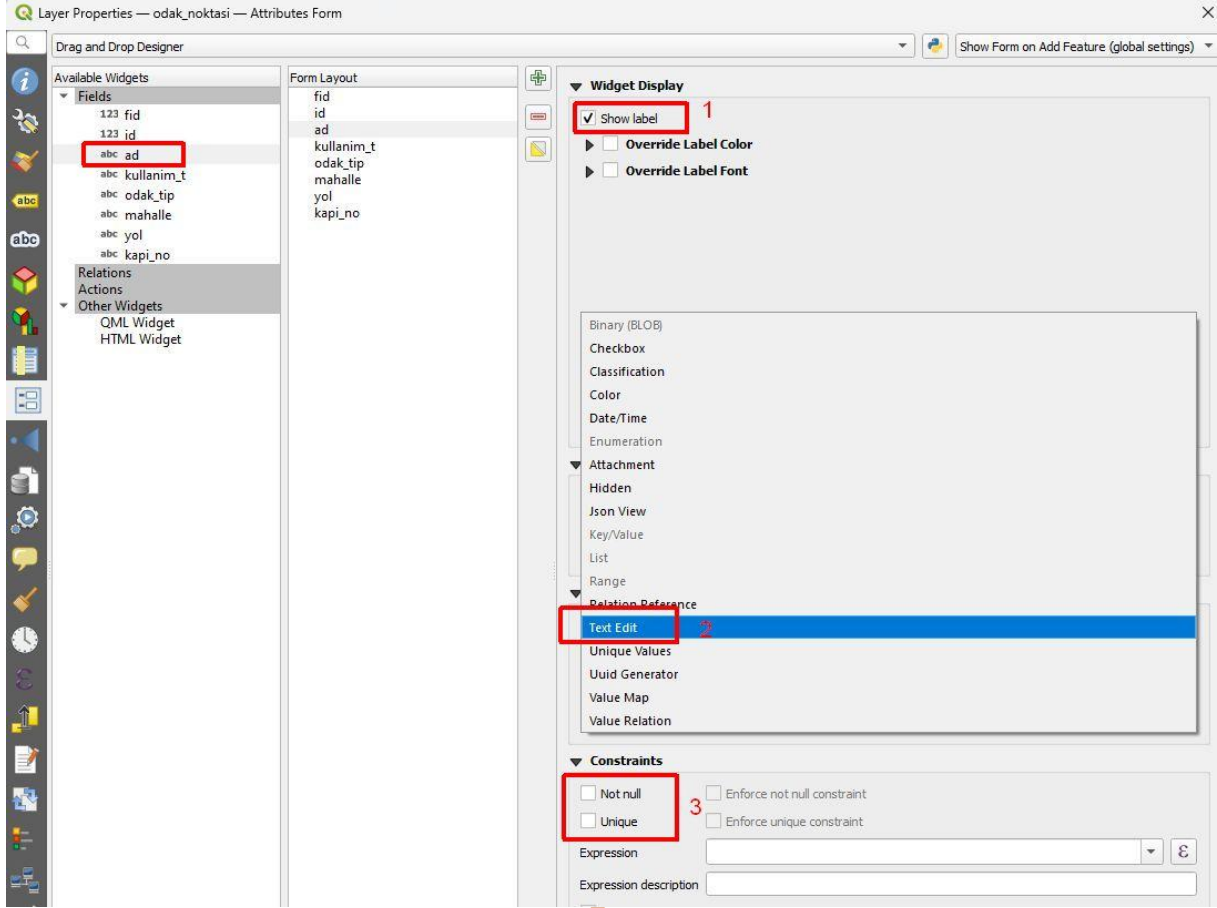
Qfield projesinde düzenlenebilir olan tabaka içinde birden fazla sahaya veri girişine dair düzenlemeler yapılabilir. Bu işlemin yapılabilmesi için tabakanın özellikler kısmına girdikten sonra *Attribute Forms* (Öznelik Formları) sekmesinde saha veri girişi ayarlamaları yapılabilir (Şekil 211).

Kullanıcının veri girişinin bir form üzerinden sağlanması, hem veri girişinin kolaylığını sağlayacak, verilerde oluşacak hatanın azalmasını sağlayacak, bazı bilgilere ait verilerin otomatik doldurulmasını sağlayacak.



Şekil 211

Şekil 212 ad isimli sahanın veri metin kutusu olarak seçilmesine dair gösterim var. 1 numaralı kısımda form üzerinde görünmesi veya tercihen sahaya ait veri girişinin gözükmemesi için onay kutusu görülüyor (Show Label). 2 numaralı kısımda veri girişinin yapılma şekli seçilmiş. 3 numaralı kısımda bir seçim yapılmamış. Not null seçeneği seçilirse, form üzerinde veri girişi sırasında verinin boş bırakılmayacağı ibaresi belirecek. Unique seçilirse, verinin kayıtlar içinde tekil olacak şekilde girilmesi gerektiği, tekrarlı verinin girilemeyeceği kontrol edilip kullanıcı uyarılacak.

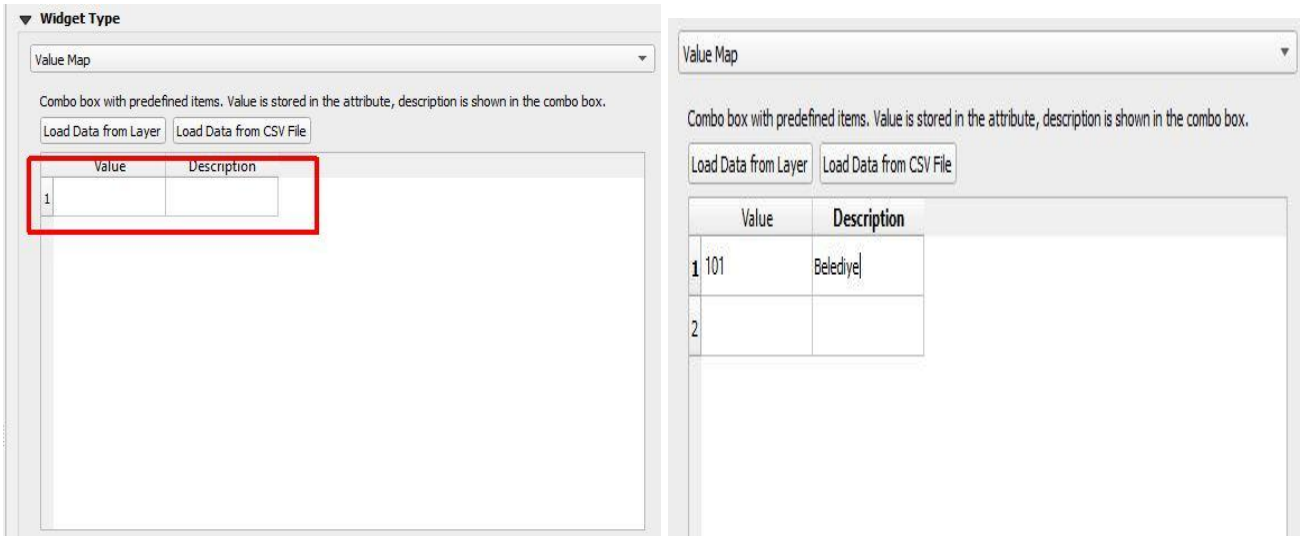


Şekil 212

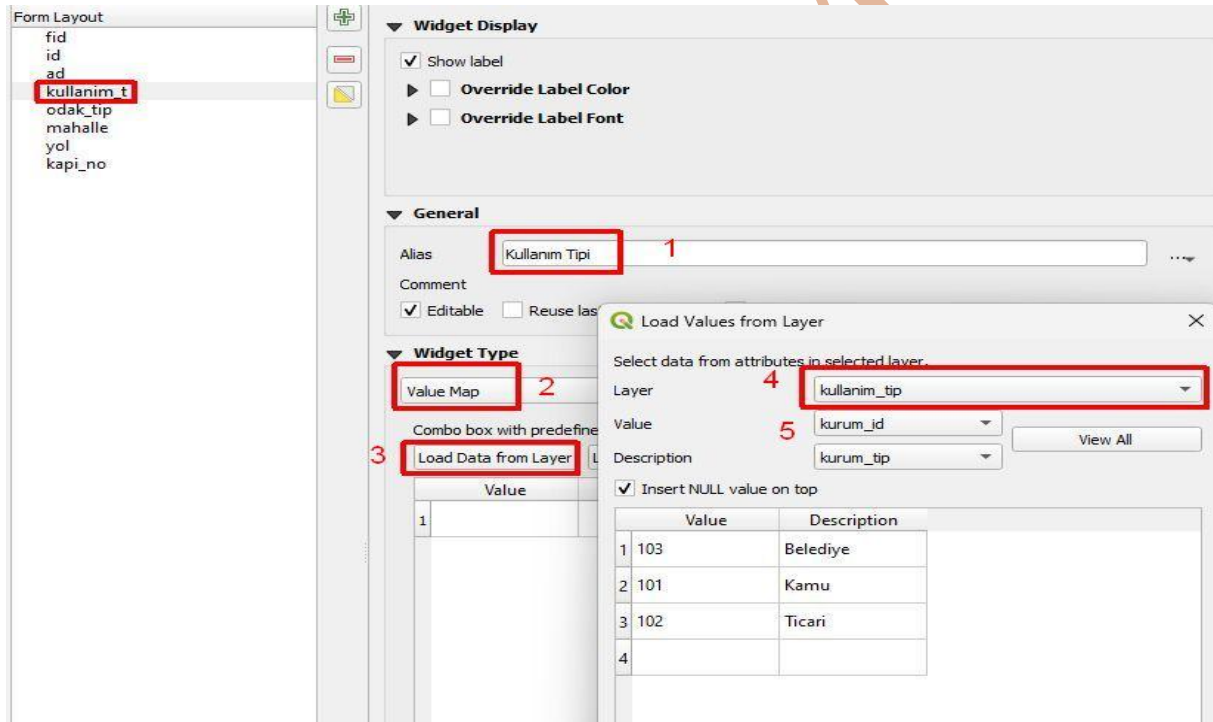
Verinin Seçim Yoluyla Girilmesinin sağlanması

Girilmesi sabit olan veri grubu olabilir. Şekil 214 bu veri tipine bir örnek verilmiştir. Örnekte kurum tipi olarak belediye, kamu veya ticari adlarında sabit veriler girilecektir. Bu sabit verilerin elle girilmesinde kullanıcının hata yapma olasılığı bulunmaktadır. Verinin elle girilmesi yerine Şekil 213’de olduğu gibi tablo doldurulup veri girişinde kullanıcının seçim yapması sağlanabilir.

Diğer bir yöntem Şekil 214’de olduğu gibi Qgis yazılımında var olan bir tablodan veri okutulabilir. 3 numaralı kısımdaki Load Data From Layer (veriyi tabakadan yükle) düğmesine tıkladığında açılacak yeni pencerede 4 numaralı kısımda tabaka seçilir, 5 numaralı kısımda Value seçeneği girilecek verinin değerinin hangi sahadan seçileceği belirtilmesini sağlar. Description kısmı ise verinin seçim esnasında görünmesi için gerekli sahanın seçiminin yapılacağı kısımdır. Örneğin (Şekil 214 numaralı resme göre) seçim esnasında Kamu görünürken kayıt esnasında 101 verisi kaydedilecektir.



Şekil 213

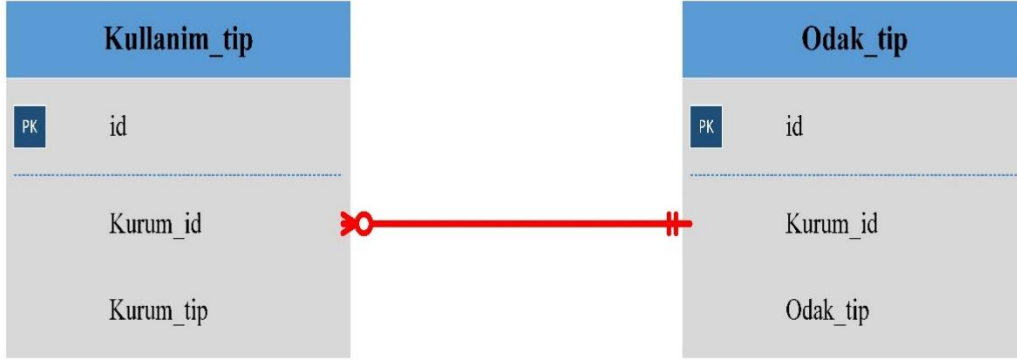


Şekil 214

Verinin Tablolar arasındaki İlişki ile Otomatik Girilmesi

Kullanıcının veri girişini kontrol ederken, birbirleriyle ilişkili olan tablolardaki veri girişinde kullanıcıya kolaylık sağlanabilir. Bu işlemin gerçekleşebilmesi için verinin çekileceği tablolarda ortak verinin olduğu (ilişkinin sağlanacağı) sahalar olmalıdır.

Şekil 215 birbirleriyle ilişki kurulabilecek tablolara örnektir. *Kullanım_tip* tabakasında ki *Kurum_id* sahası ile *Odak_tip* tablosundaki *Kurum_id* sahasları veri olarak (isimlerinin aynı olmasından bahsedilmiyor) birbirleriyle aynıdır.



Şekil 215

Bu işlemin sağlanması için Kullanım_tip sahasında Widget Type (araç tipi) olarak Value Relation seçilmeli. Verinin listelenmesi için kullanılacak tabaka olan Kullanım_tip sadece sözel verini tutulduğu tablo seçilmeli. Key column sahası olarak her iki tabloda da aynı olan Kurum_id sahası seçilmeli. Value column olarak seçim yapılacak olan saha seçilmelidir (Şekil 216 sol resimde ayarlamalar gösterilmiştir).

İlişki kurulacak olan saha odak_tip sahasıdır. Odak_tip sahası için de araç tipi olarak Value Relation seçilmelidir. Odak_tip sahasında da verilerin çekileceği tablo Layer kısmında seçilmeli. İlişkinin kurulacağı saha Key Column kısmında seçilmeli. Value Column kısmında Odak_tip sahasından seçilecek verilerin olduğu saha seçilmelidir.



Kurum_id sahasından seçilecek veriye göre Odak_tip sahasında verilerin otomatik doldurulması için Odak_tip için yapılan ayarlamalarda Filter Expression kısmında bir sorgu cümlecği yazılacaktır

"kurum_id" = current_value('kullanim_t')

Bu cümlecikteki current_value('kullanim_t') kısmıyla kullanim_t sahasından yapılan seçime göre ilişkili verinin çekimi sağlanmaktadır. Şekil 217 yapılan işleme örnek seçimleri göstermektedir.

▼ **Widget Type**

Value Relation

Select layer, key column and value column

Layer

Key column

Value column

Description column

Allow NULL value

Order by value

Allow multiple selections

Number of columns

Use completer

Filter expression

▼ **Widget Type**

Value Relation

Select layer, key column and value column

Layer

Key column

Value column

Description column

Allow NULL value

Order by value

Allow multiple selections

Number of columns

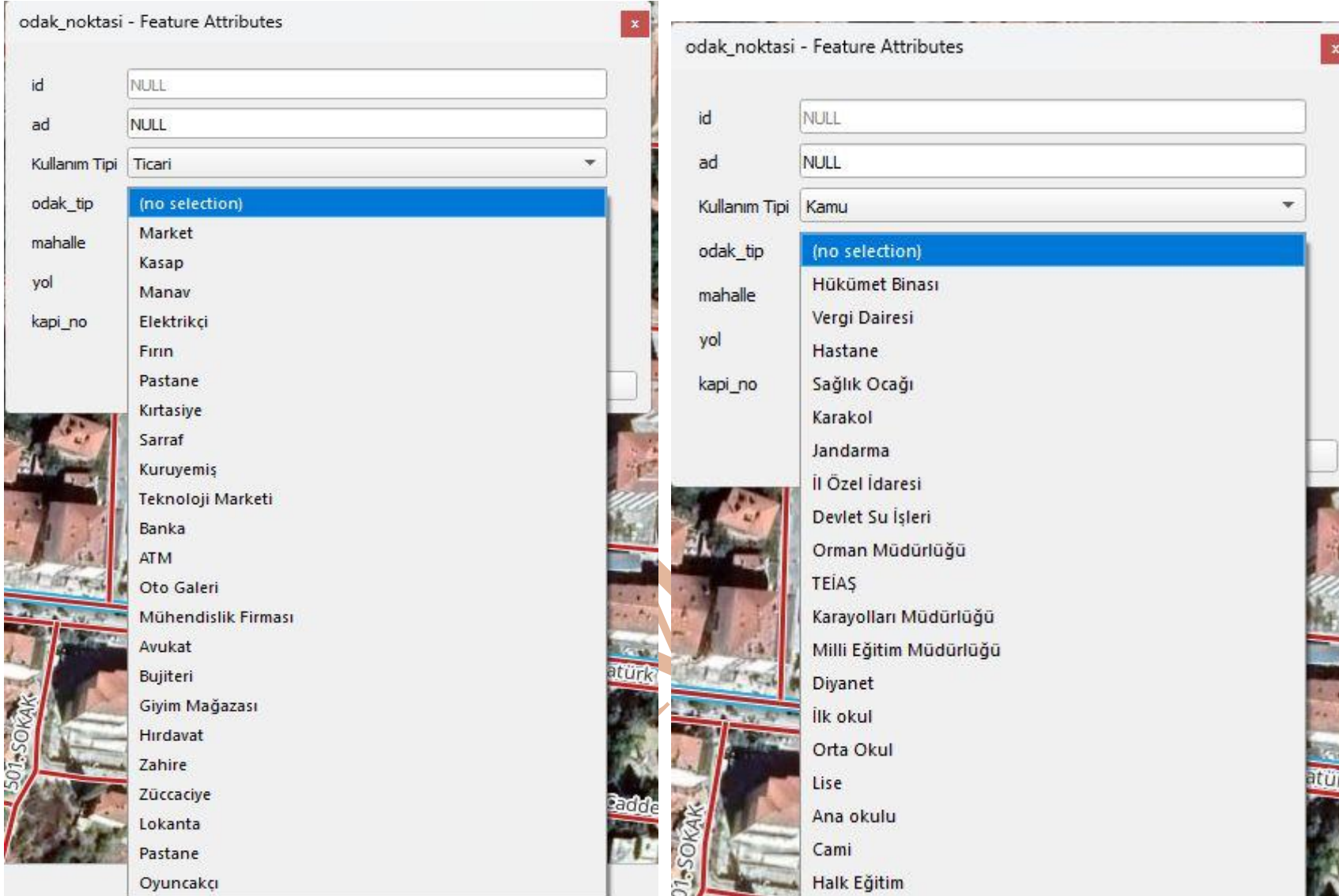
Use completer

Filter expression

```
"kurum_id"=current_value('kullanim_t')
```

Şekil 216

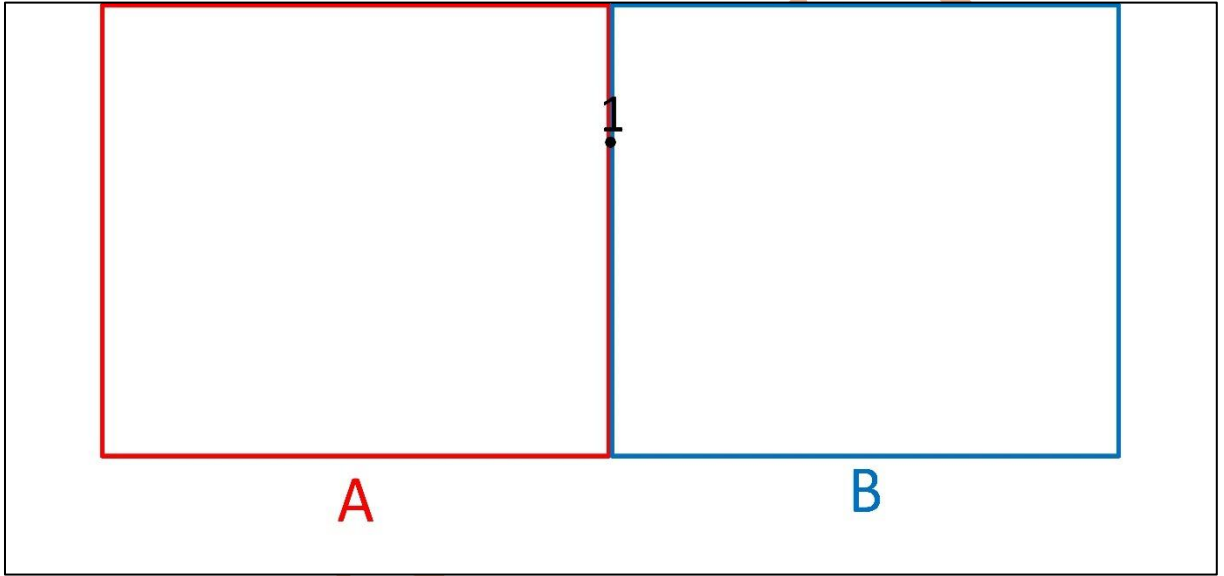
Coğrafi Bilgi Sistemleri II



Şekil 217

Harita Üzerinde Nokta Eklenmesiyle Tabakadan Çekilen Verinin Forma Eklenmesi (Kesişim Analizi)

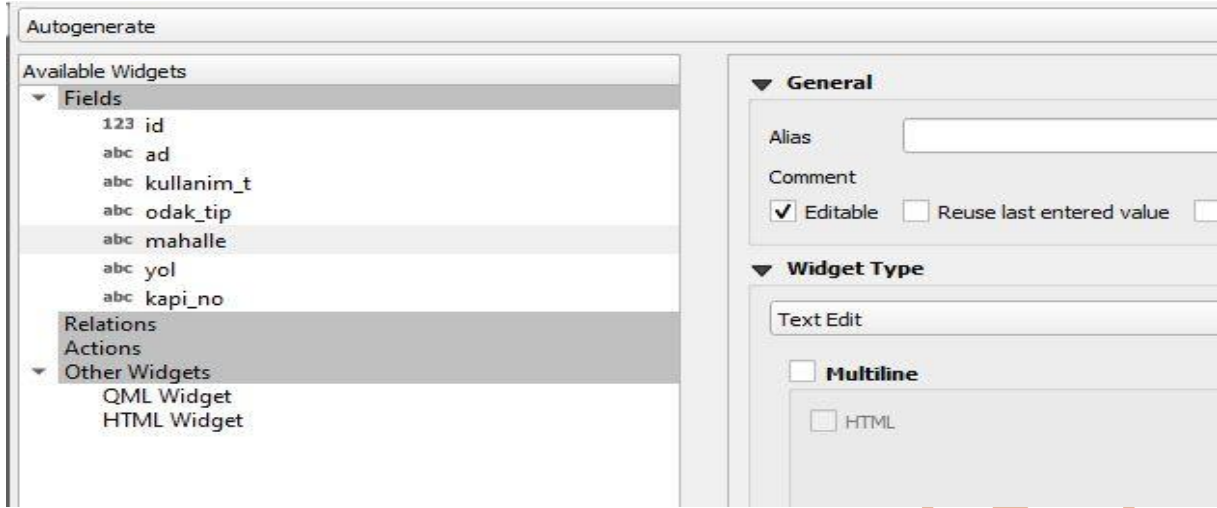
Uygulamada kullanıcının nokta eklemek için imleci harita üzerinde istediği konuma getirdikten sonra kayıt tuşuna basmasıyla noktanın içinde kaldığı mahalle bilgisinin otomatik eklenmesi istenmektedir. Bu işlemin gerçekleşmesi için birden fazla coğrafi bilgi sistemi kapsamındaki fonksiyonlardan birini kullanabilir. Örneğin bindirme (overlay) işlemlerinden kesişme (intersect), kapsama (contain), belirli bir mesafe kapsamında içinde kalma (distance with in) analizleri veya yakınlık (proximity) işlemlerinden en yakın objeyi bulma (nearest) analizi kullanılabilir. Contain analizi bu işlem için riskli olabilir. Çünkü Contain analizinde kaydedilecek nokta mahalle sınırı üzerinde kalırsa (komşu mahallelerin sınırları ortaktır) kapsama analizi olumsuz cevap verebilir (Şekil 218).



Şekil 218

Objenin içinde kaldığını tespiti için kesişim ve belirli mesafe içinde kalması analizleri tespit için en uygun analizler. İşlemin yapılmasında intersect (kesişim) analizi kullanıldı.

Noktanın ekleneceği tablonun özelliklerine girilip öznitelik form sekmesinde mahalle bilgisinin kaydedileceği saha hem düzenlenebilir hem de metin kutusu olarak işaretlenmeli (Şekil 219).

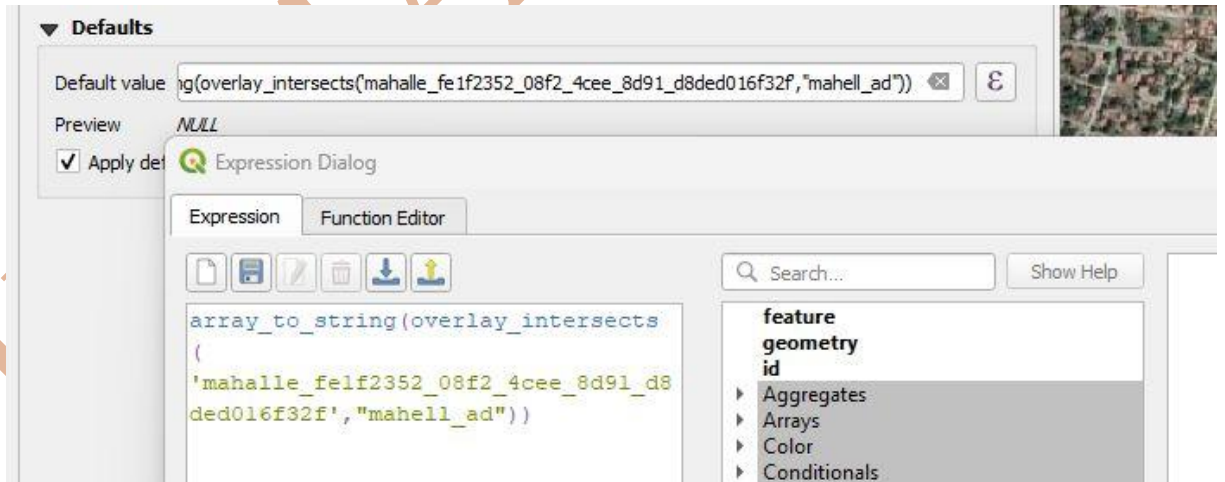


Şekil 219

Aynı pencere içinde Defaults kısmında default value değerini ifade dialog aracını kullanarak girildi (Şekil 220, Şekil 221). ifade 92 yazılan kesişim fonksiyonuna ait metin ifadesidir. İfadenin sonucunun çıktısının aktarımında `array_to_string()` fonksiyonu kullanıldı.

ifade 92

`array_to_string(overlay_intersects('mahalle_fe1f2352_08f2_4cee_8d91_d8ded016f32f',
"mahell_ad"))`



Şekil 220



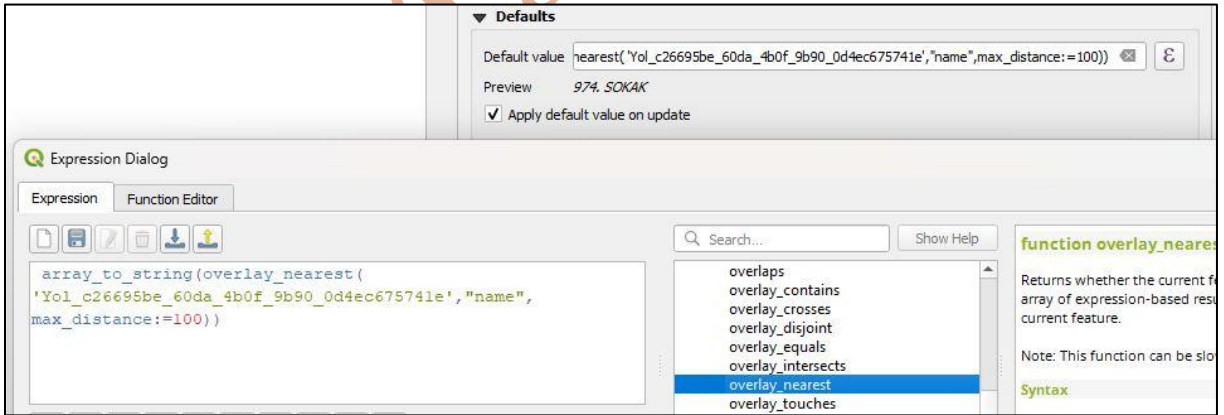
Şekil 221

Harita Üzerinde Nokta Eklenmesiyle Tabakadan Çekilen Verinin Forma Eklenmesi (En Yakın Obje Analizi)

Çalışmada noktaya en yakın yol objesinin adının form üzerine otomatik aktarılması istenmektedir. Bu işlemin yapılması için nearest fonksiyonu kullanılacak (Şekil 222).

ifade 93

```
array_to_string(overlay_nearest('Yol_c26695be_60da_4b0f_9b90_0d4ec675741e', "name",
max_distance:=100))
```



Şekil 222

Yapılan Örneklerde Kullanılan SQL Kodları**ifade 55 Örneği**

“alan_tm” tablosunun oluşturulması:

```
Create table alan_tm( id serial not null primary key, alan_ad text, geom_a
geometry('polygon',5255));
```

“alan_tm” tablosuna “ada2” adlı alan grafik objesinin eklenmesi:

```
insert into alan_tm(alan_ad,geom_a) values('ada2',
      st_makepolygon(
        st_makeline(
          array[
            st_setsrid(st_makepoint(562058.970888,4358857.670320),5255)),
            st_setsrid(st_makepoint(562010.331279,4358866.334030),5255)),
            st_setsrid(st_makepoint(562027.112028,4358918.427290),5255)),
            st_setsrid(st_makepoint(562044.024432,4358928.814340),5255)),
            st_setsrid(st_makepoint(562105.298278,4358954.192740),5255)),
            st_setsrid(st_makepoint(562129.120769,4358914.000980),5255)),
            st_setsrid(st_makepoint(562140.172595,4358884.123530),5255)),
            st_setsrid(st_makepoint(562140.102725,4358876.031240),5255)),
            st_setsrid(st_makepoint(562102.455379,4358860.889320),5255)),
            st_setsrid(st_makepoint(562092.953176,4358856.673620),5255)),
            st_setsrid(st_makepoint(562058.970888,4358857.670320),5255))
          ]
        )
      )
);
```

“alan_tm” tablosuna “ada3” adlı alan grafik objesinin eklenmesi:

```
insert into alan_tm(alan_ad,geom_a) values('ada2',
      st_makepolygon(
        st_makeline(
          array[
            st_setsrid(st_makepoint(562123.873391,4358933.986590),5255)),
            st_setsrid(st_makepoint(562112.780348,4358955.153450),5255)),
            st_setsrid(st_makepoint(562158.778276,4358977.813740),5255)),
          ]
        )
      )
);
```



```

st_setsrid(st_makepoint(562200.099473,4358912.258940),5255)),
st_setsrid(st_makepoint(562150.160697,4358881.600020),5255)),
st_setsrid(st_makepoint(562140.243847,4358905.757200),5255)),
st_setsrid(st_makepoint(562123.873391,4358933.986590),5255))
]
)
)
);

```

ifade 56 Örneği:

```

create table nokta_tm(
    id serial not null primary key,
    nokta_ad text,
    geom_n geometry('point',5255)
);
insert into nokta_tm(nokta_ad,geom_n) values(
    'çarşı camii', st_setsrid(st_makepoint(562447.449908,4358616.938430),5255)
);
insert into nokta_tm(nokta_ad,geom_n) values(
    'savcılı taksi', st_setsrid( st_makepoint( 562436.10175447, 4358469.68571878) , 5255)
);

```

ifade 57 Örneği:

```

insert into çizgi_tm(cizgi_ad,geom_c) values('yazar cad.',
    st_makeline(
    array[
    st_setsrid(st_makepoint(562324.82564915,4358766.64590441),5255),
    st_setsrid(st_makepoint(562324.82564915,4358766.64590441),5255),
    st_setsrid(st_makepoint(562324.82564915,4358766.64590441),5255),
    st_setsrid(st_makepoint(562324.82564915,4358766.64590441),5255),
    st_setsrid(st_makepoint(562324.82564915,4358766.64590441),5255),
    st_setsrid(st_makepoint(562324.82564915,4358766.64590441),5255)
    ]));

```

```
insert          into          nokta_tm(nokta_ad,geom_n)
values('P.49',st_geomfromtext('point(560126.974 4358347.657)',5255));
insert          into          nokta_tm(nokta_ad,geom_n)
values('P.2',st_setsrid(st_makepoint(560068.217,4358273.032),5255));
```

KAMANMYO

KAYNAKÇA

- BAYKAL, O. (2009). *Mühendislik Ölçmeleri-I*. İstanbul: Birsen Yayınevi.
- Bektaş, S. (tarih yok). GPS KOORDİNATLARINDAN ÜLKE KOORDİNATLARINA DÖNÜŞÜM. *Üç Boyutlu Jeodezi ve GPS*. (2018). *Büyük Ölçekli Harita ve Harita Bilgileri Üretim Yönetmeliği*. Ankara.
- Data types in Postgres*. (2023, Haziran 15). Promotic Scada Visualization software: <https://www.promotic.eu/en/pmdoc/Subsystems/Db/Postgres/DataTypes.htm> adresinden alındı
- Department of Aerospace Engineering at The University of Bristol. (2023, Şubat 04). *Orbital Frames of Reference*. Youtube: <https://youtu.be/DbYapFLJsPA> adresinden alındı
- Hofmann-Wellenhof, B., Lichtenegger, H., & Collins, J. (1994). *Global Positioning System - Theory and Practice*. Wien: Springer.
- International Association of Oil & Gas Producers. (2024, Ocak 23). *PSG Geodetic Parameter Dataset*. <https://epsg.org/home.html> adresinden alındı
- International Association of Oil & Gas Producers. (2023, Şubat 13). *About the EPSG Dataset*. EPSG Geodetic Parameter Dataset: <https://epsg.org/home.html> adresinden alındı
- Krakiwsky, E. J., & Wells, D. E. (1971, Mayıs). Coordinate Systems in Geodesy. *Ders Notları 16*. Fredericton, Kanada: New Brunswick Üniversitesi, Jeodezi ve Geomatics Mühendisliği.
- Open Geospatial Consortium. (2023, Mart 11). *Geographic information — Well-known text representation of coordinate reference systems*. www.ogc.org: <https://docs.ogc.org/is/18-010r7/18-010r7.html> adresinden alındı
- Özbenli, E. (2001). *Jeodezi-I*. Trabzon: Karadeniz Teknik Üniversitesi Matbaası.
- PostgreSQL. (2024, Ocak 23). *What is PostgreSQL?* [postgresql: https://www.postgresql.org/about/](https://www.postgresql.org/about/) adresinden alındı
- postgresql tutorial. (2023, Temmuz 1). *PL/pgSQL Block Structure*. [postgresql: https://www.postgresqltutorial.com/postgresql-plpgsql/plpgsql-block-structure/](https://www.postgresqltutorial.com/postgresql-plpgsql/plpgsql-block-structure/) adresinden alındı
- PostgreSQLTutorial. (2023, Haziran 30). *PostgreSQL PL/pgSQL*. [postgresql: https://www.postgresqltutorial.com/postgresql-plpgsql/](https://www.postgresqltutorial.com/postgresql-plpgsql/) adresinden alındı
- QGIS. (2024, Şubat 06). *QGIS*. <https://qgis.org/en/site/index.html> adresinden alındı
- Quantum Geographic Information System. (2024, Mayıs 05). *Platformun için QGIS indir*. [Qgis: https://qgis.org/tr/site/forusers/download.html](https://qgis.org/tr/site/forusers/download.html) adresinden alındı

The Open Source Geospatial Foundation. (2023, Mart 23). *St_MakePoint*. www.postgis.net:

https://postgis.net/docs/ST_MakePoint.html adresinden alındı

TORGE, W. (1991). *Geodesy*. Berlin: Walter de Gruyter.

KAMANMYO