



formalistech.com

Hibrit Veri Modeli

A.Erkan ÇELİK
Formalis Bilgi Teknolojileri
CRM Ürün Geliştirme Yöneticisi

Formalis Hakkında



Business Development and Sales

12655 W. Jefferson Blvd.
4th Floor Los Angeles
CA 90066, USA
P. +1 310 666 9425

R&D and Sales

Sahrayıcedid Mah.
Halk Sk. Pakpen Plaza No.40/4
Kadıköy İstanbul, Turkey
T. +90 216 361 5438

Formalis Hakkında

Kuruluş Yılı: **2006**

Aktif Müşteri Sayısı: **200+**

Aktif SaaS Kullanıcı: **40.000+**

Aktif On-Premise Kullanıcı: **30.000+**

Formalis Personel Sayısı: **40**

Deloitte Son 5 Yıllık Büyüme Oranı: **%382**

2018 Bilişim 500: **CRM/BPM Kategorisinde 2. en büyük yazılım üreticisi**

Deloitte'ten 5. ödül

2013, 2015, 2017 yıllarında Türkiye'de **en hızlı büyüyen ilk 50 teknoloji şirketi**.

2015, 2017 yıllarında EMEA bölgesinde (Avrupa, Afrika, Ortadoğu) **en hızlı büyüyen ilk 500 teknoloji şirketi**.

Business Development and Sales

12655 W. Jefferson Blvd.
4th Floor Los Angeles
CA 90066, USA
P. +1 310 666 9425

R&D and Sales

Sahrayıcedid Mah.
Halk Sk. Pakpen Plaza No.40/4
Kadıköy İstanbul, Turkey
T. +90 216 361 5438

50 | Technology **Fast 50**
2013
2015
2017

500 | Technology **Fast 500**
2015
2017

Deloitte.



CSM

Customer Service Management

Mi4biz



sommoni

CRM

Customer Relationship Management

Selphiu

AssistFlow

HelpoAlive

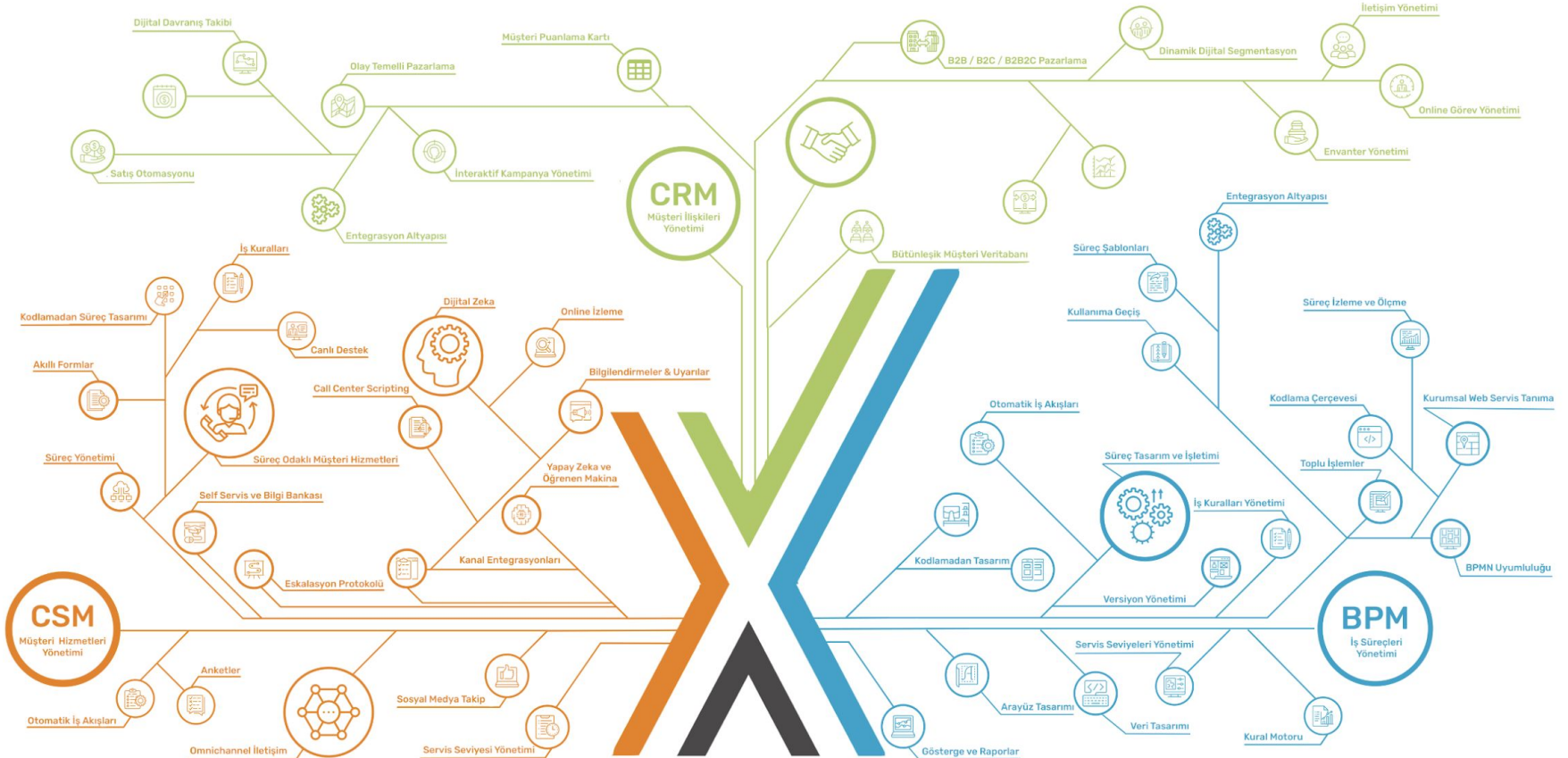
BPM

Business Process Management

opheleia

i-Veriflex

Müşteri Merkezli Dijital Dönüşüm (Next4biz)



Alternatif Bank

ANADOLUBANK

aktif bank

ING BANK

KUVEYT TÜRK

Şekerbank

Türkiye'nin Finans
Türkiye Finans

VAKIF KATILIM

KB
KREDİ KATILIM BÜROSU

Allianz

ANADOLU
SİĞORTA

NN
Hayat ve Emeklilik

sigortam.net

FOREKS

sedaş

sahibinden.com

modanisa.com

BiTaksi

getir

trendyol

evidia.com

oley.com

PASSOLİG

SUPPLEMENTLER
.com

TURKCELL
KUZEY KIBRIS

TURKISH CARGO

Onurair

HOROZ
LOJİSTİK

MUTLU

Mercedes-Benz

TOFAŞ

FAK DOĞU

LANIA

FIAT

YİĞİTAKÜ

BOSCH
Yaşam için teknoloji

HAKMAN

ALARKO
Carrier

DAIKIN

İKLİM SA
Türkiye'nin İklimlendirme Markası

TEKNO SA

UNIFREE
DUTYFREE

bernardo
Sofra Tasarım Uzmanı

Koctaş

Carrefour

[yataş]

MUDO

VAKKO

flormar
Professional Discount

ORIFLAME
SWEDEN

DANONE

MONTERO

EMAAR
SQUARE

42
MALLAK

nef

pekdemir

uyumsoft

egebimtes
1993 ... İlk Teknoloji A.Ş.

TANULU AKDEMİR

ROBERT COLLEGE
- 1863 -

DERİNDERE FİLO KIRALAMA

Hedef Filo

BOYÜKŞEHİR
BELEDİYESİ

KOCAELİ
BOYÜKŞEHİR BELEDİYESİ

BURULAŞ

COĞRAZ

AYRANCI
AYRANCI

Milangaz

HAYAT

STÄUBLI

umur

Gedikyatırım

evony

Papia

Evyap

activex

ARKO

Emotion

PRIVACY

bardata
Technology Advisory & Design

VIP
FIRST CLASS

etb group

TeknoSOR

11820
Teknoloji Destek Hattı

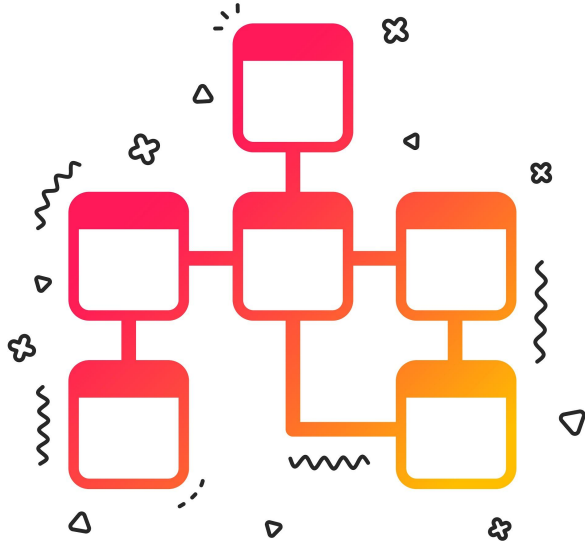
Casper

ZTE

Bingo

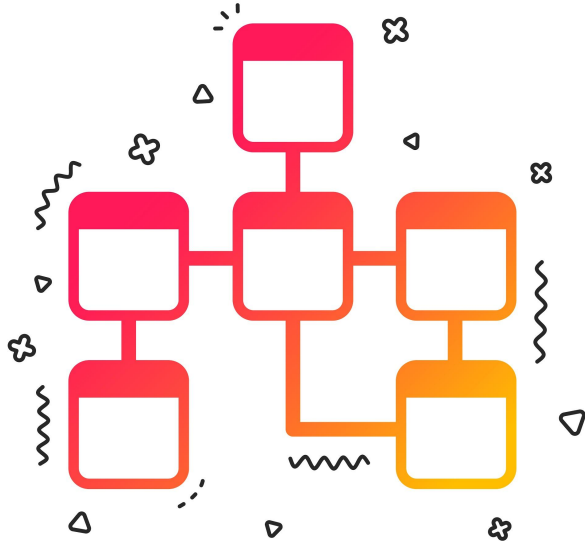
somfy.

İlişkisel Veri Modeli



İlişkisel veri tabanı, birbirinden farklı tablolara yerleştirilmiş olan verilerin birbirleri ile belirli alanlara göre ilişkilendirilerek düzenlenen veri tabanlarıdır. İlişkisel veri tabanları birden fazla tablodan oluşabilir.

İlişkisel Veri Modeli



- Veri tekrarıdan kaçınır
- Veriler yazılırken parçalanır, okunurken birleştirilir
- Daha az fiziksel depolama kullanmayı amaçlar
- Veriyi gerçek zamanlı olarak işlemek ve doğruluğunun kontrolü çok önemlidir.

İlişkisel Veri Modeli

Örnek Problem:

Bir e-ticaret sitesindeki ürünleri, ilişkili markalar ve kategoriler ile birlikte depolayacak bir ilişkisel veritabanı yapısı tasarlayınız.

Her bir kategorinin, alt kategorileri olabilir.

Ürünler en uç kategori ile ilişkili olacaktır.

Ayrıca ürünlerin, özellikleri de veritabanında tutulacaktır.

Ürün özellikleri:

- Metin
- Liste Elemanı
- Çoklu Liste Elemanı
- Sayı

Türlerinde olabilir.

İlişkisel Veri Modeli

Örnek Problem:

Örneğin, ürünün bir “Dizüstü Bilgisayar” olduğunu hayal edelim

Kategori	Elektronik ve Bilgisayar > Bilgisayar > Dizüstü bilgisayar
Marka	Lenovo
Ürün Adı	Ideapad S145-14IWL
Özellikler	Bluetooth : Var (Liste Elemanı) Ekran Boyutu: 14 (Sayı) Ram : 4 (Sayı) Görüntü Çıkışı: HDMI, VGA (Çoklu Liste Elemanı) ...

İlişkisel Veri Modeli

Örnek Problem Çözümü:



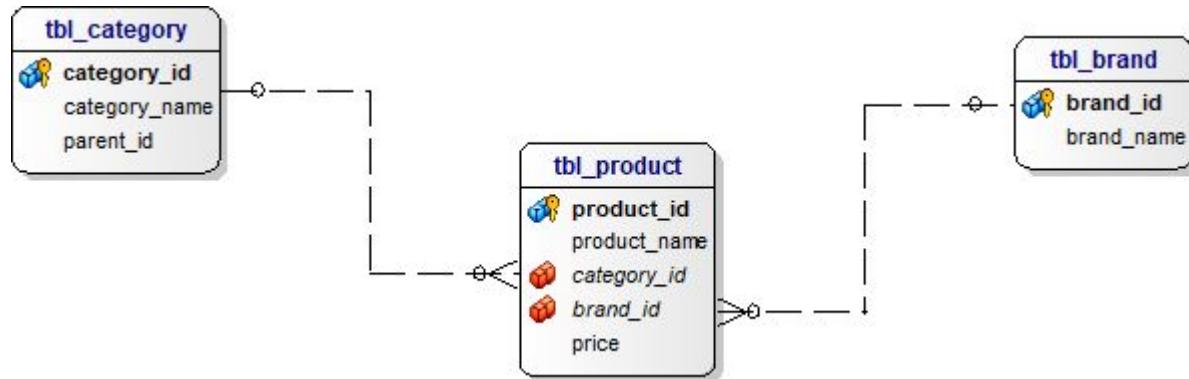
İlişkisel Veri Modeli

Örnek Problem Çözümü:



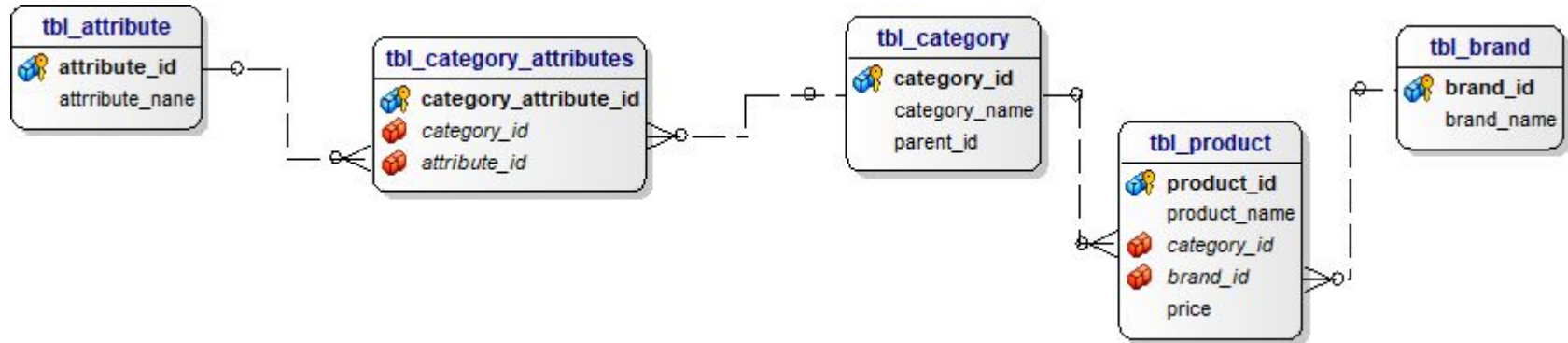
İlişkisel Veri Modeli

Örnek Problem Çözümü:



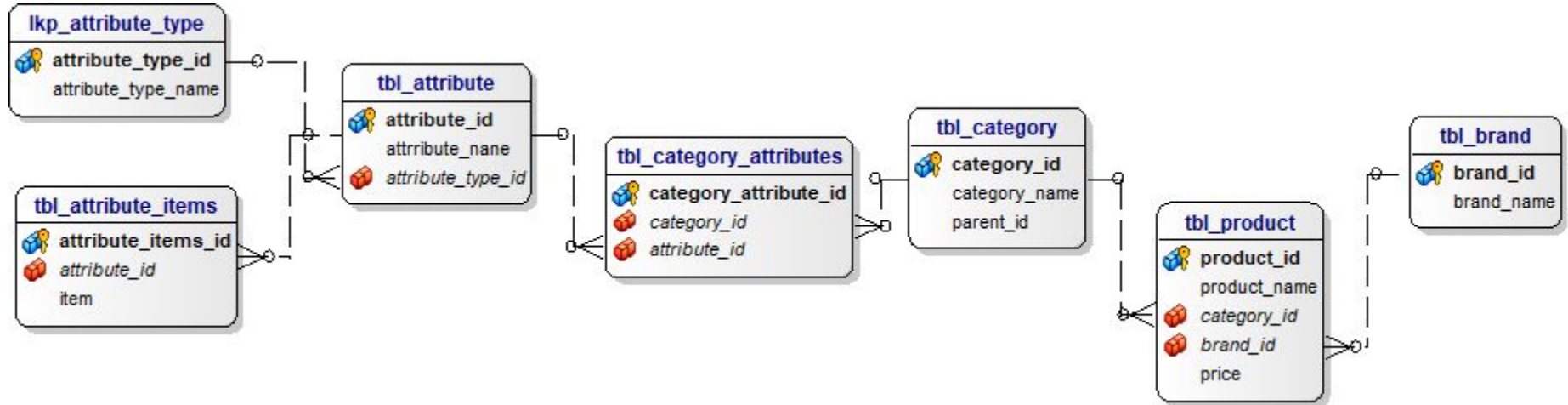
İlişkisel Veri Modeli

Örnek Problem Çözümü:



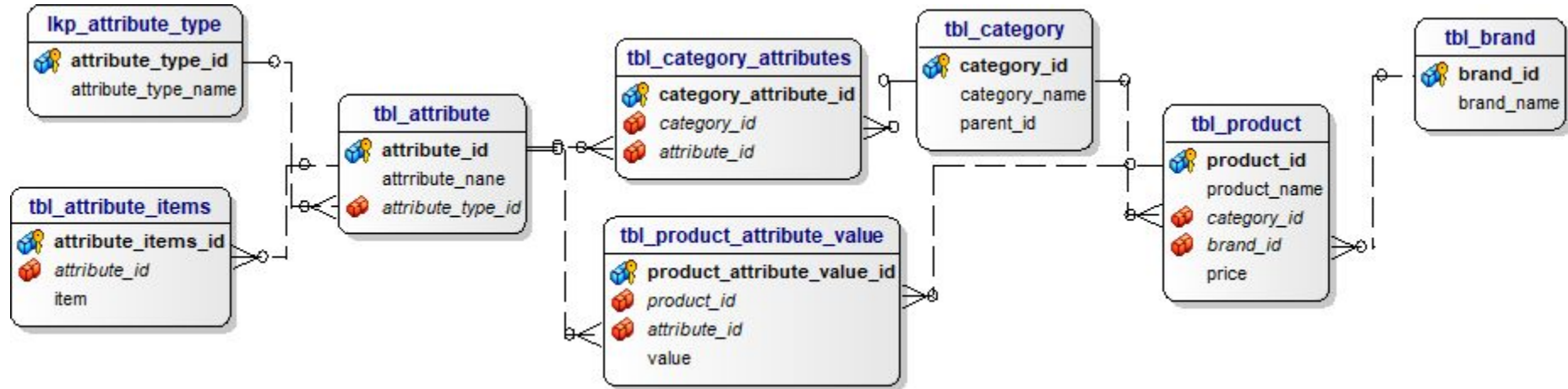
İlişkisel Veri Modeli

Örnek Problem Çözümü:



İlişkisel Veri Modeli

Örnek Problem Çözümü:



İlişkisel Veri Modeli

Örnek Problem:

Dizüstü bilgisayar kategorisinde, ağırlığı 2 kg'nin altında, RAM'i 8 GB ve üzerinde olan Diski 256 Gb ve üzeri olan, disk türü SSD olan, işlemcisi intel i7 olan bir bilgisayarları listeleyelim.

İlişkisel Veri Modeli

Örnek Problem Çözümü:

tbl_category

category_id	category_name
100	Dizüstü bilgisayar

İlişkisel Veri Modeli

Örnek Problem Çözümü:

tbl_attribute

attribute_id	attribute_name
8	ağırlık
15	bellek
18	disk kapasitesi
22	disk türü
25	işlemci türü

İlişkisel Veri Modeli

Örnek Problem Çözümü:

tbl_attribute_items

attribute_item_id	attribute_id	item
3	25	intel i7
2	22	SSD

İlişkisel Veri Modeli

Örnek Problem Çözümü:

```
SELECT
    tbl_brand.brand_name,
    tbl_product.product_name,
    tbl_product.price
FROM
    tbl_product
    INNER JOIN tbl_brand ON (tbl_brand.brand_id = tbl_product.brand_id)
    INNER JOIN tbl_category ON (tbl_category.category_id = tbl_product.category_id)
    INNER JOIN tbl_product_attribute_value AS agirlik ON (agirlik.product_id = tbl_product.product_id)
    INNER JOIN tbl_product_attribute_value AS bellek ON (bellek.product_id = tbl_product.product_id)
    INNER JOIN tbl_product_attribute_value AS disk_kapasitesi ON (disk_kapasitesi.product_id = tbl_product.product_id)
    INNER JOIN tbl_product_attribute_value AS disk_turu ON (disk_turu.product_id = tbl_product.product_id)
    INNER JOIN tbl_product_attribute_value AS islemci_turu ON (islemci_turu.product_id = tbl_product.product_id)
WHERE
    agirlik.value::real <= 2 AND
    bellek.value::integer >= 8 AND
    disk_kapasitesi.value::integer >= 256 AND
    disk_turu.value::integer = 2 AND
    islemci_turu.value::integer = 3
```

İlişkisel Veri Modeli

Bu örnekten yola çıkarak ilişkisel veri modelinde gözlemlediğimiz problemler:

- ürün tablosunda 1 milyon ürün olsa
- bir ürün için ortalama 10 özellik olsa
- ürün özellikleri tablosunda 10 milyon kayıt olmalı
- Sorgulama sırasında 10 kez INNER JOIN yapıldığında 10 milyon kayıt içeren aynı tabloyu 10 kez okumak gerekecektir!

İlişkisel Olmayan Veri Modeli (NoSQL)

NoSQL: Not Only SQL

NoSQL veritabanları, belirli veri modelleri için özel olarak tasarlanmıştır ve modern uygulamalar oluşturmaya yönelik esnek şemalara sahiptir. NoSQL veritabanları uygun ölçekte geliştirme kolaylığı, işlevselliği ve performansıyla geniş çaplı kabul görmüştür. document, graph, key-value, in-memory ve search dahil olmak üzere çeşitli veri modelleri kullanır.

Bu tür veritabanları, özellikle büyük veri hacmi, düşük gecikme süresi ve esnek veri modelleri gerektiren uygulamalar için optimize edilmiştir. Bu gereksinimler, diğer veritabanlarının veri tutarlılığı kısıtlamalarının bir kısmı esnetilerek karşılanır.

(**kaynak:** aws)

İlişkisel Olmayan Veri Modeli (NoSQL)

Aynı problemi NoSQL ile çözmeye çalışırsak Ürün Dökümanı şöyle olur:

```
{
  "id": "5771953cac3790b64d8b4567",
  "product_name": "Huawei Mate 20 Lite 64 GB",
  "brand": {
    "brand_id": "6871151cbd3701b64d8b4510",
    "brand_name": "Huawei"
  },
  "category": {
    "category_id": "6151665abc3801aa4d8b451a",
    "category_name": "Cep Telefonları"
  },
  "attributes": {
    "Ağırlık": "172 g",
    "Dahili Hafıza": "64 GB",
    "RAM Kapasitesi": "4 GB RAM",
    "Ön (Selfie) Kamera": "24 MP + 2 MP",
    "İşlemci Kapasitesi": "2,2 GHz Quad Core + 1,7 GHz Quad Core",
    "Kamera Çözünürlüğü": "20 MP + 2 MP",
  },
  "price": "2169.99"
}
```

İlişkisel Olmayan Veri Modeli (NoSQL)

Benzer şekilde category, brand ve category attribute dökümanları da oluşturulacaktır. Buradaki problem veri bütünlüğü bozulacaktır. Bir ürüne ait marka bilgisi hem product tablosunda hem de brand tablosunda olmalıdır.

```
{  
  "id": "6871151cbd3701b64d8b4510",  
  "brand_name": "Huawei"  
}
```

Bir marka isminde düzenleme yapılacak olursa, bu markaya ait tüm ürünlerde de bu güncelleme yapılması gereklidir. Bu maliyetin büyüklüğü bu tür işlemlerin geri planda yapılmasını zorunlu kılar.

Örneğin bankacılık işlemleri için bu hoş bir durum değildir.

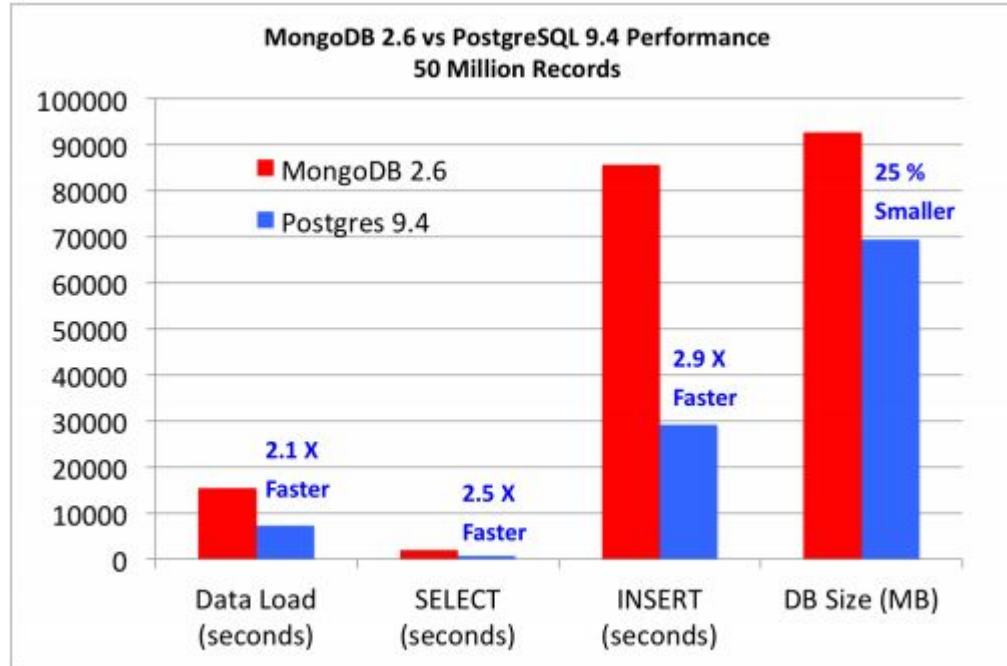
İlişkisel Olmayan Veri Modeli vs İlişkisel Veri Modeli

Table 1: MongoDB and Postgres Comparison (Absolute)

	MongoDB 2.6	PostgreSQL 9.4
Data load (s)	15391	7319
Inserts (s)	85639	29125
Selects (s)	1929	753
DB Size (GB)	92.63	69.36


Figure 10: Relative Performance Comparison of MongoDB 2.6 with PostgreSQL 9.4

İlişkisel Olmayan Veri Modeli vs İlişkisel Veri Modeli




Ne zaman NoSQL

Bir e-ticaret sitesinde Bir milyon ürün olabilir. Bu ürünlerin her birinin detay sayfası vardır. Bu detay sayfalarına ziyaretçiler, tıklar. Her bir tıklama kayıt altına alınmalıdır. Bu tıklamaların bazıları, kimliği belli tıklamalardır. Yani sitenin üyesi olan, oturum açmış müşterilerin tıklamalarıdır.

tbl_product_click	
 product_click_id	BIGSERIAL
product_id	BIGINT
clicked_at	TIMESTAMP
user_id	BIGINT

Ne zaman NoSQL

Bir e-ticaret sitesinde Bir milyon ürün olabilir. Bu ürünlerin her birinin detay sayfası vardır. Bu detay sayfalarına ziyaretçiler, tıklar. Her bir tıklama kayıt altına alınmalıdır. Bu tıklamaların bazıları, kimliği belli tıklamalardır. Yani sitenin üyesi olan, oturum açmış müşterilerin tıklamalarıdır.

tbl_product_click	
 product_click_id	BIGSERIAL
product_id	BIGINT
clicked_at	TIMESTAMP
user_id	BIGINT

Ne zaman NoSQL

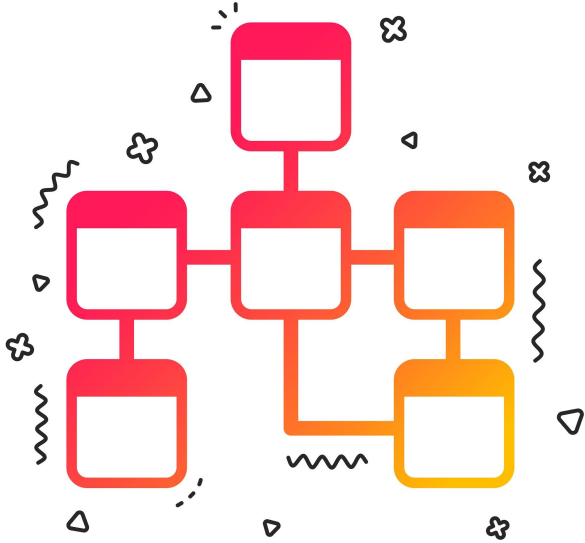
Böyle bir tabloda milyonlarca kayıt bulunur ama hiç birisi tek başına bir anlam ifade etmez. Ancak bir araya geldiklerinde bazı soruların yanıtı olurlar:

- X ürününe kaç kişi bakmış?
- Bir üye X ürününü farklı zamanlarda kaç kez incelemiş?
- En çok hangi ürünler ziyaret edilmiş?
- X ürününü ziyaret edenler başka hangi ürünleri en çok ziyaret etmiş? Buradan ürün önerileri de çıkar. Bu ürünü inceleyenler şu ürüne de baktılar gibi.

Ne zaman NoSQL

Görüldüğü üzere bu tablodaki verilerin kullanılabilmesi için bir işleme tabi tutulmaları gerekir. Devasa büyüklükte olacağı tahmin edilebilen bu tablo ilişkisel yapıda olduğunda veri tabanını da şişirecektir! Bu tablonun ilişkisel bir veri tabanında olmasına gerek yoktur. Ancak işlem sonrasında ortaya çıkan rakamlar ilişkisel veri tabanına yazılabilir.

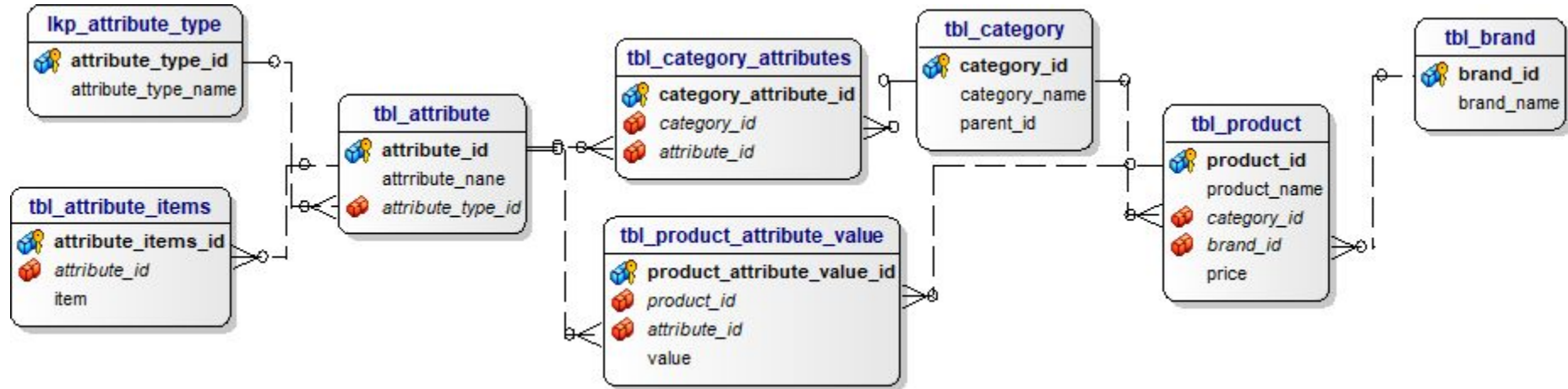
HİBRİT VERİ MODELİ



Hibrit data Model bu gibi durumlar için kullanılabilecek bir çözümdür. İlişkisel veritabanında ilişkisel olmayan verileri tutmaya yarar.

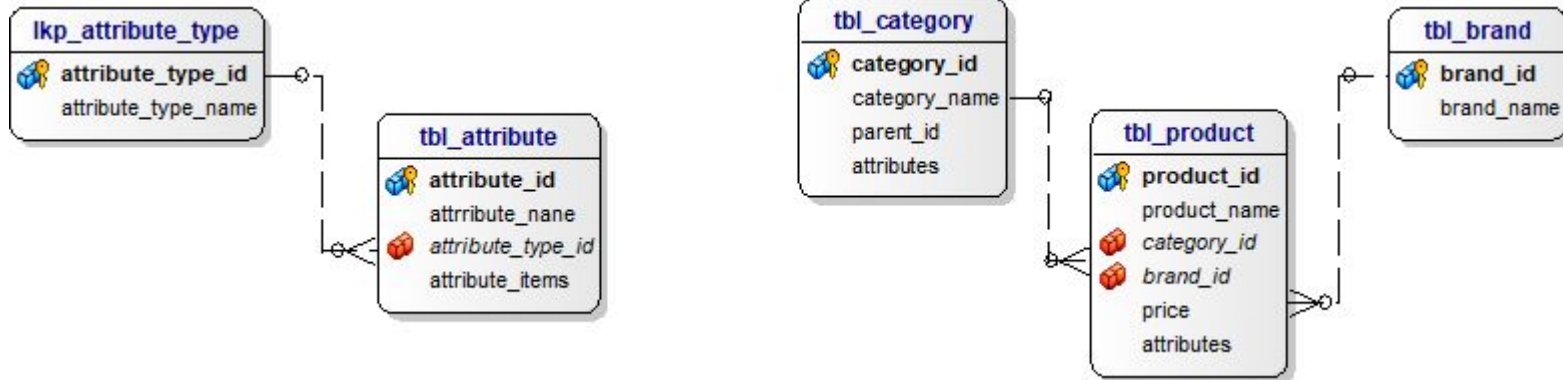
HİBRİT VERİ MODELİ

İlişkisel Modeldeki Şu Çözüm :



HİBRİT VERİ MODELİ

Hibrit Model ile şu şekilde olabilir :



HİBRİT VERİ MODELİ

Avantajları

- İlişkisel veri tabanı olduğu için daha az disk kullanır
- veri bütünlüğü korunur
- okuma yazma hızından taviz verilmez. Hatta jsonb tipindeki bir alanda sorgulama doğal alanlarda sorgulamadan daha hızlıdır.
- SQL kullanarak sorgu yazarsınız

HİBRİT VERİ MODELİ

Dezavantajları

- DBMS'nin yerleşik araçları ile import/export yapmak mümkün olmaz. Tüm DB'yi yada tüm tabloyu import export yapabilirsiniz ancak bir csv dosyasından bir tablo oluşturamazsınız. Bunun için DB'nin programlama dilinde kod yazmanız gerekir.
- JSONB türündeki alanlar GIN index ile indexlenebilir ancak index'in diskteki boyutu oldukça büyüktür.
- tarih gibi tiplerin JSON karşılığı olmadığı için kendi çözümünüzü üretmek zorundasınız.

HİBRİT VERİ MODELİ

PostgreSQL

- Postgresql 9.0 versiyonundan itibaren json veri tiplerini desteklemektedir.
- Oracle, MySQL ve MSSQL de json veri tipini destekliyor ancak Postgresql kadar ileri seviyede veri manevraları henüz ya yapamıyorlar yada uzun fonksiyon isimleri yüzünden karmaşıklştırıyorlar.
- Postgresql GIN index ile bu tipteki alanların sorgulamasını hızlandırmaktadır.



HİBRİT VERİ MODELİ

PostgreSQL json ve jsonb operatörleri

Operator	Açıklama	Örnek	Sonuç
->	Bir json içinden key yada index ile belirtilen değeri okur sonuç yine json yada jsonb tipindedir.	'[{"a":"foo"}, {"b":"bar"}, {"c":"baz"}]::json->2	{"c":"baz"}
->>	Bir json içinden property yada index ile belirtilen değeri okur sonuç text tipindedir.	'[1,2,3]::json->>2	3
#>	Bir json objesinden, belirlenen bir path deki json'ı alır. Sonuç yine json yada jsonb tipindedir.	'{"a": {"b":{"c": "foo"}}}'::json#>'{a,b}'	{"c": "foo"}
#>>	Bir json objesinden, belirlenen bir path deki json'ı alır. Sonuç text tipindedir.	'{"a":[1,2,3], "b":[4,5,6]}'::json#>>'{a,2}'	3

HİBRİT VERİ MODELİ

PostgreSQL json ve jsonb operatörleri

Operator	Açıklama	Örnek
@>	Soldaki json/jsonb içinde sağdaki json/jsonb yada değer var mı? (arama en üst seviyede yapılır)	'{"a":1, "b":2}':jsonb @> '{"b":2}':jsonb
<@	Sağdaki json/jsonb içinde soldaki json/jsonb yada değer var mı? (arama en üst seviyede yapılır)	'{"b":2}':jsonb <@ '{"a":1, "b":2}':jsonb
?	Bu json/jsonb içinde böyle bir key var mı?	'{"a":1, "b":2}':jsonb ? 'b'
?	Bu json/jsonb içinde böyle bir keylerden birisi var mı?	'{"a":1, "b":2, "c":3}':jsonb ? array['b', 'c']

HİBRİT VERİ MODELİ

PostgreSQL json ve jsonb operatörleri

Operator	Açıklama	Örnek
?&	Bu json/jsonb içinde bu keylerin hepsi var mı? (arama en üst seviyede yapılır)	<code>['a', 'b']::jsonb ?& array['a', 'b']</code>
	iki json /jsonb yi birleştirir. birleştirme en üst seviyede yapılır.	<code>['a', 'b']::jsonb ['c', 'd']::jsonb</code>
-	Bir json/jsonb den key-value çifti silmek için kullanılır	<code>['a': 'b']::jsonb - 'a'</code>
#-	Belirlenen bir path 'deki json objexini silmek için kullanılır	<code>['a', {'b': 1}]::jsonb #- '{1,b}'</code>

HİBRİT VERİ MODELİ

PostgreSQL json ve jsonb fonksiyonları

- `jsonb_array_length`
- `to_jsonb`
- `jsonb_each`
- `jsonb_each_text`
- `jsonb_extract_path`
- `jsonb_extract_path_text`
- `jsonb_object_keys`
- `jsonb_populate_record`
- `jsonb_populate_recordset`
- `array_to_json`
- `row_to_json`
- `jsonb_array_elements`
- `jsonb_array_elements_text`
- `jsonb_typeof`
- `jsonb_to_record`
- `jsonb_to_recordset`
- `jsonb_strip_nulls`
- `jsonb_set`
- `jsonb_pretty`
- `jsonb_build_array`
- `jsonb_build_object`

HİBRİT VERİ MODELİ

PostgreSQL'de yeni bir tablo oluşturalım

```
create table public.tbl_product
(
  product_id bigserial not null
    constraint tbl_product_pk
      primary key,
  product_name varchar(512),
  properties jsonb,
  brand_id bigint,
  category_id bigint,
  price real,
  real integer,
  product_slug varchar(255)
);
```


HİBRİT VERİ MODELİ

Bir de index ekleyelim

```
CREATE INDEX idx_tbl_product_properties  
ON tbl_product USING GIN(properties);
```

HİBRİT VERİ MODELİ

Örnek Problem:

Dizüstü bilgisayar kategorisinde, ağırlığı 2 kg'nin altında, RAM'i 8 GB ve üzerinde olan Diski 256 Gb ve üzeri olan, disk türü SSD olan, işlemcisi intel i7 olan bir bilgisayarları listeleyelim.

İlişkisel Veri Modeli

Örnek Problem Çözümü:

```
SELECT
    tbl_brand.brand_name,
    tbl_product.product_name,
    tbl_product.price
FROM
    tbl_product
    INNER JOIN tbl_brand ON (tbl_brand.brand_id = tbl_product.brand_id)
    INNER JOIN tbl_category ON (tbl_category.category_id = tbl_product.category_id)
    INNER JOIN tbl_product_attribute_value AS agirlik ON (agirlik.product_id = tbl_product.product_id)
    INNER JOIN tbl_product_attribute_value AS bellek ON (bellek.product_id = tbl_product.product_id)
    INNER JOIN tbl_product_attribute_value AS disk_kapasitesi ON (disk_kapasitesi.product_id = tbl_product.product_id)
    INNER JOIN tbl_product_attribute_value AS disk_turu ON (disk_turu.product_id = tbl_product.product_id)
    INNER JOIN tbl_product_attribute_value AS islemci_turu ON (islemci_turu.product_id = tbl_product.product_id)
WHERE
    agirlik.value::real <= 2 AND
    bellek.value::integer >= 8 AND
    disk_kapasitesi.value::integer >= 256 AND
    disk_turu.value::integer = 2 AND
    islemci_turu.value::integer = 3
```

HİBRİT VERİ MODELİ

Örnek Problem Çözümü:

```
SELECT
tbl_brand.brand_name,
tbl_product.product_name,
tbl_product.price
FROM
tbl_product
INNER JOIN tbl_brand ON (tbl_brand.brand_id = tbl_product.brand_id)
INNER JOIN tbl_category ON (tbl_category.category_id = tbl_product.category_id)
WHERE
(tbl_product.properties->>'agirlik')::real <= 2 AND
(tbl_product.properties->>'bellek')::integer >= 8 AND
(tbl_product.properties->>'disk_kapasitesi')::integer >= 256 AND
tbl_product.properties->>'disk_turu' = 'ssd' AND
tbl_product.properties->>'islemci_turu' = 'intel i7'
```

HİBRİT VERİ MODELİ

Güncelleme Sorgusu:

```
UPDATE tbl_product
  SET properties = properties || '{"kameralar":["ön","arka"]}'
WHERE product_id=1;
```

```
UPDATE tbl_product
  SET properties = properties || jsonb_build_object('kameralar',ARRAY['ön','arka'])
WHERE product_id=1;
```

HİBRİT VERİ MODELİ

Sanal Tablo Problemi:

- Bir bulut (SAAS) CRM uygulamasında farklı sektörler farklı veri kümelerine ihtiyaç duyar
 - Sigorta sektörü Poliçe satar
 - Havayolu firması bilet satar
 - Mobilya firması ürün satar
- Tüm bu ihtiyaçları bulut uygulamada yerleşik olarak oluşturmanın maliyeti (zaman, para, iş-gücü) çok fazladır.
- Bu sebeplerle hibrit veri modeli kullanarak, müşterilerimizin sanal tablolar oluşturabilmesine olanak sağladık.

HİBRİT VERİ MODELİ

Sanal Tablo Problemi:

tbl_vtable

vtable_id	name	title	channel_id
100	urun	Ürün	89

HİBRİT VERİ MODELİ

Sanal Tablo Problemi:

tbl_vtable_row

vtable_row_id	vtable_id	related_id	custom_data	channel_id	relation_kind_id
1	100	NULL	{"urun_adi": "Performance 22VH3021 22",}	89	3

HİBRİT VERİ MODELİ

Sanal Tablo Problemi:

tbl_vtable

vtable_id	name	title	channel_id
100	urun	Ürün	89
101	siparis	Sipariş	89
102	siparis_urunleri	Sipariş Ürünleri	89

HİBRİT VERİ MODELİ

Sanal Tablo Problemi:

tbl_vtable_row

vtable_row_id	vtable_id	related_id	custom_data	channel_id	relation_kind_id
3	101	427406	{"siparis_no": "55422",}	89	1
2	102	3	{"miktar": 2,"urun_id":3,}	89	3
1	100	NULL	{"urun_adi": "Performance 22VH3021 22",}	89	3

HİBRİT VERİ MODELİ

Sanal Tablo Problemi:

```
SELECT
siparis.custom_data->>'siparis_no',
siparis_kalemi.custom_data->>'adet',
urun.custom_data->>'urun'
FROM
tbl_vtable_row as siparis
INNER JOIN tbl_vtable_row as siparis_kalemi ON
    ( siparis_kalemi.related_id = siparis.vtable_row_id ) AND siparis_kalemi.vtable_id = 102
INNER JOIN tbl_vtable_row as urun ON
    ( siparis_kalemi.custom_data->>'urun_id' = urun.vtable_row_id ) AND urun.vtable_id = 100
WHERE
siparis.vtable_id = 101 AND related_id = 427406;
```

ATÖLYE ÇALIŞMASI

Gerekenler:

- Postgresql 11
- DB yönetim aracı (DataGrip)
- Örnek DB
<https://github.com/aerkanc/hybrid-db>

ATÖLYE ÇALIŞMASI

Cep telefonu kategorisi altında hangi ürün özellikleri var?

```
SELECT
  item ->> 0 as attribute
FROM (
  SELECT
    jsonb_array_elements(attributes) as item
  FROM
    tbl_category
  WHERE
    category_id = 2
) as attribute_items;
```

ATÖLYE ÇALIŞMASI

Bir Cep Telefonu Kategorisinde Pil Gücü özelliğinin seçenekleri neler?

```
SELECT
  DISTINCT properties->>'Pil Gücü'
FROM
  tbl_product
WHERE
  category_id = 2;
```

ATÖLYE ÇALIŞMASI

2000 mAh dan büyük pil gücüne sahip telefonlar neler?

```
SELECT
    brand_name,
    product_name,
    price
FROM
    tbl_product
    INNER JOIN tbl_brand ON (tbl_brand.brand_id = tbl_product.brand_id)
WHERE
    category_id = 2 AND
    REPLACE(properties -> 'Pil Gücü', ' mAh', '')::INTEGER > 2000
```

ATÖLYE ÇALIŞMASI

Wifi'si 802.11 ac yi destekleyen telefonlar neler?

```
SELECT
    brand_name,
    product_name,
    tbl_product.properties->>'Wi-Fi' as "Wi-Fi",
    price
FROM
    tbl_product
    INNER JOIN tbl_brand ON (tbl_brand.brand_id = tbl_product.brand_id)
WHERE
    category_id = 2 AND
    tbl_product.properties->>'Wi-Fi' ~ '^802\11.*ac.*$'
```


Teşekkürler