ANALYSIS DOCUMENTATION-TABLE CREATION

Proje: DRIVERA Araç Kiralama Uygulaması

Ürün: DRIVERA

Sorumlu: Backend Developer

Bu doküman, DRIVERA Araç Kiralama Uygulaması için hazırladığımız SIGN UP sürecine ait veritabanı analiz dokümanıdır. Kullanıcı kayıt işlemleri için veri tabanında tutulması gereken veri başlıklarını, veri tiplerini, foreign key'leri ve tabloların yapısını detaylı bir şekilde açıklamaktadır.

Doküman İçeriği:

1. **Giriş (Introduction):** Projenin tanımı ve amacı.
2. **Gereksinimler (Requirements):** Kullanıcı bilgileri ve veri doğrulama süreçleri.
3. **Veritabanı Tasarımı (Database Design):**
    - **Tablolar ve Alanlar**
    - **Veri Tipleri**
    - **Tablolar Arasındaki İlişkiler**
4. **Tablo Değişikliklerinin Yönetimi (Managing Table Changes):** Tablolarda yapılacak değişiklikler için alternatif çözüm yolları (Update, Flag, Status Label kullanımı).

Dokümanda ayrıca, her tablo ve alan için uygun veri tipleri, ilişkiler ve constraint'ler detaylandırılmıştır. Tablolarda yapılacak değişikliklerin yönetimi konusunda da alternatif çözüm yollarını karşılaştırarak sunduk. Bu bölümde, her yöntemin avantajları ve dezavantajları belirtilmiştir.

1. Giriş (Introduction)

- **Proje Tanımı:** Driveera araç kiralama uygulaması için SIGN UP sürecine yönelik veritabanı tasarımı.
- **Amaç:** Kullanıcı kayıt işlemleri için veri tabanında tutulması gereken veri başlıklarını, veri tiplerini, foreign key'leri ve bu tablolarda yapılacak değişikliklerin yönetimini açıklamak.

2. Gereksinimler (Requirements)

- **Kullanıcı Bilgileri:** Kullanıcıların kaydolurken vermesi gereken bilgiler.
- **Veri Doğrulama:** Giriş ve doğrulama için gerekli olan süreçler.

3. Veritabanı Tasarımı (Database Design)

- **Tablolar**:
  - **users:** Kullanıcıların temel bilgilerini saklar.
  - **signup_method:** Kullanıcının kayıt türü veya yöntemi hakkında bilgi saklar. Bu tablo, kullanıcıların hangi yöntemle (e-posta, sosyal medya, telefon vb.) kayıt olduğunu belirtir.
  - **login_history:** Kullanıcıların giriş ve çıkış tarihlerini kaydeder.
  - **social_logins:** Sosyal medya hesaplarıyla kayıt olan kullanıcıların sosyal medya bilgilerini saklar. Bu tablo, sosyal medya girişlerini ve entegrasyonları yönetir.
  - **otp_verifications:** Telefon numarası ile girişlerde OTP doğrulama bilgilerini yönetir.
  - **permissions:** Kullanıcı izinlerini ve rolleri saklar.

1.USERS

Column Name	Data Type	Constraint	Default Value	Description
user_id	İnteger	PRIMARY KEY	nextval('users_user_id_seq')	Kullanıcının benzersiz kimliği.
username	character varying(50)	NOT NULL, UNIQUE		Kullanıcı adı.
email	character varying(100)	NOT NULL, UNIQUE		Kullanıcının e-posta adresi.
password_hash	character varying(255)	NOT NULL		Kullanıcının şifresi (hashli).
phone_number	character varying(25)	UNIQUE (if not null)		Kullanıcının telefon numarası.
is_2fa_enabled	Boolean	NOT NULL	False	İki faktörlü kimlik doğrulama etkin mi?
method_2fa	character varying(20)			İki faktörlü kimlik doğrulama yöntemi.
first_name	character varying(50)	NOT NULL		Kullanıcının adı.
last_name	character varying(50)	NOT NULL		Kullanıcının soyadı.
birth_date	Date	NOT NULL		Kullanıcının doğum tarihi.
birth_place	character varying(100)	NOT NULL		Kullanıcının doğum yeri.
current_location	character varying(100)	NOT NULL		Kullanıcının mevcut konumu.
citizenship	character varying(2)	NOT NULL		Kullanıcının vatandaşlık bilgisi (ISO kodu).
consent_date	timestamp with time zone	NOT NULL		Kullanıcının kayıt tarihini belirtir.

consent_type	character varying(10)	NOT NULL		Kullanıcının rıza türünü belirtir.
check_phone_number	Boolean	CHECK (is_2fa_enabled = false OR phone_number IS NOT NULL)		İki faktörlü kimlik doğrulama etkinse telefon numarası olmalı.
profile_picture_url	character varying(255)			Kullanıcının profil resmi URL'si.
signup_method	character varying(20)	NOT NULL	'manual_signup'::character varying	Kullanıcının kayıt yöntemi (manual_signup, social_media).

Kısıtlar (Constraint) Bölümüne İlişkin Bazı Önemli Notlar:

- Telefon No: Kullanıcı, iki faktörlü doğrulama (2FA) tercih ederse, bu doğrulama işlemi kullanıcı tarafından sağlanan telefon numarası üzerinden gerçekleştirilecektir. Bu alanın doğrulaması, veritabanında oluşturulan check_phone_number() fonksiyonu ile yapılacaktır. **check_phone_number() Fonksiyonu:** Fonksiyon, is_2fa_enabled alanının değeri TRUE olduğunda phone_number alanının NULL olmasını engellemek için kullanılır.
    - Eğer is_2fa_enabled TRUE ise, phone_number alanı NOT NULL olmalıdır.
    - Eğer is_2fa_enabled FALSE ise, phone_number alanı NULL olabilir.
- Vatandaşlık Bilgisi (citizenship): Kullanıcıya ait veri gizliliği politikaları vatandaşlık bilgisine göre sunulacaktır.
    - TC Vatandaşları: KVKK (Kişisel Verilerin Korunması Kanunu) gereksinimlerine uygun olarak veri gizliliği politikası uygulanacaktır.
    - Avrupa Vatandaşları: GDPR (General Data Protection Regulation) gereksinimlerine uygun olarak veri gizliliği politikası uygulanacaktır.
    - Diğer: Drivera tarafından oluşturulmuş kullanıcı veri transferi, veri saklama ve veri gizliliği politikası uygulanacaktır.
- Signup Metodu(signup_method): Kullanıcı kayıt ekranından kayıt işlemlerini tamamlar ve “Register” butonuna basarsa default olarak veritabanına “manuel_signup” olarak kayıt oluşturulacaktır. Sosyal medya hesaplarıyla kayıt oluşturulması durumunda ise “social_media” olarak değer atanacaktır.

Veritabanı Tasarımı (Database Design) Bölümüne İlişkin Bazı Önemli Notlar:

- Şifre Gizliliği: Kullanıcı şifreleri (password_hash) veritabanına hashlenerek eklenmiştir. Bu, kullanıcı şifrelerinin güvenliğini sağlar.
    - Şifre Hashleme: Kullanıcı şifreleri güvenlik amacıyla Scrypt, Bcrypt ve ARGON2 gibi güçlü hashleme algoritmaları kullanılarak saklanmalıdır.
- Zaman Damgaları: TIMESTAMPTZ veri tipi kullanılarak zaman damgaları UTC olarak saklanır. Bu, geniş bir kullanıcı kitlesi olan uygulamalarda zaman dilimi problemlerini önler ve gerektiğinde zaman dilimi bilgisi ile görüntülenebilir.

Indeks Yönetimi:

Tablo üzerinde veri aramayı hızlandıran bir veri yapısıdır. Genellikle bir veya daha fazla sütunu hedef alarak, bu sütunlardaki değerler için hızlı erişim sağlar. Arama performansını artırır ve sıralama, gruplama işlemlerini hızlandırır.

- User_id, primary key olması dolayısıyla otomatik olarak indekslenmiştir.
- "username, email, phone_number" sütunlarının kullanıcı kayıt işlemlerinden sonra login türüne bağlı olarak arama ve sıralama işlemlerinde sıkça kullanılacak sütunlar olması beklenmektedir. Bu alanlardan biri ya da birkaçı (kompozit olarak) indekslenebilir.

2. SIGNUP METHOD TABLE

Column Name	Data Type	Constraint	Default Value	Description
signup_method_id	integer	PRIMARY KEY	nextval('signup_methods_signup_method_id_seq')	Kayıt yöntemi kaydının benzersiz kimliği. Otomatik olarak artan bir sayı.
user_id	integer	FOREIGN KEY (user_id) REFERENCES public.users(user_id) ON DELETE CASCADE		Kullanıcının kimliği. users tablosuna referans. Kullanıcı silindiğinde ilgili kayıt da silinir.
method	character varying(20)	NOT NULL		Kayıt yöntemi. Örneğin, 'email', 'google', 'facebook', 'instagram' gibi değerler alabilir.

Kısıtlar (Constraint) Bölümüne İlişkin Bazı Önemli Notlar:

- **user_id:** Bu sütun, users tablosuna referans verir ve bir kullanıcının kimliğini belirtir. FOREIGN KEY olarak tanımlandığından, users tablosundaki bir kullanıcı silindiğinde, bu izinlerle ilgili kayıtların nasıl işleneceği (örneğin, ON DELETE CASCADE kuralı ile silinmesi) belirlenmelidir. Bu kural, kullanıcıyla ilişkili kayıt yöntemlerinin tutarlılığını sağlar.
- **method:** Kayıt yöntemini belirtir ve NOT NULL olarak tanımlanmıştır. Bu sütun, kayıt yöntemlerinin türlerini (örneğin, email, google, facebook, instagram) açıkça belirtir ve bu türlerin geçerli ve anlamlı olduğundan emin olur.

Veritabanı Tasarımı (Database Design) Bölümüne İlişkin Bazı Önemli Notlar:

- **Kullanıcı Kayıt Yöntemi:** Bu tablo, kullanıcıların hangi yöntemle kayıt olduklarını izlemek için kullanılır. method sütunu, kullanıcıların hangi platformdan veya yöntemle kayıt olduklarını belirler ve bu bilgi, kullanıcıların daha iyi yönetilmesi ve analiz edilmesi için kullanılır.
- **Otomatik Artan ID:** signup_method_id sütunu, her kayıt için benzersiz bir kimlik sağlar ve nextval('signup_methods_signup_method_id_seq') ile otomatik olarak artan bir değer alır. Bu, kayıtların benzersiz ve izlenebilir olmasını sağlar.

İndeks Yönetimi:

- **user_id:** FOREIGN KEY olarak tanımlandığından, bu sütun üzerinde bir indeks oluşturulması, belirli bir kullanıcının kayıt yöntemlerinin hızlı bir şekilde sorgulanmasını sağlar. Örnek bir indeks ekleme kodu:

3. LOGIN HISTORY TABLE

Column Name	Data Type	Constraint	Default Value	Description
login_id	İnteger	PRIMARY KEY	nextval('login_history_login_id_seq')	Giriş geçmişinin benzersiz kimliği.
user_id	İnteger	FOREIGN KEY (user_id)		Kullanıcının kimliği.
login_time	timestamp with time zone	NOT NULL		Giriş zamanı.
logout_time	timestamp with time zone			Çıkış zamanı.
ip_address	character varying(45)			Giriş yapılan IP adresi.
device_info	character varying(255)			Giriş yapılan cihaz bilgisi.

Sütun Açıklamaları (Description) Bölümüne İlişkin Bazı Önemli Notlar:

- **user_id**: Kullanıcının kimliği. Bu sütun, girişin hangi kullanıcıya ait olduğunu belirler ve users tablosuna dış anahtar (foreign key) ile bağlanır.
- **logout_time**: Çıkış zamanı. Kullanıcının sistemden çıktığı zamanı belirtir. Bu alan boş olabilir (NULL), çünkü kullanıcı çıkış yapmadan oturumunu kapatabilir.
- **ip_address**: Giriş yapılan IP adresi. Kullanıcının giriş yaptığı cihazın IP adresini belirtir. character varying(45) veri türü, hem IPv4 hem de IPv6 adreslerini destekler.
- **device_info**: Giriş yapılan cihaz bilgisi. Kullanıcının giriş yaptığı cihaz hakkında bilgi sağlar (örneğin, tarayıcı ve işletim sistemi bilgileri).

Tabloya İlişkin Ek Bazı Notlar:

- **Veri Güvenliği ve Gizliliği**: IP adresleri ve cihaz bilgileri gibi hassas verilerin toplanması ve saklanması sırasında veri güvenliği ve gizliliği politikalarına uyulmalıdır. Kullanıcıların gizliliğini korumak için uygun önlemler alınmalıdır. Bu politikalarla ilgili bilgilendirmeniz beklenmektedir.
- **Giriş ve Çıkış Zamanlarının Kaydı**: login_time ve logout_time sütunları, kullanıcıların aktifliklerini izlemek ve güvenlik analizleri yapmak için kullanılır. logout_time boş olabilir, bu yüzden bu sütunun NULL değeri alabilmesi uygun bir tasarımdır.
- **Performans İyileştirme**: Login kayıtlarının sıkça sorgulanacağı durumlarda, user_id, login_time, ve ip_address gibi sütunlarda indeks oluşturulması, sorgu performansını artırabilir.
- **Veri Saklama Süresi**: Login geçmişi verilerinin ne kadar süreyle saklanacağı ve eski verilerin nasıl arşivleneceği hakkında bir politika belirlenmelidir.

4. SOCIAL LOGIN TABLE

Column Name	Data Type	Constraint	Default Value	Description
social_login_id	İnteger	PRIMARY KEY	nextval('social_logins_social_login_id_seq')	Sosyal giriş kaydının benzersiz kimliği.
user_id	İnteger	FOREIGN KEY (user_id)		Kullanıcının kimliği.
provider	character varying(50)	NOT NULL		Sağlayıcı adı (ör. Google, Facebook).
provider_user_id	character varying(255)	NOT NULL		Sağlayıcı kullanıcı kimliği.
access_token	character varying(255)			Erişim belirteci.
refresh_token	character varying(255)			Yenileme belirteci.

Tabloya İlişkin Bazı Önemli Notlar ve Kısıtlar:

- Sosyal medya ile kayıt sırasında, sosyal medya sağlayıcısının kimlik doğrulaması kullanılarak mevcut kullanıcı olup olmadığı kontrol edilir; eğer kullanıcı mevcut değilse, sosyal medya bilgileri ile yeni bir hesap oluşturularak social_logins tablosuna ilgili veriler eklenir.
- Sosyal Medya ile Kayıt Süreci
    - **Sosyal Medya Sağlayıcısının Seçilmesi:** Kullanıcı, Google, Facebook veya Twitter gibi bir sosyal medya sağlayıcısını seçer ve genellikle bir butona tıklayarak işlemi başlatır.
    - **OAUTH 2.0 ile Kimlik Doğrulama:** Kullanıcı, seçilen sosyal medya sağlayıcısına yönlendirilir ve burada kimlik doğrulama işlemi gerçekleştirilir. Kullanıcı, sosyal medya hesabıyla giriş yapar ve gerekli izinleri verir.
    - **Kullanıcı Bilgilerinin Alınması:** Sosyal medya sağlayıcısı, başarılı kimlik doğrulama sonrasında kullanıcı bilgilerini (ad, soyad, e-posta vb.) geri gönderir.
    - **Veritabanına Kayıt:** Alınan bilgiler kullanılarak sistemde yeni bir kullanıcı hesabı oluşturulur. Bu aşamada bazı alanlar otomatik olarak doldurulur; geri kalan bilgileri ise kullanıcı manuel olarak sağlar.
- **Foreign Key Kısıtlaması**: social_logins tablosundaki user_id sütunu için FOREIGN KEY kısıtlaması oluşturulurken, ON DELETE CASCADE seçeneğinin eklenmesi önemlidir. Bu, sosyal medya ile kaydolan bir kullanıcının users tablosundan silinmesi durumunda, bu kullanıcıya ait tüm sosyal giriş kayıtlarının da otomatik olarak silinmesini sağlar. Bu, veri tutarlılığını korur ve gereksiz veri kalıntılarını önler.
- **Veri Güvenliği ve Gizliliği**: access_token ve refresh_token gibi hassas bilgilerin saklanması ve yönetilmesi sırasında veri güvenliği önlemlerine uyulmalıdır. Token'lar, gizli ve güvenli bir şekilde saklanmalı ve gerektiğinde yenilenmelidir.
- **Token Yönetimi**: access_token ve refresh_token sütunları, sosyal girişlerin geçerliliği ve sürekliliği için önemlidir. Token'lar yönetilirken uygun güvenlik önlemleri alınmalıdır.
- **İndeksleme**: Sosyal girişlerin hızlıca erişilmesi ve sorgulanması için user_id ve provider_user_id gibi sütunlarda indeksleme yapılabilir.

5. OTP VERIFICATION TABLE

Column Name	Data Type	Constraint	Default Value	Description
otp_id	İnteger	PRIMARY KEY	nextval('otp_verifications_otp_id_seq')	OTP doğrulama kaydının benzersiz kimliği.
user_id	İnteger	FOREIGN KEY (user_id)		Kullanıcının kimliği.
phone_number	character varying(20)	NOT NULL		Kullanıcının telefon numarası.
otp_code	character varying(10)	NOT NULL		OTP kodu.
is_verified	Boolean		false	OTP doğrulandı mı?
expiry_date	timestamp with time zone	NOT NULL		OTP kodunun son kullanma tarihi.
created_at	timestamp with time zone	DEFAULT CURRENT_TIMESTAMP		Kaydın oluşturulma tarihi.

Kısıtlar (Constraint) Bölümüne İlişkin Bazı Önemli Notlar:

- **user_id:** Bu sütun, users tablosuna referans verir ve bir kullanıcının kimliğini belirtir. ON DELETE CASCADE kuralı ile kullanıcı silindiğinde ilgili OTP kayıtları da otomatik olarak silinir. Bu, veri tutarlılığını sağlar ve kullanılmayan kayıtların temizlenmesine yardımcı olur.
- **phone_number:** Telefon numarası, iki faktörlü doğrulama (2FA) için kullanılır. Eğer 2FA etkinse, telefon numarası NOT NULL olmalıdır. Kullanıcı, 2FA'yı etkinleştirdiğinde, telefon numarasının

sağlanması zorunludur. Eğer 2FA etkin değilse, telefon numarası NULL olabilir. Bu durum, is_2fa_enabled alanının değeriyle ilişkilidir ve telefon numarasının geçerli olup olmadığını doğrulayan bir check_phone_number() fonksiyonu tarafından kontrol edilir.

- **otp_code:** OTP kodu, geçici doğrulama kodunu içerir ve kullanıcının kimliğini doğrulamak için kullanılır. Bu kodun NOT NULL olarak tanımlanması, her OTP kaydında bir kod bulunmasını garanti eder.
- **is_verified:** Bu sütun, OTP kodunun doğrulanıp doğrulanmadığını belirler. Varsayılan olarak false olarak ayarlanır ve doğrulama işlemi tamamlandığında true olarak güncellenir.
- **expiry_date:** OTP kodunun geçerlilik süresini belirtir. Bu tarih geçtikten sonra kod geçersiz hale gelir. Bu sütun NOT NULL olarak tanımlanmıştır, böylece her OTP kaydında bir son kullanma tarihi bulunur.
- **created_at:** Kayıdın oluşturulma tarihini ve saatini gösterir. Varsayılan olarak CURRENT_TIMESTAMP kullanılır ve bu, kayıt oluşturulduğu anı doğru bir şekilde yansıtır.

Veritabanı Tasarımı (Database Design) Bölümüne İlişkin Bazı Önemli Notlar:

- **Şifre Gizliliği:** OTP kodları, güvenlik amacıyla hashlenmiş veya şifrelenmiş olarak saklanmaz; ancak bu kodların geçerliliği ve doğrulama süreci dikkatlice yönetilmelidir.
- **Zaman Damgaları:** TIMESTAMPTZ veri tipi kullanılarak zaman damgaları UTC olarak saklanır. Bu, zaman dilimi farklılıklarından kaynaklanan sorunları önler ve verilerin her zaman doğru bir zaman diliminde saklanmasını sağlar.

İndeks Yönetimi:

- **user_id:** PRIMARY KEY olarak tanımlandığından otomatik olarak indekslenmiştir. Bu, kullanıcıya ait OTP kayıtlarına hızlı erişim sağlar.
- **phone_number:** OTP doğrulama işlemlerinde telefon numarasına göre arama ve sıralama yapılabilir. Bu nedenle, performansı artırmak için phone_number sütununa indeks eklemek yararlı olabilir.

  Örnek Indeksleme Kodu:

  “**CREATE INDEX** idx_phone_number **ON** public.otp_verifications(phone_number);”

6. PERMISSION TABLE

Column Name	Data Type	Constraint	Default Value	Description
permission_id	İnteger	PRIMARY KEY	nextval('permissions_permission_id_seq')	İzin kaydının benzersiz kimliği.
user_id	İnteger	FOREIGN KEY (user_id)		Kullanıcının kimliği.
permission_type	character varying(50)	NOT NULL		İzin türü (ör. read, write).
granted_at	timestamp with time zone	DEFAULT CURRENT_TIMESTAMP		İznin verildiği zaman.
expires_at	timestamp with time zone			İznin sona erme tarihi.
status	character varying(20)	DEFAULT 'active'		İznin durumu (ör. aktif, pasif).
description	Text			İzinle ilgili açıklama.

Kısıtlar (Constraint) Bölümüne İlişkin Bazı Önemli Notlar:

- **user_id:** Bu sütun, users tablosuna referans verir ve bir kullanıcının kimliğini belirtir. FOREIGN KEY olarak tanımlanmıştır ve kullanıcının silinmesi durumunda (ON DELETE CASCADE veya benzeri bir kural) ilgili izin kayıtları da otomatik olarak silinmelidir. Bu, veri bütünlüğünü ve tutarlılığını sağlar.
- **permission_type:** Bu sütun, izin türünü tanımlar (örneğin, read, write, admin). NOT NULL olarak tanımlanmıştır ve her izin kaydında bir izin türü belirtildiğinden emin olunmalıdır.
- **granted_at:** İznin verildiği zamanı gösterir ve varsayılan olarak CURRENT_TIMESTAMP kullanılarak otomatik olarak ayarlanır. Bu, izinlerin ne zaman verildiğini takip etmek için önemlidir.
- **expires_at:** İznin sona erme tarihini belirtir. Bu sütun NULL olabilir, yani bazı izinler süresiz olarak geçerli olabilir. İzinlerin geçerlilik süresinin son bulmasını yönetmek için kullanılır.
- **status:** İznin geçerli olup olmadığını belirten bir durum sütunudur. Varsayılan olarak 'active' olarak ayarlanmıştır. İzinler, 'active', 'inactive' gibi durumlar arasında değiştirilebilir. İzinlerin geçerliliğini yönetmek için önemlidir.
- **description:** İzinle ilgili açıklamaları içerir. Bu alan, izinlerin amacı ve kapsamı hakkında bilgi vermek için kullanılır. Text veri tipi kullanılarak, uzun açıklamaların da saklanabilmesi sağlanır.

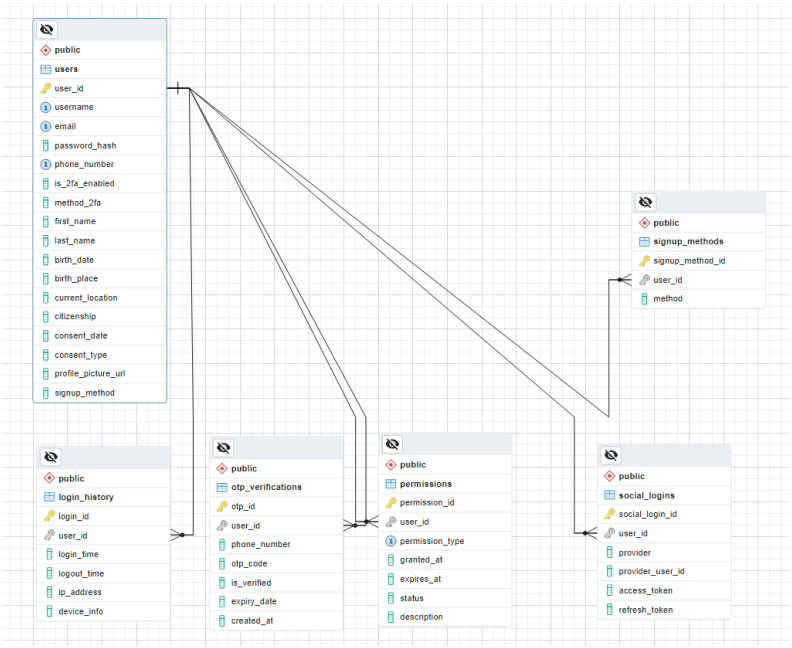
Veritabanı Tasarımı (Database Design) Bölümüne İlişkin Bazı Önemli Notlar:

- **İzinlerin Geçerliliği:** İzinlerin ne zaman verildiği (granted_at) ve ne zaman sona ereceği (expires_at) dikkatlice takip edilmelidir. expires_at sütunundaki değer NULL olduğunda izin süresiz geçerli olabilir.
- **Zaman Damgaları:** TIMESTAMPTZ veri tipi kullanılarak zaman damgaları UTC olarak saklanır. Bu, zaman dilimi farklarından kaynaklanabilecek sorunları önler ve izinlerin zaman dilimi farklarına bağlı olarak doğru bir şekilde yönetilmesini sağlar.
- **İzin Durumu:** status sütunu, izinlerin aktif mi yoksa pasif mi olduğunu gösterir. Bu, izinlerin geçerliliğini kontrol etmek ve yönetmek için önemlidir.

İndeks Yönetimi:

- **user_id:** FOREIGN KEY olarak tanımlandığı için otomatik olarak indekslenebilir. Bu, kullanıcının izin kayıtlarına hızlı erişim sağlar.
- **permission_type:** İzin türlerine göre arama ve sıralama yapılması gerekebilir. Performansı artırmak için permission_type sütununa indeks eklemek yararlı olabilir.
- **İlişkiler:** Tablolar arasındaki ilişkilerin belirlenmesi.
    - **ERD (Entity-Relationship Diagram)**

ERD, veri tabanındaki varlıkların (entities) ve bu varlıklar arasındaki ilişkilerin (relationships) grafiksel bir temsilidir. Veri tabanı tasarımının planlanması ve anlaşılması için kullanılır, varlıkların öznitelikleri ve birbirleriyle olan bağlantılarını gösterir.



4. Drivera Araç Kiralama Uygulaması: Tablo Değişikliklerinin Yönetimi

Drivera araç kiralama uygulamasında, veritabanı tablolarında yapılacak değişikliklerin yönetimi için birkaç alternatif çözüm yolu bulunmaktadır. Uluslararası kullanıcı kitlesi ve büyük veri hacmi göz önüne alındığında, her yöntemin avantajları ve dezavantajları detaylandırılmıştır.

1. Update Yöntemi

Açıklama: Mevcut kayıtları doğrudan günceller. SQL UPDATE komutları ile istenilen veri değişiklikleri yapılır.

- Kullanım Senaryosu: Küçük ve basit veri değişiklikleri, örneğin kullanıcı iletişim bilgileri güncellemeleri.
- Avantajlar: Uygulaması kolay ve hızlı değişiklik sağlar.
- Dezavantajlar: Büyük veri setlerinde performans sorunlarına yol açabilir. Geçmiş verinin izlenmesi ve geri dönüşü zor olabilir.

2. Flag Yöntemi

Açıklama: Kayıtların geçerliliğini flag veya status sütunları ile işaretler. Eski veriyi inactive olarak belirleyip yeni veriler ekler.

- Kullanım Senaryosu: Kullanıcı hesaplarının devre dışı bırakılması, eski araç kiralama bilgilerini arşivleme.
- Avantajlar: Geçmiş verinin korunmasını sağlar; veri geçerliliği ve durumu hakkında net bilgi sunar.
- Dezavantajlar: Ek sütunlar ve kayıtlar gerektirir; büyük veri setlerinde performans etkileri olabilir.

3. Status Label Kullanımı

Açıklama: Kayıtların geçerliliğini status veya is_active gibi sütunlarla yönetir. Aktif/pasif durumları belirler.

- Kullanım Senaryosu: Kullanıcı hesapları, rezervasyonlar gibi aktif/pasif durumların yönetimi.
- Avantajlar: Kayıtların durumunu açıkça belirler; performans yönetimi için etkili olabilir.
- Dezavantajlar: Performans etkileri, özellikle büyük veri setlerinde göz önünde bulundurulmalıdır; ek sorgular gerektirir.

4. Versiyon Kontrolü ve Yedekleme

Açıklama: Veritabanı şeması değişikliklerinin tarihçesi izlenir. Değişiklikler öncesinde veritabanının yedeği alınır.

- Kullanım Senaryosu: Köklü değişiklikler veya kritik veritabanı güncellemeleri.
- Avantajlar: Değişikliklerin geri alınabilirliğini sağlar; veri kaybı önlenir.
- Dezavantajlar: Uygulaması zaman alabilir ve karmaşık olabilir.

Öneriler

Küçük Ölçekli Güncellemeler:

- Update Yöntemi: Basit veri güncellemeleri için uygundur. Performansı izleyin ve gerekirse indeks optimizasyonu yapın.

Geçmiş Veriyi Koruma:

- Flag Yöntemi: Eski kayıtları status sütunu ile işaretleyin (örneğin, inactive). Bu, geçmiş veriyi korurken yeni verilerin eklenmesini sağlar.
- Status Label Kullanımı: status veya is_active sütunları ile verinin geçerliliğini yönetin. Bu yöntem veri bütünlüğünü sağlar ve büyük veri setlerinde performansı iyileştirir.

Büyük Ölçekli Değişiklikler:

- Geçici Tablo Kullanımı: Köklü değişikliklerde geçici bir tablo oluşturup mevcut verileri bu tabloya aktarın, sonra yeni tabloya geçiş yapın. Bu, veri kaybını önler ve performansı artırır.
- Versiyon Kontrolü: Veritabanı değişikliklerini izlemek için versiyon kontrol sistemleri (örneğin, Git) kullanın. Değişikliklerin tarihçesini takip eder ve geri dönüş yapmanıza olanak tanır.

Yetkilendirme ve Erişim:

- Yetkilendirme: Yalnızca yetkili kişilerin değişiklik yapabilmesini sağlayın. Bu veri güvenliğini artırır.
- Erişim Kontrolleri: Tablo değişiklikleri ve veri güncellemeleri için erişim kontrollerini belirleyin. Güvenliği ve kontrolü sağlar.

Drivera araç kiralama uygulaması için, yukarıdaki yöntemlerden hangisinin en uygun olduğunu belirlerken veri hacmi, performans gereksinimleri ve veri güvenliği öncelikli olarak değerlendirilmelidir.