İleri Veritabanı Yönetim Sistemleri Özet

Bölüm 1 - Veritabanına Giriş

VTYS

Kullanıcıları tanımlama, oluşturma, yönetme ve veritabanına erişim kontrolü sağlayan bir yazılım sistemidir.

Neden ortaya çıktı?

Verinin tanımlanması ayrı ve bağımsız bir şekilde saklanmaktan ziyade, uygulama programının içine gömülmüştür. Uygulama haricinde veriye erişim ve kontrol olmadığı için veritabanı ihtiyacı doğmuştur.

Veritabanına kontrollü erişim şunları içerir: güvenlik, bütünlük, eşzamanlılık, kurtarma kontrol sistemleri ve bir kullanıcı erişebilir katolog.

VTYS'nin bileşenleri

- **Hardware**:PC'den bilgisayar ağına kadar değişebilir
- **Software**: VTYS, işletim sistemi, ağ yazılımı(gerekliyse)
- **Data**: Organizasyon tarafından kullanılan ve tanımlanan bu veriye şema denir.
- **Procedures**: Veritabanı ve VTYS'nin kullanımı ve dizaynı için uygulanması gereken komut ve kurallardır.
- **People**

VTYS'nin Avantajları

- Veri tekrarının kontrolü
- Veri tutarlılığı
- Aynı miktardaki veriden daha fazla bilgi etmek
- Verinin paylaşımı

VTYS'nin Dezavantajları

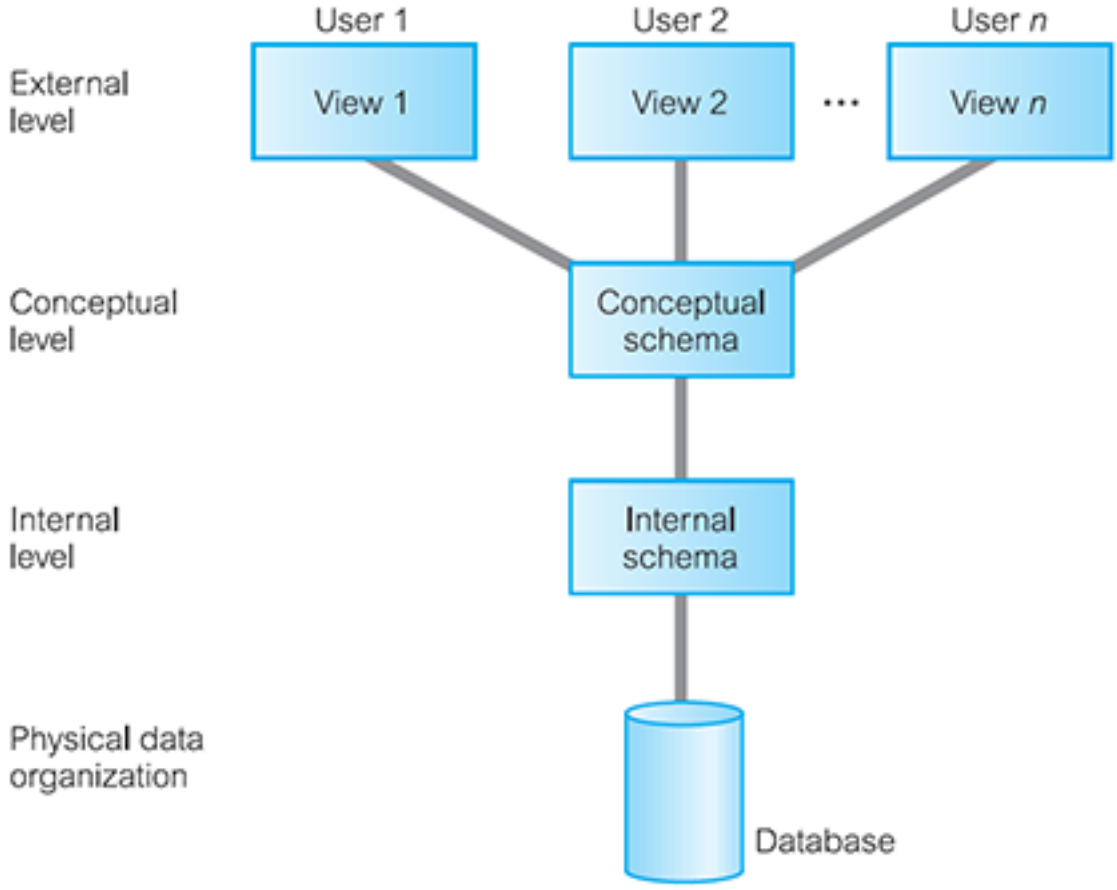
- Karmaşıklık
- Boyut
- VTYS'nin maliyeti
- Ek donanım maliyeti
- Performans

Veritabanındaki Roller

- Veri Yöneticisi
- Veritabanı Yöneticisi
- Veritabanı Tasarımcıları
- Uygulama Programcıları
- Son Kullanıcılar

Bölüm 2 - Veritabanı Ortamı

ANSI-SPARC 3 Katmanlı Mimari



Dış Katman: Veritabanının kullanıcılar tarafından görünümü

Kavramsal Katman: Veritabanının topluluk görünümü. Hangi veri veritabanında saklanıyor ve ilişkileri nedir açıklar.

İç Katman: Veritabanının bilgisayardaki fiziksel sunumu. Verinin veritabanında nasıl saklandığını açıklar.

Fiziksel Veri Bağımsızlığı: Kavramsal şemanın, iç şemadaki değişimden etkilenmemesi

Veri Modeli

Amaç: Veriyi anlaşılabilir bir yolla sunmak

Şunları içerir:

- Nesne tabanlı: Varlık - İlişki, Anlamsal, Fonksiyonel, Nesne yönelimli
- Kayıt tabanlı: İlişkisel Veri Modeli, Ağ veri modeli, Hiyerarşik Veri Modeli
- Fiziksel: Fiziksel Veri Modeli

VTYS'nin Fonksiyonları

- Veri depolama, bulma ve güncelleme
- Bir kullanıcı erişebilir katalog
- Hareket desteği
- Eşzamanlılık kontrol servisleri
- Kurtarma servisleri

Bölüm 3- Veritabanı Mimarileri ve Web Saydamlığı

VTYS'nin Bileşenleri

- Sorgu İşlemci
- Veritabanı Yönetmeni(Database Manager - DM): Yetkilendirme kontrolü, komut işlemci, bütünlük denetimi, sorgu iyileştirici, hareket yöneticisi, zamanlayıcı, kurtarma yöneticisi, tampon yöneticisi, kurtarma yöneticisi
- Dosya Yöneticisi
- DML önişlemci
- DDL derleyici
- Katalog Yöneticisi

Bölüm 5 - İlişkisel Cebir

Çıktı ilişkisel olmaktadır

İşlemler: seçim, izdüşüm, kartezyen çarpım, birleşim, küme farkı, birleştirme, kesişme, bölme

Seçim: σsalary > 10000(Staff)

İzdüşüm: ΠstaffNo, fName, lName, salary(Staff)

Birleşim: Πcity(Branch) ∪ Πcity(PropertyForRent)

Küme Farkı: Πcity(Branch) − Πcity(PropertyForRent)

Theta Join: R  S = σF (R × S)

Aggregation ve grouping: AL (R) : count, sum, avg, min, max

Bölüm 7- Veri Tanımlama

Bütünlük geliştirme özellikleri (IEF)

1. Gerekli veri: `cinsiyet char not null`
2. Alan kısıtlama
3. Referans
4. Genel kısıtlama
5. Varlık bütünlüğü

check Kullanımı

```
1   check (cinsiyet IN ('M','F'))
```

Bir başka örnek:

```
1   create domain CinsiyetTip as CHAR
2   cinsiyet CinsiyetTip NOT NULL;
```

Foreign Key Kullanımı

```
1   foreign Key(branchNo) references Branch
```

Aşağıdaki sql ifadesi ile Staff tablosundan bir satır silindiğinde PropertyForRent tablosundaki **staffNo** değeri otomatik olarak NULL yapılır.

```
foreign key (staffNo) references Staff ON DELETE SET NULL
```

Assertion Kullanımı

```
create assertion StaffNotHandlingTooMuch
check (NOT exists (select staffNo from PropertyForRent group by StaffNo having count(*)>100))
```

View Oluşturma

```
create view Manager3Staff AS select * from Staff where branchNo='BO'
```

Grant Kullanımı

```
grant all privileges on Staff To Manager with grant option
grant select on Branch to Public
revoke select on Branch from Public #yetkiyi kaldırır
revoke all privileges on Staff from Director
```

Bölüm 8 - İleri SQL

Atama İşlemi

```
vStaffNo:='SG14';
select * from Branch where staffNo=vStaffNo;
```

IF Deyimi

```
IF (position='Manager') THEN
salary:=salary*1.05;
ELSE
salary:=salary*1.02;
END IF;
```

CASE Deyimi

```
update staff set salary = CASE
when position='Manager'
then
salary*1.05;
ELSE
salary*1.02
END;
```

LOOP

```
x:=1;
myLoop:
LOOP
x:=x+1;
if(x>3) then exit myLoop;
END LOOP myLoop;
```

FOR Deyimi

```
myLoop:
FOR iStaff AS select count(*) from PropertyForRent where staffNo='SG14' Do..
END For myLoop1;
```

Cursor

Bir sorgu sonucu satırlara teker teker erişebilmeyi sağlar. İşlem tamamlanınca kapatılır. Cursor ile satırlar güncellenebilir

Trigger

Olay: insert, update veya delete

Zamanlama: Before, after veya instead of

Avantajlar: Gereksiz kod ortadan kalkar değişikleri basitleştirir.

Dezavantajlar: Performans gerektirir, basamak etkiler, az taşınabilir.

Bölüm 9 - Nesne İlişkisel Veritabanı

IVTYS'nin Zayıf Yönleri

Gerçek dünyanın zayıf temsili

Anlamsal aşırı yükleme

Integrity için zayıf destek

Sınırlı İşlemler

NIVTYS

Kullanıcı genişletilebilir tipler

Encapsulation, inheritence, polymorphism

Metodların dinamik bağlanması

Dezavantajları

Artan maliyet, karmaşıklık

Satır Tipleri

Değişkenlerde saklanır

Programlara argüman olarak geçer

Fonksiyon çağrılarından dönüş değerleri olarak dönme

Sütun içerisinde satır tanımlama olanağı

Örnek:

```
create table Branch(branchNo char(4), address ROW(street varchar(25), city varchar(15)))
insert into Branch values('B005',ROW('SW1','4EH'))
```

Kullanıcı Tipler

```
create type OwnerNoType as varchar (5) Final;
```

Fonksiyon

```
Function fname(p PersonType) returns varchar(5) p.fName
```

Nesne yaratılması

```
SET P=NEW PersonType();
```

Constructor Tanımı

```
create constructor method PersonType(fn varchar(15), ln varchar(15))
returns PersonType self as Result
BEGIN
SET self.fName=fn;
SET self.lName=ln;
return self;
END;
SET P = new PersonType('John','white');
```

Array Tanımı

```
telno varchar(13) array[3]
select telno[1] from Branch...
```

Alt Tip Tanımı

```
create type StaffType under PersonType As (staffNo varchar(5))
instantiable
not final
instance method isManager() returns boolean;
create instance method isManager()
returns boolean for StaffType
BEGIN
IF self.position='manager' then
return true;
else
return false;
END IF
END
```

Örnek:

```
select s.lName, s.age from Staff s where s.isManager;
```

Bölüm 10- Veritabanı Sistemi Geliştirme Yaşam Döngüsü

Adımlar

1. Veritabanı planlama
2. Sistem tanımı
3. Gereksinimleri toplama ve analiz
4. Veritabanı tasarımı. (Yukarıdan aşağı, aşağıdan yukarı, içeriden dışarı)
5. VTYS seçimi
6. Uygulama tasarımı
7. Prototip (isteğe bağlı)
8. Veri dönüşümü ve yükleme
9. Test
10. Bakım

Veritabanı tasarımı üç aşamadan oluşmaktadır

Kavramsal Tasarım: Verilerin bir modeli oluşturulur

Mantıksal Tasarım: İlişkisel gibi belli bir veri modeli oluşturulur.

Fiziksel Tasarım

Kavramsal veri modeli geliştirilir ve bir mantıksal veri modeli üzerine haritalanır.

VTYS Seçimi

Çalışmanın görev tanımı yapılır

İki ya da üç ürün listesi oluşturulur

Ürünler değerlendirilir

Seçim tavsiye edilir ve rapor yazılır

Veri yöneticisi: Kurumsal veri yönetir

Veritabanı yöneticisi: Kurumsal veritabanını yönetir

Bölüm 11 - Veritabanı Analizi

Olgu Bulma Teknikleri: Sistemler, gereksinimler ve tercihler hakkındaki olguları toplamak için görüşme ve anket gibi teknikleri kullanan süreçtir.

Ne zaman kullanılır?: Veritabanı planlama, sistem tanımı, gereksinimleri toplama ve analiz aşaması

Teknikler: Belgeleri inceleme, Görüşme (En sık kullanılan), kuruluşun işlemde gözlenmesi, araştırma, anket

Bölüm 12 - Varlık / İlişki Modelleme

Entity -> Tablo

İlişki -> Tablolar arasındaki bağıntı

1-> Varlık

->Tipi: Aynı özelliğe sahip nesne kümesi

->Oluşumu: Varlık tipinin benzersis tanımlanabilen nesnesi

2->İlişki

->Tipi: Varlık tipleri arasındaki anlamlı işbirleri kümesi

->Oluşumu

->Tipleri

->Derece: Varlıkların sayısı

->Derece ilişkisi: İki, üç, dört

->Recursive

3->Attribute(Nitelik): Bir valığın veya bir ilişkinin tipinin özellikleri

Nitelik alanı: Bir veya daha fazla nitelik için izin verilen değerlerin kümesi
Basit nitelik: Atomik niteliktir. Bir personelin maaşı daha küçük parçaya bölünemez
Birleşik nitelik: Adres gibi, küçük parçalara bölünebilir
Tek değerli nitelik: Branch entity'sinin branchNo sütunu eşşizdir. Primary key
Çok değerli nitelik: Telefon numarası gibi
Türetilmiş nitelik: Süre değeri gibi. Bitiş-Başlangıç olarak hesaplanınca türetilmiş olmaktadır

4->Keys:

Candidate(Benzersiz): Primary ve composite keyleri temsil eder.

5->Tuzaklar:

->Fan

->Uçurum

Bölüm 14- Normalizasyon

1NF -> Tekrarlayan sütun olmaz

2NF -> Tüm sütunlar primary key'e bağlı olacak. Kısmi bağımlılık remove edilmeli. Bu işlem için yeni tablolar oluşturulur.

3NF -> Transitive (A->B, B->C ise A->C) ilişkisi düzenlenir.

Bölüm 16- Kavramsal Veritabanı Tasarımı

Üç ana faz:

Kavramsal DB tasarımı

Mantıksal DB tasarımı

Fiziksel DB tasarımı

Kavramsal Veritabanı Tasarımı: Tüm fiziksel hususlardan bağımsız bir işletmede kullanılan verilerin bir modelini oluşturma süreci

Mantıksal Veritabanı Tasarımı: Belirli bir VTYS ve diğer fiziksel hususlardan bağımsız, belirli bir veri modeline dayalı (Örn: ilişkisel), bir işletmede kullanılan verilerin bir modelini oluşturma süreci

Veritabanı tasarımında kritik başarı faktörleri

Mümkün olduğunca kullanıcı ile etkileşimli çalışmak

Bir veri odaklı yaklaşım kullanmak

Veri modeli diyagramlarını tamamlamak için bir veri sözlüğü oluşturmak

Kavramsal Veritabanı Tasarımı Adımlar: Mantıksal veritabanı modeli için ilişkileri türetmek, normalleştirme kullanarak ilişkileri doğrulamak, bütünlük kısıtlamaları tanımlamak.

Fiziksel DB tasarım Adımları: Temel ilişkileri tasarlamak, türetilen verilerin gösterimini tasarlamak, genel kısıtmalamarı tanımlamak

Bölüm 24 - Dağıtık Veritabanı Yönetim Sistemleri

Dağıtık işleme: Bir bilgisayar ağı üzerinde erişilebilir bir merkezi veritabanı

Paralel VTYS

->paylaşılan

->hafıza

->disk

->hiçbir şey

Avantajlar: Yerel özerklik, kullanılabilirlik, güvenirlik, performans,ekonomik

Dezavantajlar: Karmaşıklık, tecrübe eksikliği, maliyet, güvenlik

Tipleri:

Homojen: Tüm siteler aynı VTYS(Veritabanı Yönetim Sistemi) ürününü kullanır

Hetorojen: Tüm siteler farklı VTYS ürününü kullanır

Çoklu veritabanı sistemi: Lokal olarak bağımsız veritabanı

Nelere Sahip Olmalıdır?

->Genişletilmiş

->İletişim Hizmetleri

->Veri Sözlüğü

->Sorgu işleme

->Eş zamanlılık kontrolü

->Kurtarma hizmetleri

Tasarım

->**Parçalanma**: İlişki sonra dağıtılsın diye alt ilişkilere bölünür. Yatay, dikey, karışık, türetilmiş türlerine sahiptir

->**Tahsis**(allocation): Her parça optimum dağıtım ile sitelere dağıtılır

->**Kopya**: Parçanın kopyası çeşitli sitelerde bulunabilir

Tasarım Metodolojisi

Veritabanların yer alacağı yeri belirlemek için topoloji çıkarılır.

En önemli hareketler analiz edilip, yatay, dikey şekillerde tablolar parçalanır

Saydamlıklar

Dağıtım Saydamlığı: Veritabanı dağıtık olmasına karşın kullanıcılar tek bir veritabanı olarak görmesidir

->Parçalanma, Konum, çoğaltma, yerel haritalama, adlandırma

Haraket Saydamlığı

->Eş zamanlılık

->Başarısızlık

Adlandırma Saydamlığı: Her öğe farklı isimde olmalı

Eş zamanlılık: Tüm hareketler keyfi seri sırayla, teker teker çalıştırılırken elde edilen sonuçlar mantıksal olarak tutarlı olmalıdır.

Performans Saydamlığı: DVTYS sanki bir merkezi VTYS' imiş gibi çalışır

DVTYS için Date'in 12 Kuralı

0. Temel İlke: Bir dağıtık sistem kullanıcıya tam bir dağıtık olmayan sistem gibi görünmelidir
1. Yerel Özerklik
2. Merkezi bir sitede güven yok
3. Sürekli çalışma
4. Yer bağımsızlığı
5. Parçalanma bağımsızlığı
6. Çoğaltma bağımsızlığı
7. Dağıtık sorgu işleme

8. Dağıtık hareket işleme
9. Donanım bağımsızlığı
10. İşletim sistemi bağımsızlığı
11. Ağ bağımsızlığı
12. Veritabanı bağımsızlığı