YBS Ansiklopedi

www.YBSAnsiklopedi.com

Cilt 4, Say 3, Eylül 2017

ETL Süreçleri (ETL Processes)

Eda Eşmekaya[1], Şadi Evren ŞEKER[2]

1. *Süleyman Demirel Üniversitesi, Bilgisayar Mühendisliği Bölümü*
2. *İstanbul Şehir Üniversitesi, Yönetim Bilişim Sistemleri Bölümü*

Özet

Bu makalenin amacı, literatürde çıkarım, dönüşüm ve yükleme olarak geçen kavramları açıklamak ve her aşamasında yaşanan problemleri, bu problemlere karşı geliştirilen jenerik çözümleri ve hem endüstride hem de akademik ve bilimsel çalışmalarda sık kullanılan çıkarım, dönüşüm ve yükleme yöntemlerini açıklamak, bu yöntemleri karşılaştırarak avantaj dezavantajlarını ortaya koymaktır. Ayrıca ETL bir süreç olarak ele alındığında bu sürecin yönetilmesi ve bir yaşam döngüsü olarak ele alınması da mümkündür. Bu yazı kapsamında 3 aşamanın detaylarına ilave olarak ETL bir süreç şeklinde ele alınmış ve farklı durumlardaki kullanımlarına örnekler de verilmiştir. Son olarak, ETL sürecinin sonunda hem başarı, hem performans hem de bazı metriklerin çıkarılması için kritik öneme sahip veri doğrulama adımından da bu makalede bahsedilmiştir.

Anahtar Kelimeler:*ayrıştırma,dönüştürme,yükleme ,veri tabanı , veri madenciliği, veri ambarı*

Abstract

The aim of this article is to explain, extract, transform and load steps of ETL and explain some generic solutions for some well-known problem types. ETL is a method, widely used in the industry and academic or scientific studies and in this paper we also mention about the well known methods for solving the problems together with their advantages and disadvantages. It is possible to consider ETL as a process so the management of process or considering this process as a lifecycle is also possible. Within the scope of this article, besides the details of three ETL steps, their daily applications are explained over samples. Finally, the last step of ETL process is none of these three steps but the data validation step which is also useful to measure the success, performance and also yield some metrics.

Keywords:*extract,transform,load,database,data mining, data warehouse*

1. Giriş

Literatürde çıkarım, dönüşüm ve yükleme (extract, transform, load ETL) olarak geçen kavram, uzun yıllardır kullanılan bir veri işleme yöntemidir. Özellikle veri ambarı, veri tabanı, veri bilimi, gibi alanlarda sıkça kullanılan bu kavram, verinin bir ortamdan çıkarılıp başka ortama geçirilmesi (extract), işlemeye uygun şekle dönüştürülmesi (transform) ve verinin işleme ortamına yüklenmesi (load) aşamalarından oluşan geleneksel bir yöntemdir. Günümüzde, hemen her ölçekte ve her alanda sıkça kullanılan bir yöntem olmasına rağmen çokça yeni problem ve yeni çözüm yöntemi çıkmasına müsait bir alandır. Herhangi bir veri madenciliği (data mining, DM) veya iş zekası (business intelligence, BI) projesinin maliyeti, proje için görevlendirilen mimarinin kararından

doğrudan etkilenir. ETL süreçlerinde bazı seçimler, problemin yapısı itibariyle zorunlu olurken bazı seçimler de stratejik kararlar, teknoloji yönetimi veya projenin yöntemi süreçleri ile ilgili sebeplerle uzmanların seçimine açık olabilmektedir. Bu yazıda ETL süreçlerinin yanında alternatif olarak hızla gelişen çıkarım, yükleme ve dönüşüm sırasıyla ilerleyen (extract, load, transform ELT) süreçleri de ele alınacaktır. Seçim yapılırken, proje için belirlenen amaçlara ve her iki yaklaşımın çeşitli güçlü ve zayıf yönlerini dikkate alarak yapılmalıdır [1].

ETL her aşamasında problemler içeren bir süreç olmakla birlikte, sürecin nasıl yönetileceği ile ilgili ilave problemler de bulunmaktadır. Örneğin çok farklı kaynaklardan aynı anda ETL süreçlerini işletmek için sürecin paralelleştirilmesi ve kaynaklardan birisinden çıkarım yapılırken bir diğerinden gelen verinin dönüşümü ve diğer bir kaynaktan gelen verinin ise yüklenmesi mümkün olabilmektedir [2]. Günümüzde gelişen büyük veri teknolojileri ile haritalama ve indirgeme (map reduce) işlemlerinin veri işleme üzerindeki etkisi daha da önemli bir hal almıştır ve bu yüzden ETL süreçlerinin paralleleştirilmesi ve haritalama indirgeme süreçleri ile entegre olması söz konusudur [3].

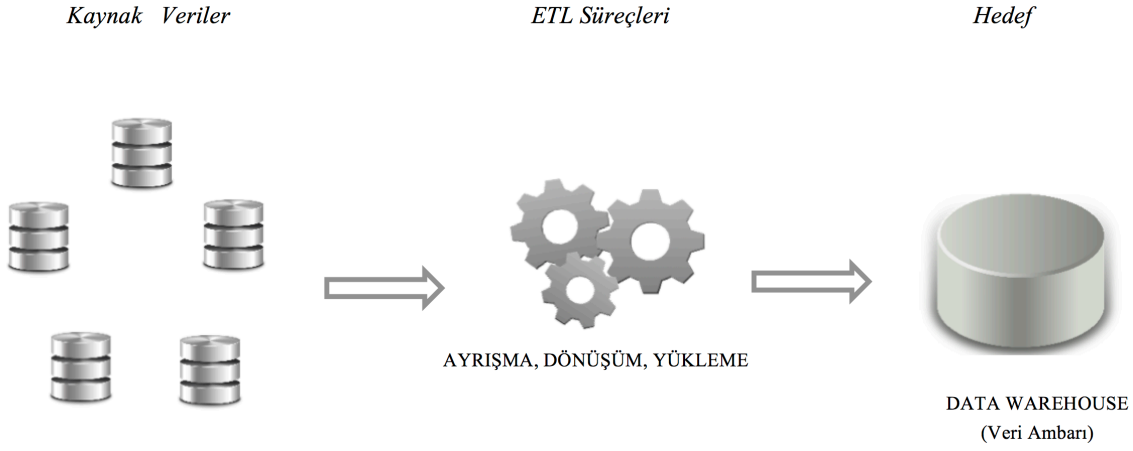
Büyük veri çalışmalarının getirdiği, diğer bir yenilik ise, verilerin yapısal olmayan kaynaklardan da alınma zorunluluğudur. Örneğin sosyal ağlar, blog'lar, wiki'ler gibi çok farklı kaynaklardan verilerin alınarak işlenmesi gerekmektedir [4]. Ayrıca bu veriler zamana bağlı olarak sürekli ve değişik hızlarda akan (velocity) , büyük hacimlerde (volume), farklı kaynaklardan gelen (variety) ve varlığı zaman içerisinde değişen (veracity) yapılarda olduğu için, ETL süreçlerinin, bütün bu problemleri çözecek şekilde evirilmesi de gerekmektedir. Özellikle bu alanda akan veri madenciliği (streaming data mining) teknikleri gelişmiş [5] ve zamana [6] ve mekana [7] bağlı yeni mantıksal çerçeveler içerisinde ele alınır hale gelmiştir.

Bu yazı kapsamında, klasik ETL süreçleri ele alınacak, bir verinin kaynağından alınarak işlenebilir hale getirilmesi sırasında geçeceği adımlar incelenecek ve büyük veri dünyasındaki yenilikler açıklanacaktır.

2. ETL (Extract, Transform, Load) (Verinin Ayrıştırılması, Dönüştürülmesi ve Yüklenmesi)

ETL, bir veri kaynağından veya birçok veri kaynaklarından gelen verileri bir veri ambarına yerleştirmekten sorumlu ve veri ambarı üzerinde çalışan bir işlem sürecidir. Bu işlem sırasında veriler, bir kaynak sistemden alınır, analiz edilebilir bir formatta dönüştürülür ve bir veri ambarına veya başka bir sisteme depolanır.

ETL, bütün veri ambarı dönüşümü, veri madenciliği, veri bilimi ve yapay zeka uygulamaları için gereklidir.



Şekil 1- ETL Süreci, Genel Görünüm

Şekil 1'de de gösterilen, ETL süreci temel olarak 3 adımdan oluşur ve bu adımlar kısaca aşağıdaki gibidir:

Extract (Ayrışma, Çıkarım): Verinin genelde yapısal olan kaynaklardan alınarak ayıklanması sürecidir. Genelde veri ambarı veya büyük veri dönüşümlerinde bütün verilere ihtiyaç duyulmaz. Sadece ihtiyaç duyulan verinin alınarak temiz bir şekilde aktarılması ayrışma aşamsını oluşturur. Genelde veri ambarı uygulamalarında yapısa, büyük veri uygulamalarında ise yapısal / yapısal olmayan veri kaynaklarının taranabilir olması hedeflenir.

Dönüşüm (Transform): Verinin işlenebilir ve istenen yapıya dönüştürülmesi sürecidir. Verinin belirli kurallara uygun hale getirilmesi, bazı yardımcı liste ve tabloların kullanılarak (lookup tables) verinin dönüştürülmesi veya farklı kaynaklardan gelen verinin birleştirilmesi gibi işlemler bu aşamada yapılır.

Yükleme (Load): Verinin hedef bir veri tabanına yüklenmesi işlemidir.

3. E (Extract – Ayrışma – Çıkarım)

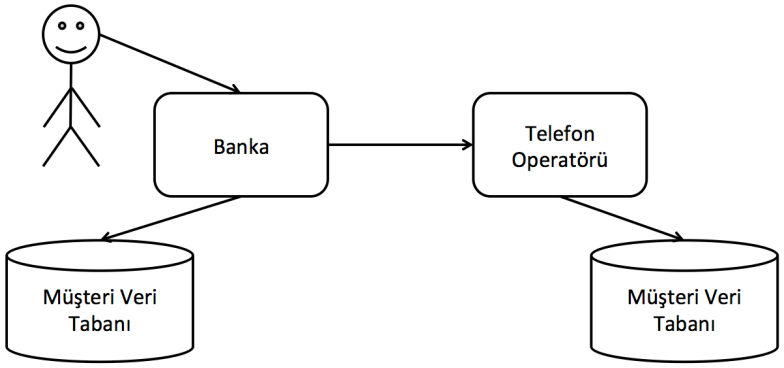
Kaynak sistemdeki verinin alınıp daha sonraki adımlarda verinin erişilebilir şekle getirilmesidir. Klasik veri ambarı yaklaşımlarında genellikle yapısal olan ve farklı kaynaklardan alınan veri tek bir sistem üzerinde toplanır ve farklı kaynaklardan çakışan özellikler kontrol veya düzeltme amacıyla kullanılır. Örneğin internet üzerinde satış yapan bir perakende firmasını, web üzerinden aldığı müşteri kayıtları ve üyelik kartları (sadakat kartları, loaylity cards) kayıtlarında bulunan adres bilgilerinin tutuşup tutuşmadığının kontrol edilmesi ve ihtiyaca göre güncellemelerin yapılarak en son adres bilgisine ulaşılması bu çapraz sorgulamanın ve kontrolün bir çeşididir. Ayrışma aşamasında verinin kirli (veya gürültülü) olması da sık karşılaşılan problemlerdendir. Örneğin yaş hanesinde 3000 gibi yüksek sayıların yazması veya eksi değerlerin bulunması bir hatadır ve bunları kirli veri olarak kabul edebiliriz. Literatürde her ne kadar kirli ve gürültülü veri birbiri yerine kullanılabiliyor olsa da, genelde kirli veri her çeşit veride yaşanan problemleri ifade ederken gürültülü veri, bir sinyal işleme terimidir ve sinyal haline getirilebilen ve sinyal işleme yöntemleri ile analiz edilebilen verilerdeki problemleri ifade etmektedir. Örneğin bir zaman serisi olarak borsanın kapanış değerlerini ele alırsak, bu veriyi sinyal işleme yöntemleri ile işleyebilir ve bu veri üzerinde gürültüden bahsedebiliriz.

3.1. Mantıksal Çıkarım Metotları

Çıkarım yapma süreci farklı şekillerde işletilebilir. Aşağıda sık kullanılan 3 farklı çıkarım yöntemi açıklanmıştır:

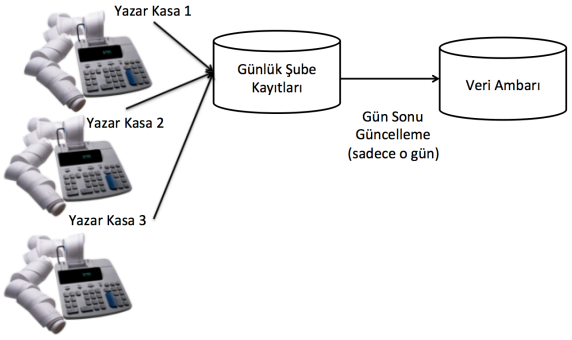
Güncelleme bildirimi (update notification): bir sistemde değişiklik oluğunda ve bu değişiklik bir güncelleme gerektirdiğinde, sistemin çıkarım aşamasına bir bildirim göndermesi anlamına gelir. Örneğin, bir bankanın, Telekom firmaları ile senkronize şekilde çalıştığını kabul edelim ve bankaya giriş sırasında veya kritik işler sırasında müşterilere SMS mesajı ile ulaştığını düşünelim. Bu durumda banka, müşterilerinin Telekom operatörleri ile ilgili bilgileri tutmak ve bunları güncellemek zorundadır. Bir müşterinin Telekom firmasında bir değişiklik olduğunda, örneğin telefon hattı kapatıldığında bu bilginin banka tarafında da güncellenmesi gerekir. Telekom firması müşterilerinin bu işlemlerini bankaya otomatik olarak bildiriyorsa, bu tip bildirimlere, güncelleme bildirimleri denir ve çıkarım (extract) işleminin tipini güncelleme bildirimi olarak kabul edebiliriz.

Yukarıdaki bu örneğin tam tersi de düşünülebilir. Örneğin ödeme sistemi için bir bankayı kullanan Telekom operatörü bankadaki güncellemeleri otomatik olarak almak istiyor olabilir. Bankanın Telekom firmasına otomatik olarak güncellemeleri gönderme imkanı olması durumunda ise, senaryonun Şekil 2'de görselleştirilmiş hali ortaya çıkacaktır. Bu örnekte bankada bir güncelleme yapan müşterinin hem bankada tutulan bilgileri, banka veri tabanında güncellenmiş hem de Telekom firmasına güncelleme bildirimi gönderilmiştir. Dolayısıyla Telekom firmasında tutulan veri ambarının gelen güncelleme sonunda otomatik olarak veri çıkarımı yapacağı kabul edilebilir.



Şekil 2: Banka ve Telefon Operatörü arasında güncelleme bildirimi senaryosu

Artırımlı Çıkarım (Incremental Extract): Bazı sistemlerde, değişiklik güncellemeleri otomatik olarak çıkarılamıyor veya çıkarılmak istenmiyor olabilir. Genelde banka, Telekom firmaları gibi otonom sistemlerin birbirini güncellemesi yerine kendi otonomilerini koruması bir tasarım kararı olabilir. Veya teknik bazı kısıtlardan dolayı da bu güncellemeler yapılamıyor olabilir. Örneğin bir perakende satış firmasının yazarkasalarından gelen verilerin sürekli ve anlık güncelleme yapması teknik bazı zorluklar içerebilir. Bunun yerine genelde sistemlerin verilerinin depolandığı geçici çözümler üretilir. Bilgisayar bilimlerinde tampon bellek (buffer) veya ön bellek (cache) olarak geliştirilen çözümlere benzer şekilde veriler geçici bir depolama biriminde (storage) tutularak belirli aralıklarla ana sisteme veya veri ambarına güncelleme geçilebilir. Bu sistemlerin en büyük özelliği, belirli bir zaman periyodu için güncelleme yapmalarıdır. Örneğin yazar kasaların bütün gün boyunca yaptıkları satışların gün sonunda ana sisteme yüklenmesi ve dolayısıyla sadece 1 günlük verinin tutuluyor olması ve dolayısıyla firmanın kurulduğu tarihten itibaren bütün satışların geçmesi yerine sadece 1 günlük verinin artırımlı olarak ana sistemde güncellenmesi söz konusudur.



Şekil 3: Artırımlı çıkarım örneği olarak yazar kasa bilgilerinin gün sonu güncellemeleri

Örneğin, Şekil 3'te gösterildiği gibi 3 farklı yazar kasa bilgisi gün içerisinde sürekli olarak şubedeki geçici kayıtları güncellemekte ve gün sonunda şubede tutulan kayıtlar veri ambarına aktarılmaktadır. Yüzlerce şubesi olan bir işletme için gün sonunda bütün şubeler benzer şekilde o günlük kayıtları aktarabilmektedir. Ayrıca geçmişe yönelik olarak istenilen periyotta bilgi veri ambarında aktif olarak durabilir. Örneğin son 3 yıllık bütün yazar kasa kayıtları ana sistemde tutulup bunlar üzerinden rapor alınabilir. Artırımlı çıkarım senaryolarında genelde silme işlemleri kaybedilir. Örneğin gün içerisinde yazar kasalarda bir iptal veya iade yaşanması durumunda, özel bir tasarım olmadıkça, bu verilerin tutulması anlamsızdır ve güncelleme bildirimlerine göre bu açıdan ya veri içeriğinde kayıplar yaşanmasına açıktır ya da özel bir dikkat gerektirir.

Tam Çıkarım (Full Extract): Bazı sistemlerde değişimin algılanması mümkün olmamaktadır. Genelde veri ambarlarının ilk kurulumu sırasında çok sayıda farklı kaynaktan verinin aktarılması ve o zamana kadar olan verilerin yeni sisteme geçirilmesi bu tip bir tam çıkarım gerektirir. Sistemde bulunan bütün veriler çıkarılarak yeni sisteme taşınır. Örneğin bir veri tabanındaki bir tablonun tamamının üzerinde çıkarım işleminin çalışması veya bir dosyanın tamamının sisteme taşınması bu şekilde çıkarımlar olarak görülebilir. Bazan de farklı sistemlerden gelen veriler teknik kısıtlar yüzünden ancak tam çıkarımla alınabilmektedir. Örneğin bir eski sistem (legacy system) olarak müdahalesi mümkün olmayan müşteri ilişkileri sistemi veya stok yönetim sistemi sadece tam çıktı verebiliyorsa bu çıktı ya tamamen işlenerek sisteme alınacak veya öncesinde bir ön işlemeden geçirilerek yeni olan kısımlar bulunacak ve artırımlı çıkarıma dönüştürülecektir. Her iki durumda da tam çıkarım yapılmış sayılabilir.

3.2. Fiziksel Çıkarım Metotları

Çıkarım (extract) işlemi fiziksel seviyedeki duruma göre de iki farklı şekilde incelenebilir. Bu yöntemler, literatürde çevrim içi (online) ve çevrim dışı (offline) çıkarım olarak geçmektedir.

Çevrimiçi Çıkarım (Online Extraction): Veri doğrudan kaynak sistem içerisinden çıkarılır. Kaynak sistem fiziksel olarak hedef sistemle aynı yerde veya farklı yerlerde, aynı yapıda veya farklı yapılarda olabilir. Hatta, iki sistem arasında ara sistemler ve veri depoları da bulunabilir. Ancak amaç kaynak sistemdeki bir değişimin çevrimiçi (online) olarak hedef sisteme iletilmesidir. Örneğin Şekil 2'de gösterilen banka ve Telekom firması örneğindeki bağlantının çevrim içi olarak aktarılması senaryosunu düşünebilirsiniz. Genelde bu iletişim, uygulama programlamanabilir arayüzleri (application programmable interface (API)), Web Servisleri, basit nesne erişim protokolleri (simple object Access protocol SOAP) veya, orta katman yazılımlar (middleware) gibi çözümler üzerinden işlemektedir.

Çevrimiçi çıkarımda bir tasarım problemi olarak, verinin ve verideki değişiklikten sorumlu varlıkları (entities) doğrudan veri ambarına erişmesi veya arada farklı katmanlar ve veri depolarının bulunması gibi tercihler bulunur. Örneğin Şekil 3'teki yazar kasaların doğrudan veri ambarına erişerek veri güncellemeleri yapması mümkünken araya ilave bir katman kurulup her şubedeki veri toplanarak, toplanmış verinin aktarılması da mümkündür. Burada problemler değişik seviyelere aktarılmış olur. Örneğin ürün iadesi yapan müşterilerin ortalama iade sürelerinin bulunması istenen bir senaryoda, bir yazarkasadaki satışın aynı yazarkasada iadesi, şubedeki farklı bir yazarkasada iadesi veya farklı bir şubedeki iadesi ve satış zamanının arasındaki farkın yakalanması, farklı seviyelerde işlem gerektirir ve bu farklar tasarım ile ilgili detaylardır.

Çevrimdışı çıkarım (offline extraction): Genelde sistemlerin anlık iletişiminin mümkün olmadığı durumlarda sistemlerin birbirine çevrimdışı iletimleriyle sağlanan çıkarım yöntemleridir. Bu sistemlerde de verinin çıkarımı ile ilgili kuralların tanımı, çıkarılacak olan verinin yapısı ve diğer bütün detaylar ön tanımlı olabilir. Çok kullanılan bazı çevrimdışı çıkarım yöntemleri: düz dosyalar (flat files), kayıtlar (redo veya arşiv kayıtları olabilir, sistemdeki hareketlerin dökümü olabilir), XML, CSV veya JSON formatındaki çıktılar olabilir. Bu dosya yapıları çevrim içi çıkarımda da kullanılabilmekle birlikte, çevrimdışı çıkarımın en önemli farkı, bu dosyaların belirlenen

periyot için çıkarımının alınarak sistemler arasında çevrimdışı taşınmasıdır. Çok daha klasik bir örnek, çevrim dışı yedekleme sistemlerinden getirilen veriler olarak düşünülebilir.

3.3. Değişim ve Algısı

Çıkarım süreçlerindeki en önemli konulardan birisi de değişim kavramı ve bu kavramın sistem tarafından doğru algılanmasıdır. Örneğin verilerde değişim var mı? Bu değişim ne zaman oldu? Hangi veriler değişti? Gibi soruları sorabiliyor ve doğru cevaplayabiliyor olmamız gerekir. Bütün çıkarım yöntemleri için bu değişim bilgisinin algılanması önemlidir. Aksi halde güncel bilginin kaybolması veya sistemde tutarsız verilerin çıkması gibi problemler ortaya çıkabilir.

Genel olarak 3 temel çözüm stratejisi sunulabilir, bunlar zaman etiketleri (timestamps), zamanlanmış veri bölütleri ve güncelleme çözümleridir.

Zaman Etiketleri (timestamps): Temel olarak verinin yanında işlem zamanını etiketlemek mantığına dayanmaktadır. Büyük veri dünyasında da sıkça başvurulan bir çözümdür ve veriyi bütün sistemler için kabul edilebilir bir zaman etiketi ile etiketler (saat farkı olması durumları göz önüne alınmalıdır). Bu sayede en güncel veri veya veri üzerindeki işlem zamanına erişmek mümkün olacaktır.

Zamanlanmış veri bölütleri veya indeksler (Clusters, Partitions, Fragments): Farklı teknolojilerde farklı isimlerle geçen bu yaklaşımda ise, verinin zamana göre farklı bir bölütte tutulması hedeflenir. Örneğin veri tabanındaki bir tablonun, işlem zamanına göre alt parçalara bölündüğünü ve her haftanın verisinin farklı bir bölütte tutulduğunu düşünelim. Bu durumda, veri ambarı güncelleme stratejisi ilgili bölüte erişerek işlem yapacak ve zaman farkı veri işlemeye dahil edilebilmiş olacaktır. Benzer şekilde zamana bağlı indeksler oluşturulması da veri tabanı ve veri tabanı tabloları üzerinde işlem hızı ve verinin zaman algısını beraberinde getirecektir.

Güncelleme Çözümleri (Updates): Çoğu veri tabanı teknolojisi, veri tabanındaki değişikliklerin farklı aksiyonları tetiklemesine imkan verir. Çoğu veritabanında tetik (trigger) olarak geçen bu kavram sayesinde herhangi bir güncelleme veri tabanı seviyesinde diğer güncellemeleri de tetikleyebilir. Ayrıca veri tabanı seviyesine gelmeden, uygulama katmanında da benzer çözümleri gerçeklemek mümkün olabilmektedir. Amaç, verinin değiştirilmesi, eklenmesi veya silinmesi gibi durumlarda veri ambarını ilgilendiren çıkarım süreçlerini tetiklemektir.

4. Veri Temizleme (Data Cleaning)

Genel olarak çıkarım (extract) aşamasının bir parçası olmasına karşılık, veri çıkarımı (extract) ile veri dönüşümü (transform) arasında bir katman olarak da düşünülebilir ve her iki tarafı da ilgilendiren adımlar barındırır. Genel olarak verinin kirli veya gürültülü olma ihtimalini göz önünde bulundurmak zorundayız ve dolayısıyla bu verilerin sistemde temizlenmesi, çıkacak sistemin başarısı için kritik önemdedir. Çok basit bir örnek olarak bir sınıftaki öğrencilerin yaşlarını hesapladığımızı ve -3000 yaşında birisinin kayıtlı olduğunu düşünelim. Hatalı giriş yapıldığı kesin olan bu satırı hesaplamalara katarsak muhtemelen sınıftaki kişilerin yaş ortalaması oldukça hatalı çıkacaktır. Bunun yerine basit bir kontrol ile, bu şekilde hatalı olduğu kesin olan verilerin temizlenmesi sistemin çok daha doğru sonuçlara ulaşmasını sağlayacaktır.

Veri temizleme aşamasında, önemli konulardan birisi de eksik verilerdir. Çeşitli sebeplerle bütün veriler sisteme girilmemiş veya girilen verilerin hatalı olduğu tespit edilmiş olabilir. Eksik verilerin düzeltilmesi için literatürde töhmet (imputation) olarak geçen yaklaşımlar kullanılabilir [9].

Veri temizleme aşamasında genel olarak kural tabanlı yaklaşımlar kullanılır (rule based approach). Örneğin yaş hanesinde eksi değer yazılamayacağı bir kuraldır ve sistemde buna benzer çok sayıda kural uzmanlar tarafından sisteme yüklenir. Büyük veri çalışmaları ve makine öğrenmesi ve veri analitiği gibi yöntemlerin yaygın kullanımı ile bu kural tabanlı yaklaşımların da artık makineler tarafından kuralların otomatik çıkarıldığı yapılara dönüşmeye başladığını görüyoruz. Ancak bu yazının kapsamının dışına çıkmamak adına, klasik olarak bilinen bazı kurallara örnekler vererek veri temizleme konusunu biraz daha açalım:

Verilerin tek ve eşsiz şekilde ifade edilmesi: Örneğin Kadın/Erkek, K/E ifadelerinin birleştirilmesi, cinsiyetin bilinmediği durumlara izin verilip verilmemesi K/E/B şeklinde ilave durumların nasıl çözüleceği gibi kuralların belirlenmesi.

Veri tabanında null olarak geçen kavramların belirli bir standarda oturtulması: Örneğin yaş bilgisinin null olması durumu ile belirtilmemiş olması durumun eşitlenmesi ve aynı şekilde ifade edilebiliyor olması ve bütün null değerlerin anlamlı bir dönüşüme sokulması.

Veri gösterimlerinin standartlaştırılması. Örneğin telefon numarası gibi bilgilerin belirlenen formata çevrilmesi (nümerik veya dizgi (string)) ve bütün sistemin buradaki uygun yapıya dönüştürülmesi: +9 0 555 55 55 veya +90 555-55-55 veya 5555555 gibi gösterimlerin tek bir gösterim altında birleştirilerek standartlaştırılması

Verilerin doğrulanabilir bir yapıya oturtularak standartlaştırılması. Örneğin adres alanındaki sokak/sk. Veya cadde/cad./cd./cad/cd gibi farklı gösterimlerin tek bir yapıya indirgenmesi veya farklı yapıların desteklenmesi durumunda bütün bu yapıların aynı şeyi ifade ettiği bir doğrulama sisteminin geliştirilmesi.

Verilerin içeriklerinin doğrulanması. Örneğin mahalle/ilçe/şehir/ülke bağlantısının doğru olması. Örneğin İzmir Göztepe ile İstanbul Göztepe eşleşmelerinin doğru yapılması Göztepe/Kadıköy/İstanbul veya Göztepe/konak/İzmir şeklinde.

Yukarıdaki bu kurallara ilave olarak sistemde marjinal verilerin yakalanması (outlier detectino), tekrarlı rutin örüntülerin çıkarılması (pattern), iş süreçlerinin otomatik olarak algılanarak bu süreçlerin dışına çıkan durumların tespiti (automatic business process modeling) gibi çok sayıda makine öğrenmesi yaklaşımı ile kuralların otomatik çıkarılması mümkündür.

5. T(Transform- Dönüşüm)

Veri dönüştürme aşamasında, çıkarılan verilere son hedefe yüklenmek üzere hazırlanmak için bir dizi kural veya işlev uygulanır. Bazı veriler hiç dönüşüm gerektirmez; bu tür veriler "doğrudan hareket" veya "geçiş" verileri olarak bilinir. Dönüşümün önemli bir işlevi, yalnızca "doğru" verileri hedefe ulaştırılması ve bu amaçla verilerin temizlenmesidir.

Diğer durumlarda, sunucu veya veri ambarının iş ve teknik gereksinimlerini karşılamak için aşağıdaki dönüşüm türlerinden bir veya daha fazlası gerekebilir.

Veri dönüşüm işlemlerini farklı açılardan kategorize etmek mümkündür. Örneğin, **mimari olarak veri dönüşüm türleri (Architectural Data Transformation Types)** aşağıdaki iki grupta toplanabilir:

- **Çok sahneli veri dönüşümü (multi-stage data transformation):** Bu yaklaşımda ayrışma, dönüşüm ve yükleme işlemleri sırasıyla izlenir. Çıkarılan veriler, depoya yerleştirilmeden önce dönüşümlerin gerçekleştiği bir hazırlama alanına taşınır.

- **Ambar içi veri dönüşümü (In warehouse data transformation):** Bu yaklaşımda, veriler ayıklanır ve analitik depoya yüklenir ve orada dönüşümler yapılır.

Benzer şekilde veri dönüşüm yöntemlerini temel ve gelişmiş dönüşümler olarak iki kategoride toplamak da mümkündür. Bu kategorileri ve altındaki bazı dönüşüm tiplerini aşağıdaki şekilde listeleyebiliriz:

Temel Dönüşümler (Basic transformations)

- **Temizleme (Cleaning) :** ”NULL” değeri “0 ” değerle , “Erkek” değerinin “E” değerle ,”Kadın” değerinin “K” değeri ile eşleşmesi ve zaman formatlarının tutarlılığı gibi veri temizlemelerine örnek verilebilir.
- **Tekilleştirme (Deduplication) :** Yinelenen kayıtları saptama ve kaldırma işlemleri gerçekleştirilir.
- b Karakter küme dönüştürme, ölçü birimi dönüştürme, tarih / saat dönüştürme, vb. işlemleri tekrar gözden geçirilip düzeltilmesidir.
- **Yeniden Yapılandıran Anahtar (Key Restructuring):** Tabloların diğer tablolarla ilişki kurması için gereken anahtarlar (yabancıl anahtar, foreign key) ve birincil anahtar (primary key) bağlantılarının dönüştürülmesi sağlanır.

Gelişmiş Dönüşümler (Advenced Transform)

- **Türetme (Derivation) :**Mevcut veriler üzerinden bazı yeni verilerin hesaplanmasını sağlar. Örneğin müşteri veri tabanlarının birleştirildiği bir yapıda, bir sistemde doğum tarihi diğer sistemde ise yaş bilgisi tutuluyor olabilir. Yaş bilgisi tutulan sistemde, müşterinin kayıt tarihi ve yaş bilgisinden, müşterinin doğum tarihinin hesaplanması ve diğer sistemlerle uyumlu hale getirilmesi sırasında iki farklı bilgiden (kayıt tarihi ve yaş) doğum tarihi hesaplanmış ve türetilmiştir.
- **Filtreleme (Filtering) :**Yalnızca belirli satır / sütunların seçilmesidir. Örneğin geliştirilen veri ambarı veya veri bilimi projesi için gereksiz olan verilerin alınmayarak sadece belirli özellikleri taşıyan verilerin alınması sürecidir.
- **Birleştirme (Joining) :** Ortak alanlarla ilgili verileri birleştirerek tek bir alanda sunan sistemlerdir. Örneğin web sitesinde kayıt olan müşteriler veya sadakat kartı (üyelik kartı) çıkaran müşterilerin veri tabanlarının birleştirilerek tek bir sistem haline getirilmesi, bu birleştirme işlemine bir örnek olarak düşünülebilir.
- **Bölme (Splitting) :** Tek bir sütunun birden çok sütuna bölünmesi işlemidir. Örneğin adres alanında serbest şekilde yazılmış, il, ilçe veya ülke gibi verilerin ayrı kolonlara bölünerek sistemde ayrı tutulması sırasında tek bir adres alanı birden fazla alt alana bölünmüş olur.
- **Veri doğrulama (Data Validation)** :Verilerde oluşan eksik, yanlış ve tutarsızlıkları önlemek için yapılan işlemlerdir. Veri doğrulama süreci, ETL’in her aşamasında başvurulan bir süreçtir ve bu makalede ayrı bir bölüm olarak da sunulmuştur.
- **Özetleme (Summarization)** :Birden çok sevide hesaplanan ve depolanan verilerin toplam rakamlarının özetlenmesidir. Örneğin Bir müşterinin Müşteri Hayat Boyu Değeri (Customer Lifetime Value) metriğini oluşturmak için yapmış olduğu satın alma işlemlerinin toplamıdır.
- **Toplama (Aggregation**) : Birden fazla kaynaktan verilerin toplanmasıdır.
- **Entegrasyon (Integration) :** Veri bütünleştirme, aynı veri öğesi için farklı veri adlarını ve değerleri eşleştirir

6. L(Load-Yükleme)

Yükleme aşamasında, verileri basit sınırlandırılmış düz dosya veya veri ambarı olabilecek son hedefe yükler. Kurumun gereksinimlerine bağlı olarak, bu süreç çok değişkenlik göstermektedir. Bazı veri ambarları

mevcut bilgilere kümülatif bilgilerin üzerine yazabilir; çıkarılan verilerin güncellenmesi sıklıkla günlük, haftalık veya aylık olarak yapılır. Diğer veri ambarları (veya aynı veri ambarının diğer kısımları), düzenli aralıklarla, örneğin saatlik olarak, tarihi bir biçimde yeni veriler ekleyebilir. Örneğin, geçen yılın satış kayıtlarını tutmak için gerekli olan bir veri ambarı düşünelim. Bu veri ambarında, bir yıldan eski olan tüm verilerin yeni verilerle değiştirilmesi kabul edilebilir bir durumdur. Buna benzer operasyonlarla veri ambarındaki yükün azaltılması ve hedefe yönelik olarak veri ambarının tasarlanması mümkün olmaktadır.

Değiştirilecek veya eklenecek zamanlama ve kapsam, mevcut zamana ve iş ihtiyaçlarına bağlı olarak stratejik seviyede uzmanlar tarafından tasarlanır. Daha karmaşık sistemler, veri ambarına yüklenen verilerin, tüm değişikliklerinin yani veri ambarı üzerindeki bütün süreçlerin geçmişini ve denetim izini sürdürebilir.

Yükleme aşaması bir veri tabanıyla etkileşime girdiğinde, veri şemasında tanımlanan kısıtlamaların yanı sıra veri yüklendiğinde etkinleştirilen tetikleyiciler de (genelde veri kalitesi performansına katkıda bulunan benzersizlik, zorunlu alanlar gibi) ETL sürecinde geçerlidir.

Örneğin, bir finans kuruluşu bir müşteri hakkında çeşitli departmanlarda bilgi sahibi olabilir ve her departman müşterinin bilgilerini farklı bir şekilde listeleyebilir. Üyelik departmanı müşteri adıyla listeleyebilirken, muhasebe departmanı müşteriyi numaraya göre listeleyebilir. ETL, bu veri öğelerinin tümünü paketleyebilir ve onları bir veri tabanında veya veri ambarında saklamak gibi tek bir sunumda birleştirir.

Şirketlerin ETL'i kullanmasının bir başka yolu, bilgiyi başka bir uygulamaya kalıcı olarak taşımaktır. Örneğin, bir şirkette yeni geliştirilmeye başlayan bir uygulama için başka bir veri tabanı tedarikçisi ve muhtemelen çok farklı bir veri tabanı şeması kullanabilir. ETL, veriyi yeni uygulamanın kullanması için uygun bir biçime dönüştürmek ve farklı teknolojilerdeki, amaçlardaki ve yapılardaki verileri birbirine uyumlu hale getirmek için de kullanılabilir. Bu duruma bir örnek, muhasebecilik, müşavirlik ve hukuk firmaları tarafından kullanılan bir Gider ve Maliyet Geri Kazanma Sistemi (ECRS) olacaktır. Bazı işletmeler veriyle olan serüvenini faturalandırma sisteminde sona erdirir, ancak bazı işletmeler ham verileri Fabrika Yönetimine İnsan Kaynakları (personel departmanı) ya da ekipman kullanım raporları için çalışan verimlilik raporları için de kullanabilir. Bütün bu farklı sistemlerin birlikte çalışabilmesi için de ETL kullanışlı bir araçtır.

Bir veri ambarına, veri yüklemek için iki temel yöntem vardır:

Tam Yükleme (Full Load): bir veri kaynağı depoya ilk kez yüklendiğinde gerçekleşen tüm verinin üzerinden geçilmesi ve hedef sistemde verinin yazılacağı tabloların boşaltılarak (veya ilk oluşturmada zaten boş oluşmasının ardından) verinin tamamının bir seferde boş sisteme yüklenmesidir.

Artımlı Yükleme (Incremental Load): Hedef ve kaynak verileri arasındaki fark, düzenli aralıklarla hesaplanır ve son çıkarma (extract) tarihi depolanır, böylece yalnızca bu tarihten sonra eklenen kayıtlar yüklenir. Artımlı yükler, yüklediğiniz verilerin hacmine bağlı olarak değişen iki niteliğe sahiptir:

- Akış artan yük: Daha küçük veri hacimleri için daha uygundur.
- Toplu artımlı yük: Büyük veri hacimlerini yüklemek için daha uygundur.

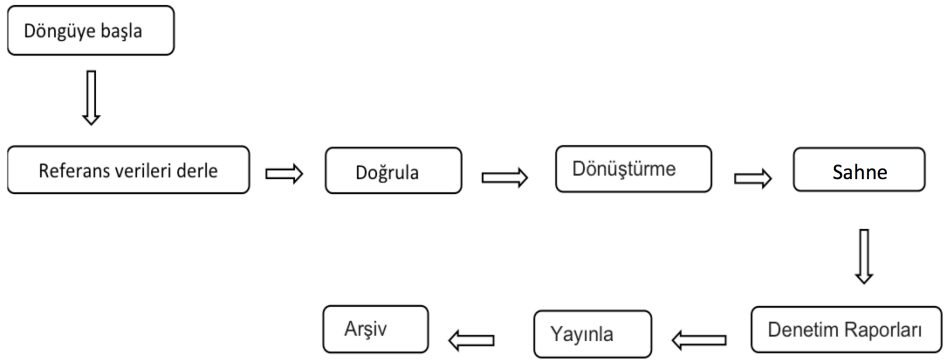
Tablo 1. Tam Yükleme ve Artırımlı Yükleme Karşılaştırması

	Tam Yük	**Artımlı Yük**
Satır uygunluğu	Kaynak verilerdeki tüm satırlar	Yalnızca yeni ve güncellenmiş kayıtlar
Zaman	Daha fazla zaman	Daha az zaman
Zorluk	Düşük	Yüksek. ETL, yeni / güncellenmiş verilerin satırları kontrol edilmeli, bir konudaki verileri kurtarma daha zor

7. ETL Yaşam Döngüsü (ETL Life Cycle)

ETL sürecini bir yaşam döngüsü olarak ele almak da mümkündür. Genel olarak bir ETL sürecinin aşağıda listelenen 10 adımdan oluşması beklenir:

- Döngünün başlangıcı (Initiation of cycle): ETL aşamalarında kullanılacak olan ortamın hazırlanması sürecidir.
- Referans verisinin inşa edilmesi (Building reference data): Veri sözlüklerinin, verinin taşıması gereken özelliklerin, anlambilimsel (semantic) bağlantıların kurulduğu aşamadır.
- Farklı kaynaklardan verinin çıkarılması (Extracting data from different sources): Verinin çekileceği kaynaklar belirlenir ve veri çekilmesi için gerekli protokol ve kod geliştirme süreci koşulur.
- Verinin doğrulanması (Validation of data): farklı kaynaklardan gelen verilerin daha önceki referans verisine göre doğrulanması aşamasıdır.
- Verinin Dönüştürülmesi (Transforming data): Veri işlemeye ve çalışmaya uygun şekle dönüştürülür.
- Sahne (Staging): veri ambarı projelerinin koşan canlı veriyi içeren ve uygulama katmanına veri besleyen aşamasıdır.
- Teftiş raporlarının üretilmesi (Generation of audit reports): çalışan ve yüklenmiş veri üzerinde kontroller yapılarak denetim raporları hazırlanır.
- Verinin yayınlanması (Publishing data): Veri uygulama katmanı ve diğer sistemleri besler.
- Arşivleme (Archiving): Veri ile ilgili detaylar, erişim raporları, karşılaşılan problemler bu aşamada arşivlenir.
- Temizleme (Cleanup): Bu aşamaya kadar olan bütün



Şekil 4 ETL Yaşam Döngüsü

ETL, büyük verilerde ve kaynak ile hedef veri tabanlarının aynı olduğu yerde daha verimli çalışır.

8. Veri Doğrulama (Data Validation)

ETL sürecinin her aşamasında bulunan ve her aşamasının sağlamasını yapan yan süreçtir. Örneğin veri çıkarım (extract) aşamasının hemen ardında verinin doğru çıkarıldığını, ver dönüşüm aşamasının hemen ardından dönüşümün istenen şekilde yapıldığını veya veri yükleme işleminin hemen ardından veri yüklemenin doğru ve sağlıklı şekilde yapıldığını doğrulamak için kullanılabilir. Bu açıdan veri temizleme (cleaning) aşaması ile birlikte düşünülebilir ancak her zaman veri temizleme ile bitmek zorunda değildir, örneğin bazı durumlarda verideki hataların bulunması ve tutuşmazlıkların anlaşılması bile yeterli olabilmektedir.

Veri doğrulamanın temel olarak 3 amacı bulunur: Uygunluk, Doğruluk ve Tutarlılık.

Genellikle " doğrulama kuralları " "doğrulama kısıtlamaları" veya "denetim rutinleri" olarak adlandırılan yordamlar kullanılır ve bu açıdan ele alındığında, veri temizleme yaklaşımına benzer şekilde kural tabanlı bir sistem olarak düşünülebilir.

Veri doğrulamasının temellerini değerlendirirken gerçekleştirilecek çeşitli doğrulama işlemlerinin kapsamı, karmaşıklığı ve amacı doğrultusunda farklı doğrulama türleri ile ilgili genellemeler yapılabilir. Örneğin sık kullanıldığı alanlar aşağıdaki şekilde sıralanabilir:

- **Veri türü doğrulama (Data Type Validation):** Verinin nümerik veya nominal olması bir metin veya etiket içermesi gibi farklı durumların ele alınarak istenen türde olup olmadığı kontrol edilir. Örneğin yaş hanesine yazıyla "kırkbeş" yazılması veya sayıyla 45 yazılması veya sayının dizgi (string) tipinde yazılması ("45" şeklinde) farklı değerlerdir ve bu türlerin istenen yapıda olması sağlanır.
- **Menzil ve sınır doğrulama (Range and Limit Validation):**Verinin tanımlı bir aralık ve sınırda olup olmadığının doğrulanmasıdır. Örneğin telefon alanının 11 hane ile limitli olması veya yaşın 0 ve 150 arasında tanımlı (menzil) olması bu tip doğrulamalara girer. Örneğin düzenli ifadeler (regular expressions) kullanılarak yapılan veri doğrulamaları bu grupta görülebilir.
- **Kod ve Çapraz Referans doğrulaması (Code and Cross Reference Validation):** Genellikle ETL süreçlerinde kod yazılması hoş karşılanmaz ve gerek standartların uygulanması gerekse de hataların minimuma indirilmesi için ETL süreçlerinin kod yerine araçlar üzerinden sürdürülmesi istenir. Doğrulama (validation) adımları için de geçerli olan bu durum, bazı istisna hallerde ihlal edilebilir. Örneğin iş süreçlerinin ETL süreçleri üzerinde test edilmesi ve verinin doğru aşamalardan doğru şekilde geçtiğinin test edilebilmesi veya farklı kaynaklardan gelen verilerin bazı işlemlerden geçirilerek birbirini doğrulaması gibi kontroller yapılabilir. Örneğin bütün kontrolleri geçen bir e-posta adresi sisteme girilmiş olabilir ancak bu e-postanın aktif olup olmadığı ve hatta çalışan bir adresten olup olmadığının kontrolü için bir kod parçası yazılarak çalıştırılabilir.
- **Veri profillemesi (Data Profiling),** var olan bir bilgi kaynağından (örneğin, bir veri tabanı veya bir dosya) mevcut verileri incelemek ve bu verilerle ilgili istatistikler veya bilgilendirici özetler toplamak işlemidir. Veri Profillemesi yalnızca veri kalitesini değerlendirmeye değil, aynı zamanda

kurumsal büyük verileri keşfetmeye, kaydettirmeye ve değerlendirmeye de yardımcı olur. Analiz sonucu aday kaynak sistemlerinin uygunluğunu belirlemek ve daha sonraki çözüm tasarımı için problemleri belirlemek için kullanılır.

- Veri profillemenin faydaları veri kalitesini artırmak, büyük projelerin uygulama döngüsünü kısaltmak ve kullanıcıların veri anlayışını geliştirmektir. Verilerin gömülü olduğu işletme bilgilerini keşfetmek, veri profillemesinden elde edilen önemli faydalardan biridir. Veri profillemesi, kurumsal veri tabanlarında veri doğruluğunu artırmak için en etkili teknolojilerden biridir.
- **Yapısal Doğrulama (Structured Validation):** Yukarıdaki doğrulama adımlarının birleştirildiği ve birden fazla doğrulama yönteminin nasıl bir yapı içerisinde birlikte çalışacağını sıralayan doğrulama sistemleri olarak düşünülebilir.

Veri doğrulama sırasında veri üzerinde farklı yöntemler kullanılarak veri hakkında kullanışlı bazı sonuçlara ulaşılabilir, bu sonuçlar daha sonra veri doğrulama testleri için kullanılır. Literatürde sık kullanılan bazı veri doğrulama testleri aşağıda sıralanmıştır:

Veri Bütünlük Testi (Data Completeness Test): Bu testin amacı, kaynakta bulunan bütün verilerin hedef sisteme yüklendiğini test etmektir. Bu amaçla satır sayısını almak veya birleştirici bazı fonksiyonları veri üzerinde çalıştırmak (minimum, maksimum, ortalama veya toplam gibi) ve çıkan sonuçları karşılaştırarak veri yükleme işleminin ne kadar sağlıklı gerçekleştiğini anlamak sıkça kullanılan yöntemlerdir.

Veri Kalite Testi (Data Quality Test): Verinin ne kadar kaliteli (sağlıklı veya kesin) olarak yüklendiğini ölçmeye yarayan testtir. Veri profillemesi gibi yöntemler kullanılarak veri üzerindeki problemler tespit edilir ve çözüm yöntemleri geliştirilir. Veri kalite testlerinin genel bir sorunu, veri kaynaklarının sürekli olarak değişime açık olması ve otomatik olarak veri kalite testlerinin çalıştırıldığı ortamlarda, veri kalitesini etkileyen yeni faktörlerin sisteme dahil olmasıdır. Bu problemin çözümü için makine öğrenmesi yöntemleri kullanılabilir.

Üst Veri Testi (Metadata Test): Veri tabanlarının ve veri kaynaklarının taşımış olduğu üst verileri kullanarak yapılan testlerdir. Örneğin veri tabanındaki bir tablonun üst verisini kontrol ederek ilgili verinin tipinin sayı veya dizgi (string) olup olmadığı ve alınan verinin bu yapıya uygunluğu test edilebilir.

Veri Dönüşüm Testi (Data Transformation Test): Genellikle, veri, ETL süreçleri ile dönüştürülerek bir sisteme yüklendikten sonra, verinin uygulama katmanında kullanılması hedeflenmektedir. Bu açıdan uygulama katmanındaki verinin kullanılabilirliği ve dönüşümün başarılı olup olmadığını anlamak için literatürde sık kullanılan iki test bulunur. Kara Kutu testi (Black-box testing) ve beyaz kutu testi (White-box testing).

Kara kutu testi (Black-box testing): basitçe bir sistemin iç yapısına karışmadan, sistemin girdileri ve çıktıları arasındaki ilişkiye bakar [8]. Veri doğrulama aşamasındaki kara kutu testleri ise temel olarak aşağıdaki fonksiyonları icra eder:

- İster analizinde bulunan dönüşüm kurallarını çıkarır
- Farklı dönüşüm (transformation) senaryoları için örnek veri kümeleri üretir
- Önceki adımda elde edilen sentetik veri ile sistemi besler ve dönüşüm sonuçlarını toplar.
- Beklenen dönüşüm sonucunu elde edilen değerlerle karşılaştırır (senaryoları çalıştırır).

Kara kutu yaklaşımının en büyük avantajı, testlerin sistemin iç yapısına müdahale etmeden çalıştırılabiliyor olmasıdır. Yani test süresince herhangi bir şekilde yeniden kodlamaya ve sistem tasarımına ihtiyaç duyulmaz. Bununla birlikte en büyük dezavantajı, test sırasında testçinin her senaryo için veri üretmesi veya otomatik olarak üretilen verilerin sistem için uygun olup olmadığını test etmesi gerekmesidir.

Beyaz Kutu testi (White-box testing): Kara kutu testinin aksine, sistemin iç yapısının incelendiği ve sistemde bulunan kodlar, programlama mantığı veya iş akışı gibi detaylara göre, özel olarak test verilerinin oluşturulduğu test tekniğidir. Dönüşüm (transformation) testleri için, genelde haritalama tasarım dökümanı (mapping design document) üzerinden yürütülen ve ETL kodları üzerinden çalıştırılarak üretilen test vakaları oluşturulur. Beyaz kutu testinin temel adımları aşağıdaki gibidir:

- Dönüşüm tasarımının anlaşılması için kaynaktan hedefe kara olan veri haritalaması (data mapping) çalışılır.
- ETL dönüşüm mantığının yansıtılması için benzer kodlar, prosedürler, SQL cümleleri üretilir
- Üretilen test prosedürleri veya kodları üzerinde örnek veriler çalıştırılarak sistemin örnek verilere verdiği dönüşüm sonuçları ile karşılaştırılır.

ETL İlkelleme Testi (ETL Regression Test): ETL sürecinde her değişimden önce verilen değerin sonra verilen aynı değer ile aynı olduğunu kontrol etmeye yarar. Herhangi bir değişim için farklı sonuçlar elde edilmesi halinde sonuçların doğrulanması yapılır.

Referans Veri Testi (Reference Data Testing): Çoğu veri tabanı alanında kısıtlı değerler arasından seçim yapılması beklenir. Örneğin Türkiye'deki adres alanları için şehir seçenekleri sınırlı sayıdaki alternatiften seçilerek oluşturulur. Referans olarak alınan bu listeler ile veri karşılaştırılarak sonuçların belirlenen referanslar arasında olup olmadığı kontrol edilir.

Artırımlı ETL Testi (Incremental ETL Testing): ETL süreci genelde tam veya artırımlı olarak iki farklı şekilde çalıştırılır. Tam çalışma şeklinde, veri tabanında bulunan bütün tabloların içi boşaltılarak verinin tamamının yeniden sisteme yüklenmesi beklenir. Artırımlı şekilde ise, sadece değişen ve yeni verilerin sisteme yüklenmesi beklenir. Artırımlı ETL sürecinin amacı, çalışma süresini ve sistem üzerindeki veri yükünü azaltmaktır. Artırımlı şekildeki ETL süreçlerini test etmek için de benzer amaçla artırımlı şeklide çalışan ETL testleri, sisteme yapılan ekleme (insert) ve güncelleme (update) sorgularını test ederler.

ETL Birleştirme Testi (ETL Integration Test): ETL süreçlerinin tamamlanması ve verinin bir veri ambarına yüklenmesinin ardından, veri üzerinde bir uygulama çalışır. Genelde iş zekası (business intelligence, BI) veya veri göç projesi (data migration) veya eski sistemlerle (legacy systems) yeni sistemler arasında entegrasyon gibi amaçlarla hazırlanan bu uygulamaların çoğunda veri entegrasyonu yapılması ve birden fazla kaynaktan veya uygulamadan gelen verilerin birlikte kullanılmasını gerektirir. ETL birleştirme testlerinin amacı ise, uçtan uca, birden fazla kaynaktan gelen ve birden fazla uygulama için kullanılan verilerin doğru şekilde birleştirildiğinden emin olmaktır.

ETL Performans Testi (ETL Performance Test): ETL projeleri için en önemli kriterlerden birisi de performanstır. Genelde geliştirme ortamlarında yeterli veri işleme ve performansı test etme imkanı bulunmaz. Geliştirme ortamları daha çok veri üzerinde tanımlı olan çıkarım, dönüşüm ve yükleme adımlarını gerçekleştirmek ve bu adımlardaki problemlere çözüm bulmak için kullanılırlar. Bununla birlikte canlı bir sisteme çıkıldığında, daha önce karşılaşılmamış boyutlardaki ve farklı kaynaklardan akan verinin işlemesi, beklenmedik performans problemlerini de beraberinde getirebilir. ETL sürecinin canlıya geçilmeden her adımının performans testlerinin yapılması mümkündür. Ayrıca sistemin uçtan uca çalışma süreleri ve sistem kaynaklarına getireceği yük gibi farklı metriklerinin çalışmadan önce kestirilmesi ve iyileştirilmesi de mümkündür.

9. ETL ve Bazı Sık Yaşanan Sorunlar ve diğer ETL Konuları

Bir ETL projesinde yanlış olan en yaygın hatalar ve problemler:

- Şirketlerin unuttuğu bazı bakım maliyetleri

- Zamanla değişen veri formatları
- Veri hızında bir artış
- Yeni veri bağlantıları ekleme zaman maliyeti
- Kırılmış veri bağlantılarını düzeltmenin zaman maliyeti
- Yeni sütunlar, boyutlar ve türevler de dahil olmak üzere yeni özellikler için talepler

Farklı kaynaktan gelen ham veriler, mantıklı olabilmesi için veri ambarına gelmeden hemen hemen her zaman temizlenmeli ve normalleştirilmelidir. Farklı sistemlerden gelen verilerde dönüştürme adımı göz ardı edilirse veri tutarlığı bozulur ve karar vericilerin doğruluğunu kaybetmesine neden olur.

İş birimlerinin de bu projenin müşterisi olduğu unutulmamalıdır. Yeterli seviyede analizler yapılmazsa, kullanıcıların en çok ihtiyaç duyduğu bilgileri sunmama, görev kritik raporlama iş akışını desteklemede başarısız olma, gelecekteki veri ihtiyaçlarının tahmin edilmesinde zorlukların yaşanmasına neden olur.

Verilerin büyüklüğü ve çeşitliliği zamanla hızlı bir şekilde artacaktır. Bunun için ETL sisteminizin altyapısı ayarlanabilir olması gerekir. Aksi halde ileride veri büyüklüğü arttığında zorluklar yaşanabilir.

Veri mühendisliği alanı şu anda inanılmaz derecede hareketli. Her dakika yeni diller ve teknolojiler ortaya çıkıyor. Amaç herkesin kullandığı teknolojiyi kullanmak değil problem için en uygun olanı kullanmak. Bu yüzden ETL araçları ikinci planda olmalıdır.

9.1. ETL'ı Kimler Kullanır?

- Yazılımcılar
- İş zekası raportörleri
- Veritabanı yöneticileri vs. kişiler kullanır. [9]

9.2. Nerede Kullanılır?

- Entegre çalışan sistemler
- Veri ambarı
- Veri temizleme
- Veri toplama
- Raporlama vs. alanlarda kullanılır.[10]

10. Bazı Örnek Projeler

Beş yaşın altındaki çocukların kazayla ölümlerinin analiz edilmesi: Bu çalışma Van ve çevresinde meydana gelen kazaların 5 yaş ve altındaki çocukların otopsi sonuçlarına göre ölüm olgularının profilinin ortaya konulması ve literatürdeki bilgileri ile karşılaştırılması amaçlanmıştır.

Van ilinde 01.01.2010 ve 31.12.2014 tarihleri arasındaki meydana gelen kazaların 5 yaş ve altındaki kişilerin verilerine göre değerlendirilmiştir.

Sonuç olarak Değerlendirilen 151 olgunun 85'i (%56,3) erkek, 66'sı (%43,7) kız çocukları olarak tespit edilmiş olup, kazalara bağlı ölümler, ölüm nedenlerine göre sınıflandırıldığında, suda boğulmaların ilk sırada yer aldığı (%28,5; n=43), bunu %27,8'lik bir oran ile (n=42) trafik kazalarının ve %17,2'lik bir oran ile (n=26) yabancı cisim ve gıda aspirasyonları ve bir cisim altında kalmaya bağlı karın-göğüs basılarını içeren diğer asfiksi türlerinin izlediği tespit edilmiştir.[11]

Diyabet hastalarına ait verilerin veri madenciliği yöntemleri ile analiz edilmesi: Bu proje kapsamında Erciyes Üniversitesi, Tıp Fakültesi Hastaneleri'nde diyabetik hasta veritabanındaki hastalara ait muayene ve laboratuvar test verilerinin veri madenciliği yöntemleri ile analiz edilerek diyabet hastalığının teşhis ve tedavisinde kullanılabilecek yöntemlerin araştırılarak geliştirilmesi amaçlanmıştır.

Aibinu ve arkadaşları tarafından yapılan çalışmalarda bireylerin diyabet olup olmamasının sınıflandırmasında karmaşık değerli otoregresif (complex-valued autoregressive) yöntemi ve karmaşık-değerli sinir ağları (complex-valued neural network) ve gerçek-değerli sinir ağları (real-valued neural network) yöntemlerine dayalı parametrik yaklaşımlar (otoregresyon yaklaşımları) önerilmiştir. Çalışmalarda Pima Yerlileri Diyabet Veritabanı kullanılmıştır.[12]

Hastane bilgi yönetim sistemlerinde OLAP yöntemleriyle karar destek modülü geliştirmek: Bu projede tanı ve tedaviye yönelik tıbbi araştırmalara cevap vermek amacıyla niteliksiz veri bombardımanı altında sağlık sektörü yöneticilerine, zaman dilimli ve çok boyutlu veri görünümü sağlayan OLAP analiz işlemi ışığında yol gösterilmeye çalışılmıştır.

Çalışmada, Microsoft SQL Server 2000'in Database Server, SQL Agent, Data Transformation Services ve Analysis Services bileşenleri kullanılmıştır. OLAPPROG uygulaması ise Borland Delphi 7.0 yazılım geliştirme ortamında kodlanmıştır [13].

11. ELT (Extract, Load, Transform) (Verinin Ayrıştırılması, Yüklenmesi ve Dönüştürülmesi)

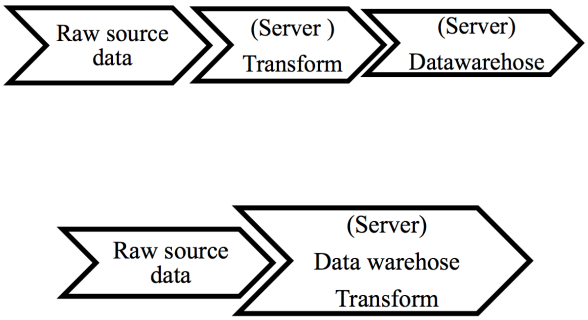
Ayıkla, Yükle, Dönüştür (ELT), ham bir veriyi bir kaynak sunucudan bir hedef sunucudaki bir veri ambarına aktarmak ve ardından bilgi akış yönündeki kullanımlar için hazırlamak için bir veri bütünleştirme işlemidir. [14]

ELT, ham verilerin doğrudan hedefe yüklenmesine ve orada taşınmasına izin verir.

Bu yetenek, iş zekası (BI) ve büyük veri analizi için gerekli olan geniş veri kümelerini işlemek için daha kullanışlıdır. İlgi çekici özeliklerden biri olan ELT ,ETL'e göre yüklenme süresini azaltır.

ETL den en büyük eksikleri arasında metadata ve CDC(Change data capture(değişen verileri, değişen verilerin yerlerini gibi))yönetimi yok. Ayrıca ELT de fazla tool olmadığından gerekli kıvraklığı her zaman sağlayamamaktadır. Aslında ikisininde ayrı ayrı performansı yükseltebilecek kullanım alanları vardır

Kuruluşlar, verileri bir araya getirmek, veri ambarı, raporlama ve analitik için genellikle gerekli olan denetim sağlamak için hem ETL hem de ELT'ye ihtiyaç duyabilirler.



Şekil 5 ETL ve ELT Yaklaşımları

Şekil 5'te gösterilen ilk yaklaşımda ETL süreci ve ikinci yaklaşımda ise ETL süreci sunucu ihtiyaçları ile verilmiştir. Buna göre ilk yaklaşımda, iki farklı sunucuya ihtiyaç duyulurken, ikinci yaklaşımda tek sunucu yeterli olabilmektedir. ETL' de T(Yani Transformation) için ayrı bir Transformation sunucu kullanılması talep edilirken, ELT' de transformation çoğunlukla ayrı bir sunucuda değil de hedef veya kaynak veri sunucularında yapılması tercih edilir. Hatta bu transformation yükü de kaynak ve hedef arasında paylaşılır.[15]

Kaynakça

[1] Etl Vs Elt: Datawarehousing, " http://blogs.perficient.com/delivery/blog/2016/07/14/elt-vs-etl-data-warehousing/2017"

[2]Suresh, Sankaran, et al. "Method and architecture for automated optimization of ETL throughput in data warehousing applications." U.S. Patent No. 6,208,990. 27 Mar. 2001.

[3] Stonebraker, Michael, et al. "MapReduce and parallel DBMSs: friends or foes?." Communications of the ACM 53.1 (2010): 64-71.

[4] Seker, Sadi Evren, and Enes Eryarsoy. "Generating Digital Reputation Index: A Case Study." Procedia-Social and Behavioral Sciences 195 (2015): 1074-1080.

[5] Al-Khateeb, Tahseen, et al. "Recurring and novel class detection using class-based ensemble for evolving data stream." IEEE Transactions on Knowledge and Data Engineering 28.10 (2016): 2752-2764.

[6] Şeker, Şadi Evren. "Temporal logic extension for self-referring, nonexistence, multiple recurrence, and anterior past events." Turkish Journal of Electrical Engineering & Computer Sciences 23.1 (2015): 212-230.

[7] Al Naami, Khaled Mohammed, Sadi Seker, and Latifur Khan. "Gisqf: An efficient spatial query processing system." Cloud Computing (CLOUD), 2014 IEEE 7th International Conference on. IEEE, 2014.

[8] Sadi Evren Seker, Aktör Ağ Teorisi (Actor Network Theory), YBS Ansiklopedi, c. 1, s. 1, syf. 17-19

[9] Sadi Evren Seker, Eda Esmekaya, Eksik Verilerin Tamamlanması (Imputation), YBS Ansiklopedi, c. 4, s. 3, syf. 10 - 17

[10] Rodic, Jasna, and Mirta Baranovic. "Generating data quality rules and integration into ETL process." Proceedings of the ACM twelfth international workshop on Data warehousing and OLAP. ACM, 2009.

[11] Demir, Uğur, et al. "Analysis of accidental deaths of children under five years of age."

[12] Aibinu, A.M., Salami, M.J.E., Shafie, A.A. 2010. "Application of Modeling Techniques to Diabetes Diagnosis", IEEE EMBS Conference on Biomedical Engineering & Sciences (IECBES 2010). Koala Lumpur, Malaysia:

[13] YALDIR, Abdülkadir, And Murat TAŞER. "HASTANE BİLGİ YÖNETİM SİSTEMLERİ İÇİN OLAP YÖNTEMLERİ İLE KARAR DESTEK MODÜLÜ TASARIMI VE UYGULAMASI." İktisadi Ve İdari Bilimler Fakültesi Dergisi 18.1 (2016): 153-171.

[14] Howatt, Anthony Philip Reid, and Henry George Widdowson. A history of ELT. Oxford University Press, 2004.

[15] ETL vs. ELT – What's the Big Difference?,"https://www.ironsidegroup.com/2015/03/01/etl-vs-elt-whats-the-big-difference/"