

BEYZA NUR KAMAŞ ABAP BOOTCAMP ÖDEV

BAPI ve BATCH INPUT

BAPI'ler, SAP sistemlerindeki işlevleri çağırmak ve SAP verileriyle etkileşimde bulunmak için kullanılan standart fonksiyonlardır. Bu fonksiyonlar, SAP sistemlerindeki verilere erişmek, veri güncellemeleri yapmak veya iş süreçlerini yönetmek için kullanılır. BAPI'ler aslında bir tür "kod parçaları seti" olarak düşünülebilir. Bu "kod parçaları seti", önceden tanımlanmış işlemleri gerçekleştirmek için kullanılır ve başka yazılımlarla veya sistemlerle iletişim kurabilir.

BAPI ve fonksiyonlar aracılığı ile sipariş oluşturma/değiştirme, sipariş simülasyon, teslimat oluşturma/değiştirme, nakliye oluşturma/değiştirme, mal çıkışı kaydetme, taşıma birimi oluşturma/bozma, WM'li sevkiyat depoda çekme, ölçü dönüşümü hesaplama, fatura oluşturma, ters kayıt işlemleri, fiyat koşulu atma, muhatap oluşturma/değiştirme, müşteri FI/SD verilerini oluşturma, ilgili kişi tayin etme, kur hesaplama, tarih hesaplama, SD belge detaylarını çekme vb. daha bir çok işlem BAPI/fonksiyonlar kullanılarak gerçekleştirilebilir. Sipariş türünde belge oluşturmak için SD_SALESDOCUMENT_CREATE, BAPI_SALESORDER_CREATEFROMDAT2 bapileri kullanılabilir.

Örneğin, bir şirketin satış verilerini yönettiği bir sistem var diyelim. Bu sistemde, yeni bir satış siparişi oluşturmak için bir BAPI bulunabilir. Bu BAPI, siparişin detaylarını alır (müşteri bilgileri, ürünler, miktarlar) ve bu bilgileri kullanarak yeni bir sipariş oluşturur.

Bir örnek vermek gerekirse: E-ticaret siteniz var ve müşteriler internet üzerinden ürünler satın alıyor. Bu alınan siparişleri firmanızın içinde kullandığı başka bir sistemde tutmak istiyorsunuz. İşte burada BAPI devreye girer. E-ticaret siteniz, müşterinin verdiği siparişi, kullanacağı bir BAPI aracılığıyla doğrudan firmanızın iç sistemine aktarabilir.

Diyelim ki, bir müşteri web sitenizden bir ürün sipariş etti. Sipariş tamamlandığında, e-ticaret siteniz bu siparişi firmanızın içindeki SAP sistemiyle bağlantı kurarak kaydetmek istiyor. İşte bu noktada e-ticaret siteniz, BAPI'yi kullanarak SAP sisteminde yeni bir sipariş oluşturabilir. Bu BAPI, sipariş bilgilerini alır ve firmanızın içindeki sistemde bir satış siparişi olarak kaydeder. Yani BAPI'ler, farklı yazılımlar arasında iletişim kurmak ve belirli işlemleri gerçekleştirmek için kullanılır. Önceden tanımlanmış görevleri yapmak üzere tasarlanmışlardır ve genellikle diğer

yazılımlar tarafından kullanılabilir. Özetle, BAPI'ler, yazılımlar arasında bilgi alışverişi ve işlem yapma süreçlerini kolaylaştıran bir araçtır.

Peki nasıl kullanabiliriz? TCode olarak BAPI'yi girebilir veya SAP Menüsü'nde bulabiliriz. Alternatif olarak, SAP Easy Access ekranında /nBAPI yazarak doğrudan BAPI TCode'una geçebiliriz.

BAPI Explorer: BAPI Transaction Code'u, BAPI Explorer benzeri bir arayüz sağlar. Bu arayüzde, sistemdeki mevcut BAPI'leri listeleyebilir ve detaylarını görebiliriz. Burada, BAPI'lerin isimlerini, açıklamalarını, giriş ve çıkış parametrelerini görebiliriz.

BAPI Seçimi: İhtiyacınıza uygun bir BAPI seçeriz. Bunu yaparken, işlevselliği, gerekli giriş parametrelerini ve çıktığı dikkate almalıyız.

BAPI Kullanımı: Seçtiğimiz BAPI'nin detaylarını inceledikten sonra, bu BAPI'yi kullanmak için uygun giriş parametrelerini belirlemeliyiz. Giriş parametrelerini doldurup BAPI'yi çağırarak işlevi gerçekleştirebiliriz. BAPI'nin kullanımı, seçilen BAPI'nin işlevine ve gereksinimlerinize bağlı olarak değişebilir. Bu adımlar genel bir rehber sağlar, ancak her BAPI farklı olabilir. İhtiyaç doğrultusunda seçtiğimiz BAPI'nin belgelendirmesini incelemek önemlidir. Bu bize gerekli giriş parametreleri ve işlevin nasıl kullanılacağı hakkında detaylı bilgi verecektir.

Batch input bir veri giriş yöntemidir ve genellikle SAP sistemine büyük miktardaki verileri otomatik olarak girmek için kullanılan bir yöntemdir. Örneğin, yüzlerce veya binlerce kaydı SAP sistemine elle girmek yerine, bu verileri otomatik olarak yüklemek için kullanılır . Bu yöntemde, kullanıcı bir işlemi gerçekleştirirken yapılan adımlar kaydedilir ve daha sonra tekrar o adımlar otomatik olarak gerçekleştirilebilir.

Örneğin, bir ekran üzerinde belirli alanlara veri girişi yapılırken yapılan tıklamalar, veri girişleri ve işlem kodları kaydedilebilir. Bu kaydedilen adımlar daha sonra tekrar çalıştırılarak aynı işlemler otomatik olarak yapılabilir. Böylece, kullanıcıların tekrarlayan işlemleri elle yapmak yerine, bu adımları otomatikleştirilmiş bir şekilde gerçekleştirmelerine olanak tanır. Bu süreci somut bir örnekle anlatmak gerekirse, öncelikle bir programla belirli bir ekranın veri giriş adımları kaydedilir. Ardından, bu kaydedilen adımlar otomatik olarak tekrarlanabilir, böylece kullanıcı aynı adımları elle yapmak zorunda kalmaz.

Örnek bir senaryo şu şekilde olabilir: Bir şirketin personel bilgilerini girdiği bir ekranı ele alalım. Kullanıcılar, her yeni personel için belirli bir ekranı açıp ad, soyad, pozisyon gibi bilgileri elle girmek zorundadır. Batch input kullanılarak, bu ekranın adımları kaydedilebilir ve yeni bir personel eklendiğinde aynı adımlar otomatik olarak çalıştırılabilir. Böylece, işlem tekrarı büyük ölçüde azalır ve veri girişi hızlanır. Bu yöntem, veri yoğun iş süreçlerinde ve tekrarlayan görevlerde kullanılarak verimliliği artırabilir. Ancak, dikkatli olunması gereken bir nokta da, bu otomatikleştirilmiş adımların hata olasılığını da beraberinde getirebilmesidir. Bu nedenle, batch input kullanırken doğru ve güvenilir veri girişi sağlanması önemlidir.

BAPI ve Batch Input arasındaki farklar nelerdir ?

Diyelim ki, müşteri bilgilerini SAP sistemine eklememiz gerekiyor. Bunu BAPI kullanarak şu şekilde yapabiliriz: BAPI'yi çağırarak müşteri ekleme işlemi başlatırız. BAPI'ye gerekli müşteri bilgilerini parametre olarak geçiririz (örneğin, müşteri adı, adresi, vb.). BAPI, müşteriye ekler ve işlem sonucunu bize döner.

Batch Input Örneği: Aynı işlemi Batch Input kullanarak gerçekleştirelim: Bir Excel dosyasında müşteri bilgilerini hazırlarız (örneğin, müşteri adı, adresi, vb. sütunlarla). SAP sistemi tarafından kabul edilen belirli bir formata dönüştürmek için bu Excel dosyasını kullanarak bir giriş dosyası oluştururuz. SAP sistemi üzerinde transaksyon kodları kullanarak manuel olarak bu giriş dosyasını yükleriz. Yüklenen veri SAP sistemi tarafından işlenir ve müşteriler eklenir.

Farklar: BAPI, daha programatik bir yaklaşımdır ve işlevleri doğrudan çağırmak için kullanılır. Batch Input, daha çok manuel işlemleri otomatize etmek amacıyla kullanılır ve genellikle veri yükleme için tercih edilir. BAPI, iş mantığını daha iyi korur ve hata durumlarına daha iyi tepki verebilir. Batch Input, genellikle daha düşük seviyeli bir yaklaşım sunar ve işlemleri transaksyon kodları ile gerçekleştirir. Bu örnekler, BAPI'nin programatik bir arayüz sağladığı ve Batch Input'un daha çok veri yükleme işlemleri için kullanıldığı konseptini göstermektedir.

VIEW ile TABLE ARASINDAKİ FARK NEDİR?

Tablo ve view, SAP ABAP programlamada kullanılan iki temel veritabanı nesnesidir. Tablo, veritabanında yapısal bir düzen içinde veri saklamak için kullanılan bir nesnedir. Alanlar (fields) adı verilen sütunlardan oluşur ve belirli bir iş süreci veya modülle ilişkilendirilmiş verileri depolar.

View, bir veya daha fazla tablodan gelen verileri içeren bir mantıksal tablo görünümüdür. Veritabanındaki tablolardan veri okuma ve işleme işlemlerini kolaylaştırarak özelleştirilmiş veri görüntüleri sağlar. Genellikle sadece okuma amaçlı kullanılır ve veriyi fiziksel olarak saklamaz. Her ikisi de veri yönetimi ve iş süreçleri açısından önemli roller üstlenir, ancak aralarında belirli farklar bulunmaktadır.

Table

Fiziksel Depolama: Tablo, veritabanında fiziksel olarak depolanan bir nesnedir. Gerçek veriler burada saklanır ve tablonun alanları belirli bir veri türünü temsil eder.

Veri Saklama ve Güncelleme: Temel amacı veri saklamak olan bir tablo, genellikle belirli bir iş süreci veya modülle ilişkilendirilmiş verileri düzenli bir şekilde depolar. Ayrıca, tablolara veri eklemek, güncellemek ve silmek mümkündür.

İlişkisel Yapılar: İlişkisel veritabanı yönetim sistemleri (RDBMS) kullanılarak tablolar arasında ilişkiler kurulabilir. Bu, veri bütünlüğünü sağlamak ve veriler arasında bağlantıları desteklemek için önemlidir.

Veri Tabanlı İşlemler: İş süreçlerinde doğrudan veri manipülasyonu için kullanılır. Örneğin, bir müşteri siparişi oluşturmak veya stok seviyelerini güncellemek gibi işlemler tablolara doğrudan etki eder.

View

Mantıksal Görünüm: Bir view, bir veya daha fazla tablodan veya başka bir view'dan gelen verileri içeren bir mantıksal tablo görünümüdür. Fiziksel bir depolama birimi değildir, ancak veriye yönelik bir sanal görünüm sunar.

Okuma ve İşleme Kolaylığı: Veritabanındaki tablolardan veri okuma ve işleme işlemlerini kolaylaştırmak için kullanılır. Birden çok tabloyu birleştirme veya belirli bir koşula göre filtreleme yeteneği vardır.

Özelleştirilmiş Görünümler: Belirli bir iş ihtiyacına yönelik özel veri görünümleri sağlar. Gereksiz veri karmaşasını önler ve sadece belirli verilere erişim sağlar. Bu, genellikle raporlama veya analiz amacıyla kullanılır.

Okuma Amaçlı Kullanım: Genellikle sadece okuma amaçlı kullanılır. View üzerinde yapılan değişiklikler genellikle tablolara doğrudan yansımaz, çünkü asıl veri tablolarda saklanır.

Farklar

Amaç: Tablo, veriyi saklamak ve güncellemek için kullanılır. View, veriyi okumak ve özelleştirilmiş görünümler sağlamak için kullanılır.

Yapı: Tablo, fiziksel bir depolama birimidir. View, mantıksal bir görünüm sunan sanal bir yapıdır.

İlişkiler: Tablolar arasında doğrudan ilişki kurulabilir. View, genellikle tablolardan gelen verilerin birleştirilmesi veya filtrelenmesiyle oluştuğu için daha esnek bir yapıya sahiptir.

Değiştirilebilirlik: Tablolara veri eklenebilir, güncellenebilir veya silinebilir. View'lar genellikle sadece okuma amaçlı kullanılır ve temelde değiştirilemez.

Bu detaylar, tablo ve view kavramlarının farklı roller ve kullanım senaryolarına sahip olduğunu göstermektedir. Tablolar genellikle veriyi depolamak ve işlemek için kullanılırken, view'lar okuma ve özelleştirilmiş veri görünümleri oluşturmak için kullanılır.

HASH TABLE

"Hash Tablo", bir anahtar-değer çiftlerini hızlı bir şekilde depolamak, ekleme, çıkarma ve erişim sağlamak için kullanılan bir veri yapısıdır. Bu veri yapısı, bir anahtarın bir değere eşlendiği bir dizi gibi düşünülebilir. Hash tabloları genellikle hızlı erişim sağlama yetenekleriyle bilinirler. Bir anahtar, bir özet (hash) fonksiyonu kullanılarak belirli bir değere dönüştürülür ve bu özet, anahtarın depolanacağı konumu belirler. Bu sayede, veriye hızlı bir şekilde erişim sağlanabilir.

Hash tabloları, özellikle büyük veri kümelerinde arama, ekleme ve silme işlemlerini hızlandırmak için kullanılır. Ancak, çakışma dediğimiz durumlar olabilir; yani, farklı anahtarlar aynı özete denk gelebilir. Bu durumu çözmek için çeşitli teknikler kullanılabilir, örneğin, çakışma durumlarını çözmek için zincirleme veya açık adresleme gibi yöntemler mevcuttur. SAP ABAP'da hash tabloları genellikle DATA: HASHED TABLE anahtar kelimesi ile tanımlanır ve READ TABLE veya INSERT komutlarıyla kullanılırlar. Bu yapıları kullanarak, verilerinizi hızlı ve etkili bir şekilde depolayabilir ve erişebilirsiniz.

STANDART TABLOYA ALAN EKLEME

SAP ABAP'ta standart bir tabloya alan eklemek için iki genel yöntem bulunmaktadır. Birincisi, tablo üzerinde doğrudan değişiklik yaparak yeni bir alan eklemek; ikincisi ise tabloya bağlı bir yapı oluşturarak bu yapıya yeni alanlar eklemek. İlk yöntemde, ABAP Dictionary (SE11) üzerinden tabloyu açarak "Append" seçeneğiyle yeni bir alan oluşturabilirsiniz. İkinci yöntemde ise bir yapı (structure) oluşturarak tabloya bu yapıyı ilişkilendirip, yapı üzerinde değişiklikler yaparak istenen alanları ekleyebilirsiniz. Bu yöntemlerle tabloya eklenen alanlar, iş sürekliliğini ve veri yönetimini sağlamak adına kullanışlı ve esnek bir şekilde genişletilebilir. Aşağıda bunun için bir örnek göstermek istedim.

Öncelikle bir structer yapısı oluşturdum.

```

82 ▶ DATA : BEGIN OF ls_ogrenci,
83 ▶     numara TYPE i,
84 ▶     adi TYPE string,
85 ▶     soyadi TYPE string,
86 ▶     yasi TYPE i,
87 ▶     END OF ls_ogrenci.

```

Oluşturduğum structure yapısından bir internal table tanımladım.

```

88 ▶
89 ▶ DATA : lt_ogrenci LIKE STANDARD TABLE OF ls_ogrenci.

```

Ardından structure alanlarına değerler atadım ve bu değerleri APPEND komutu ile internal table içerisine aktardım .

```

91 ▶ ls_ogrenci-numara=103.
92 ▶ ls_ogrenci-adi='Beyza Nur'.
93 ▶ ls_ogrenci-soyadi='KAMAŞ'.
94 ▶ ls_ogrenci-yasi=23.
95 ▶ APPEND ls_ogrenci TO lt_ogrenci.
96 ▶

```

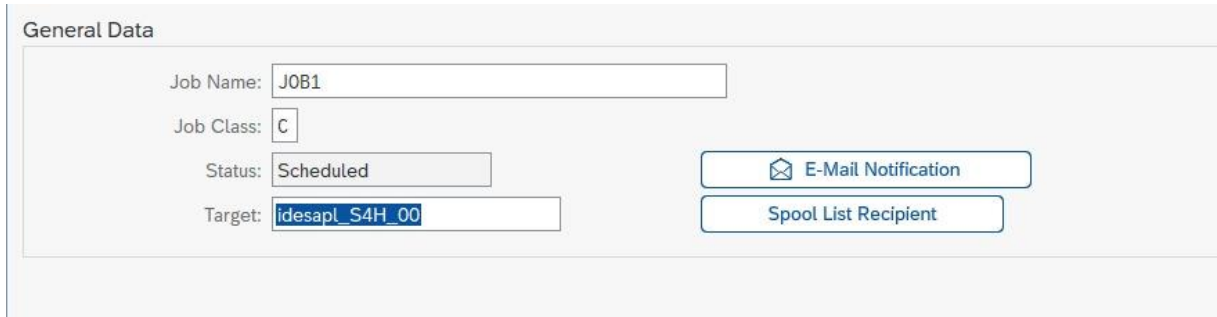
JOB OLUŞTURMA

ABAP'ta "job" terimi, arka planda çalışan ve belirli bir görevi veya işlemi gerçekleştiren programları ifade eder. Genellikle belirli bir zamanlama veya koşula bağlı olarak çalışırlar ve SAP sistemi içerisinde farklı işlemleri gerçekleştirmek için kullanılırlar.

Bu işler, "background processing" (arka planda işleme) olarak adlandırılır ve özellikle büyük veri işleme işlemleri, veri yedekleme, rapor oluşturma, veri temizleme gibi uzun süreli ve sistem kaynaklarını gerektiren görevler için kullanılır.

Job'lar, SAP sisteminde SM37 transaksyonu ile izlenebilir ve yönetilebilirler. Belirli zaman aralıklarıyla veya belirli koşullar sağlandığında çalışabilen bu işler, genellikle iş yükünü azaltmak, otomasyonu sağlamak ve sistem performansını artırmak için kullanılır. Örneğin, veri temizleme işlemleri gece saatlerinde otomatik olarak çalışacak şekilde planlanabilir ve bu iş, iş saatleri dışında sistem üzerindeki kullanıcı etkileşimini en aza indirir.

Peki job nasıl oluşturulur? İlk olarak basit bir program oluşturdum ve ardından bu programa bir varyant ekledim. Sonrasında sm36 T-Code ile JOB1 adında bir job oluşturdum.

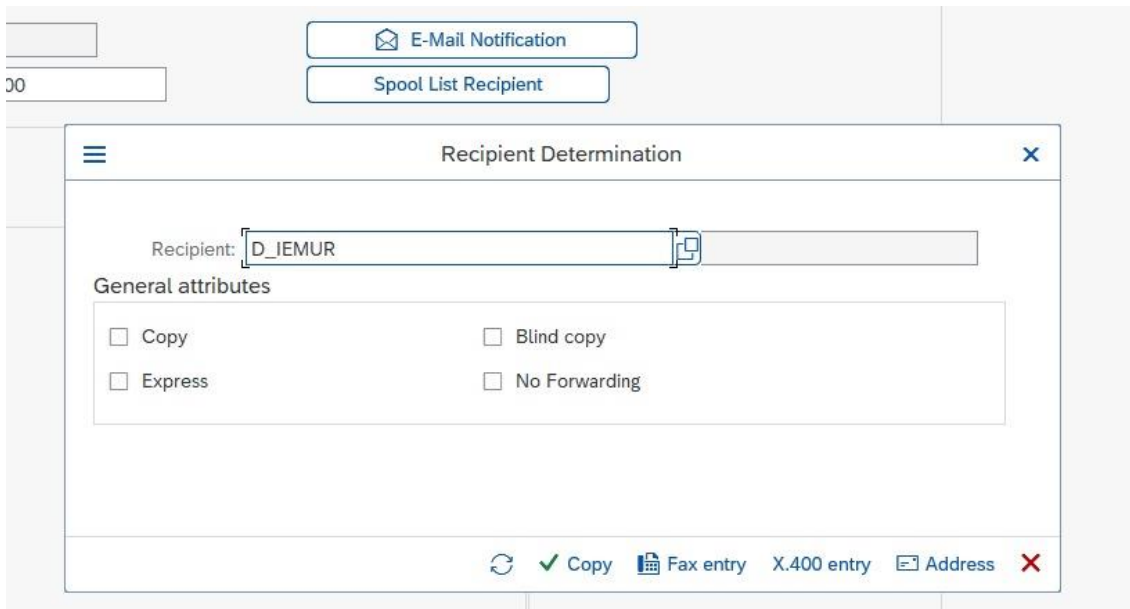


The screenshot shows the 'General Data' form for creating a job. The fields are as follows:

Field	Value
Job Name	JOB1
Job Class	C
Status	Scheduled
Target	idesapl_S4H_00

On the right side of the form, there are two buttons: 'E-Mail Notification' and 'Spool List Recipient'.

Burada target ismini seçtim ve 'Spool List Recipient' butonuna tıklayarak Recipient kısmına hocamızın kullanıcı adını ekledim.



The screenshot shows the 'Recipient Determination' dialog box. The 'Recipient' field contains 'D_IEMUR'. Below this, there are 'General attributes' with the following options:

Attribute	Value
Copy	<input type="checkbox"/>
Blind copy	<input type="checkbox"/>
Express	<input type="checkbox"/>
No Forwarding	<input type="checkbox"/>

At the bottom of the dialog, there is a row of icons: a refresh icon, a green checkmark, a fax icon, an X.400 icon, an address icon, and a red X icon.

Ardından 'Start condition' ile 'Immediate' alanından Periodic Job checkboxını işaretledim ve periyod değeri seçtim ve kaydettim.

Start conditionStepJob selectionOwn jobsJob wizardJob RepositoryMore

Start Time

ImmediateDate/TimeAfter JobAfter EventOperation Mode→

Date/Time

☒ Immediate Start

After Job

Operation Mode

After Event

☒ Periodic Job

Save

✗

✓ Check

Period values

Restrictions

Period Values

✗

Hourly

>

Daily

 <

Weekly

Monthly

Other period

Save

✗

✓ Check

Kayıt işlemini de tamamladıktan sonra artık sm37 kodu ile job durumunu görebiliriz.

Extended Job Selection

Information

More

Job Name:

User Name:

B_NURK

Job Status

☐ Sched.

☒ Released

☒ Ready

☒ Active

☒ Finished

☒ Canceled

Job Start Condition

From:

12/27/2023

To:

12/27/2023

From:

To:

Or after event:

Job Step

ABAP Program Name:

Refresh

Release

Spool

Job log

Step

Job details

Application servers

More

Job overview from: 12/27/2023 at: : :

to: 12/27/2023 at: : :

Selected job names: *

Selected user names: B_NURK

☐ Scheduled

☒ Released

☒ Ready

☒ Active

☒ Finished

☒ Canceled

☐ Event-Driven

Event ID:

☐ ABAP program

Program name :