اسکریپت ها و تشریح فرآیندهای دیتابیسی
روتین واحد دیتابیس

فهرست مطالب

تغییر دوره مالی از دائمی به ادواری (Daemi to Advari)

- برای این منظور ابتدا از دیتابیس بکاپ گرفته شود
- با رمز مدیر وارد شرکت مورد نظر شوید
- در وسط صفحه راست کلیک کرده و گزینه مدیریت دوره های مالی را انتخاب کنید
- دوره مالی مورد نظر را انتخاب کرده و دکمه اصلاح را بزنید
- بروی عنوان نوع دوره کلیک کرده و کلید ترکیبی مقابل را بگیرید. Alt+e+e+e
- امکان تغییر مقدار کمبو فعال خواهد شد نوع دوره مالی خود را به ادواری تغییر دهید و سپس تایید کنید
- در منوی حسابداری مدیریت حسابها در سطح کل برای گروه حسابهای خرید و برگشت از خرید و تخفیفات خرید براساس سیاست کاربر حسابهای لازم را تعریف کنید.
- در منوی حسابداری در بخش مدیریت حسابهای ویژه ارتباط حسابهای ایجاد شده را با حسابهای متناظر در سیستم ایجاد کنید.
- دوره مالی مورد نظر را براساس اطلاعات جدول __FiscalPeriod__ تعیین کنید.
- در صورتی که دوره مالی دارای فاکتورهای خرید و یا برگشت از خرید بود بردار حساب متناظر برای فاکتور های خرید و برگشت از خرید در جدول __SPFactor__ بروزرسانی می کنیم
- در صورت عدم تمایل کاربر برای استفاده از شناور، مرکز هزینه و یا پروژه عدد صفر درج می گردد.
- تعیین کدحساب ضروری است.
- برای انجام موارد فوق در اس کیو ال سرور یک کوئری بازکنید و دستورات زیر را اجرا کنید

```
select * from __FiscalPeriod__
select SPAccountId,SPFAccId,SPCCId,SPPrjId
from __SPFactor__ where fpid=3 and factortype in (0,2)

update __SPFactor__ set SPAccountId='hessab kharid',SPFAccId=0,SPCCId=0,SPPrjId=0 where fpid=3 and
FactorType=0
update __SPFactor__ set SPAccountId='hessab bargasht az Kharid',SPFAccId=0,SPCCId=0,SPPrjId=0 where
fpid=3 and FactorType=2
```

- سپس به کاربر اعلام می شود تا فرآیند بازسازی کاردکس را انجام دهند.

تغییر دوره مالی از ادواری به دائمی (AdvariToDaemi)

- این فرآیند کاملا مشابه فرآیند فوق است با این تفاوت که در دوره مالی دائمی میباست حسابی برای بهای تمام شده در نظر گرفته شود. و دیگر نیازی به حسابهای خرید نیست.
- در اس کیو ال سرور دستورات زیر را اجرا می کنیم

```
Declare @pFPId INT =4

select * from __SPFactor__
where (FactorType=0 or FactorType=2) and FPId=@pFPId

update __SPFactor__
set SPAccountId=0,SPFAccId=0,SPCCId=0,SPPrjId=0
where factortype in (0,2) and FPId =@pFPId
```

مشکل آی دی منفی سند (idManfisanad)

در برخی مواقع به دلیل وجود چندین دوره مالی و یا انتقال اسناد از شرکت های دیگر و سایر موارد شناسه سندی با عدد بالا در جدول درج می شود و از آنجایی که سیستم براساس بزرگترین شناسه عدد شناسه جدید را ایجاد می کند و نوع فیلد در نظر گرفته شده برای شناسه جدول سند integer مثبت می باشد. در صورتی که آخرین شناسه از رنج اعداد صحیح مثبت بیشتر شود اصطلاحا overflow یا سریز رخ می دهد و بیت علامت به عنوان بخشی از داده مقداردهی خواهد شد که موجب به وجود آمدن عدد منفی در لیست شناسه می گردد. و مشکلاتی را برای کار با سیستم فرهم می آورد. لذا در صورت بروز مشکل و یا مشاهده ارقام بالا در شناسه سند روال زیر را انجام می دهیم.

- برای این منظور ابتدا از دیتابیس بکاپ گرفته شود
- سپس یک کپی از جدول __Transaction__ که اطلاعات سند را نگهداری می کند با یک فیلد اضافه برای محاسبه شناسه جدید ایجاد می کنیم:

```
CREATE TABLE [dbo].[UD__Transaction__](
	[Id] [int] NOT NULL,
	[SanadNo] [int] NOT NULL,
	[TDate] [datetime] NULL,
	[Committed] [smallint] NULL DEFAULT ((0)),
	[Balanced] [smallint] NULL DEFAULT ((0)),
	[Type] [smallint] NULL DEFAULT ((0)),
	[UserId] [smallint] NULL,
	[FPId] [smallint] NULL,
	[TDesc] [varchar](255) NULL,
	[LRes] [smallint] NULL DEFAULT ((0)),
	[DRes] [float] NULL DEFAULT ((0)),
	[TRes] [varchar](32) NULL DEFAULT ('0'),
	[SabtCount] [smallint] NULL DEFAULT ((0)),
	[UserName] [varchar](64) NULL DEFAULT ('---'),
	[UserId2] [smallint] NULL DEFAULT ((0)),
	[UserName2] [varchar](64) NULL DEFAULT ('---'),
	[UserId3] [smallint] NULL DEFAULT ((0)),
	[UserName3] [varchar](64) NULL DEFAULT ('---'),
	[NewId] [int] NULL)
```

- فیلد [NewId] برای ذخیره شناسه جدید استفاده خواهد شد
- اطلاعات جدول اصلی را در جدول ایجاد شده ذخیره می کنیم

```
INSERT INTO [dbo].[UD__Transaction__]
	([Id],[SanadNo],[TDate],[Committed],[Balanced],[Type],[UserId],[FPId],[TDesc]
	,[LRes],[DRes],[TRes],[SabtCount],[UserName],[UserId2],[UserName2],[UserId3]
	,[UserName3],[NewId])
SELECT [Id],[SanadNo],[TDate],[Committed],[Balanced]
	,[Type],[UserId],[FPId],[TDesc],[LRes],[DRes],[TRes],[SabtCount],[UserName]
	,[UserId2],[UserName2],[UserId3],[UserName3],ROW_NUMBER()over(order by SanadNo,TDate,FPId,Id)
FROM __Transaction__
ORDER BY  SanadNo,TDate,FPId,Id
```

- حال در جدول موقت اطلاعات سند به همراه ستون [NewId] شماره ترتیبی از یک را برای اسناد از کوچکترین را داراست که از آن به عنوان شناسه جدید سند استفاده خواهیم کرد.

```
UPDATE T set Id=T1.[NewId]
FROM __Transaction__ T inner join UD__Transaction__ T1
                    ON T.sanadNo=T1.sanadno and T.FPId=T1.FPId AND T.Id=T1.Id
```

- از آنجایی که ارتباطات در طراحی در نظر گرفته شده اند با بروزرسانی جدول سند فیلد متناظر در جدول آرتیکل به طور آبشاری بروز خواهد شد. ارتباط اسناد با سایر زیر سیستم ها از طریق فیلد های موجود در جدول آرتیکل نگهداری و مدیریت می گردد و تنها زیر سیستمی که این مورد را شامل نمی شود زیر سیستم اموال است. لذا در صورتی که شرکت دارای زیرسیستم اموال باشد، میبایست اطلاعات جدول اموال نیز بروزرسانی شود.

```
UPDATE A SET TransId=T1.[NewId]
FROM __AssetFinancialArtcl__ A inner join UD__Transaction__ T1
                    ON A.FPId=T1.FPId AND A.TransId=T1.Id
```

افزایش و کاهش طول کد کالا

در نرم افزار تدبیر امکان تعریف کالا تا ۷ سطح وجود دارد که سطح هفتم میبایست حتما از نوع کالا باشد و نمی تواند گروه باشد. این امکان در نسخه لایت تا سه سطح میباشد.

سطح بندی کالا در تدبیر از صفر می شود و سطوح شامل ۰-۶ است

اطلاعات طول کد برای هر سطح در جدول __MerchCodeLength__ ذخیره شده است ، فیلد LevelId اشاره به شماره سطح و فیلد Length به طول در نظر گرفته شد برای آن سطح اشاره دارد.
اولین کاری که قبل انجام هرگونه فرآیندی روی دیتابیس صورت می گیرد تهیه بکاپ است.
پس از آن براساس نظر کاربر در دوره مالی آخر یا در دوره های فعال کاربر افزایش طول کد در سطح یا سطوح مورد نظر کاربر انجام می گیرد. به عنوان مثال اگر در سال مالی ۱۴۰۲ هستیم و کاربر هنوز در حال مرتب سازی دوره مالی ۱۴۰۱ است بهتر است تغییرات در هر دو دوره انجام شود با این حال کار بنا برنظر کاربر انجام خواهد شد.

افزایش طول کد کالا (MerchLenght_ADD)

با توجه به اینکه نیاز به تغییرات در کدام سطح است و قرار است تا چه تعداد کراکتر به آن سطح اضافه شود تعدادی صفر قبل از کد سطح مورد نظر قرار خواهد گرفت تا طول کد آن سطح متناظر با مقدار مورد درخواست کاربر گردد به عنوان مثال در صورتی که بخواهیم سطح دوم را از ۲ رقم به ۳ رقم تغییر دهیم ابتدا در جدول تنظیمات عدد مربوط به سطح ۱ (شمارش سطح از صفر شروع می شود) را به ۳ تغییر می دهیم و سپس در کد ها به فیلد G2 به قبل از مقدار آن یک صفر اضافه می کنیم.
با توجه به داده های اعلامی کاربر و تنظیمات سیستم می توان از یک یا چند اسکریپت زیر برای انجام تغییرات استفاده کرد

```
Declare @pFPId int =2
--صفر سطح (کالا مستقیم تعریف)
begin tran A
UPDATE __MerchCodeLength__ SET Length=3 WHERE LevelId=0
UPDATE __Merchandise__ SET G1='00'+G1 WHERE Mlevel>=0 and FPId=@pFPId
UPDATE __Merchandise__ SET code=G1 WHERE MLevel=0 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2  WHERE MLevel=1 and FPId=@pFPId
```

```
UPDATE __Merchandise__ SET Code=G1+G2+G3  WHERE MLevel=2 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3+G4  WHERE MLevel=3 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3+G4+G5  WHERE MLevel=4 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3+G4+G5+G6  WHERE MLevel=5 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3+G4+G5+G6+G7  WHERE MLevel=6 and FPId=@pFPId
commit tran A
```

```
--کالا مستقیم تعریف) یک سطح)
begin tran A
UPDATE __MerchCodeLength__ SET Length=5 WHERE LevelId=1
UPDATE __Merchandise__ SET G2='00'+G2 WHERE MLevel>0 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2  WHERE MLevel=1 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3  WHERE MLevel=2 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3+G4  WHERE MLevel=3 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3+G4+G5  WHERE MLevel=4 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3+G4+G5+G6  WHERE MLevel=5 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3+G4+G5+G6+G7  WHERE MLevel=6 and FPId=@pFPId
commit tran A
```

```
--کالا مستقیم تعریف) دو سطح)
begin tran A
UPDATE __MerchCodeLength__ SET Length=5 WHERE LevelId=2
UPDATE __Merchandise__ SET G3='00'+G3 WHERE MLevel>1 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3  WHERE MLevel=2 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3+G4  WHERE MLevel=3 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3+G4+G5  WHERE MLevel=4 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3+G4+G5+G6  WHERE MLevel=5 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3+G4+G5+G6+G7  WHERE MLevel=6 and FPId=@pFPId
commit tran A
```

```
--کالا مستقیم تعریف) سه سطح)
begin tran A
UPDATE __MerchCodeLength__ SET Length=5 WHERE LevelId=3
UPDATE __Merchandise__ SET G4='00'+G4 WHERE MLevel>2 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3+G4  WHERE MLevel=3 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3+G4+G5  WHERE MLevel=4 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3+G4+G5+G6  WHERE MLevel=5 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3+G4+G5+G6+G7  WHERE MLevel=6 and FPId=@pFPId
commit tran A
```

```
--کالا مستقیم تعریف) چهار سطح)
begin tran A
UPDATE __MerchCodeLength__ SET Length=5 WHERE LevelId=4
UPDATE __Merchandise__ SET G5='00'+G5 WHERE MLevel>3 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3+G4+G5  WHERE MLevel=4 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3+G4+G5+G6  WHERE MLevel=5 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3+G4+G5+G6+G7  WHERE MLevel=6 and FPId=@pFPId
commit tran A
```

```
--کالا مستقیم تعریف) پنج سطح)
begin tran A
UPDATE __MerchCodeLength__ SET Length=5 WHERE LevelId=5
UPDATE __Merchandise__ SET G6='00'+G6 WHERE MLevel>4 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3+G4+G5+G6  WHERE MLevel=5 and FPId=@pFPId
```

```
UPDATE __Merchandise__ SET Code=G1+G2+G3+G4+G5+G6+G7  WHERE MLevel=6 and FPId=@pFPId
commit tran A
```

```
--(کالا مستقیم تعریف) شش سطح
begin tran A
UPDATE __MerchCodeLength__ SET Length=5 WHERE LevelId=6
UPDATE __Merchandise__ SET G7='00'+G7 WHERE MLevel>5 and FPId=@pFPId
UPDATE __Merchandise__ SET Code=G1+G2+G3+G4+G5+G6+G7  WHERE MLevel=6 and FPId=@pFPId
commit tran A
```

کاهش طول کد کالا (MerchLenght_SUB)

در کاهش نکته حائز اهمیت این است که قبل از هر گونه تغییراتی در ابتدا بررسی کنیم که آیا کاهش طول کد در سطح درخواستی موجود ایجاد کد کالای تکراری نگردد. و در صورت بروز این شرایط اعلام می گردد که کاهش امکان پذیر نمی باشد.
در صورتی که بررسی های لازم برای اعتبارسنجی عدم بروز کد تکراری در کاهش انجام و مشکلی مشاهده نشد براساس مقادیر درخواستی کاربر اطلاعات در جدوال تراز خواهند شد. میتوانید از کد زیر به عنوان نمونه استفاده کنید و فقط لازم است برای سطح مورد نظر این کار انجام شود و کد زیر فقط به عنوان راهنمای انجام کار است.

```
DECLARE @pFPId int =2

begin tran A
UPDATE __MerchCodeLength__ SET [Length]=2 WHERE LevelId=0
UPDATE __MerchCodeLength__ SET [Length]=3 WHERE LevelId=1
UPDATE __MerchCodeLength__ SET [Length]=3 WHERE LevelId=2
UPDATE __MerchCodeLength__ SET [Length]=3 WHERE LevelId=3
UPDATE __MerchCodeLength__ SET [Length]=3 WHERE LevelId=4
UPDATE __MerchCodeLength__ SET [Length]=3 WHERE LevelId=5
UPDATE __MerchCodeLength__ SET [Length]=3 WHERE LevelId=6
UPDATE __Merchandise__  SET G1=substring(G1,4,2) WHERE Id<>0 AND FPId= @pFPId
UPDATE __Merchandise__  SET G2=substring(G2,10,3) WHERE Id<>0 AND FPId= @pFPId AND MLevel>=1
UPDATE __Merchandise__  SET G3=substring(G3,14,3) WHERE Id<>0 AND FPId= @pFPId AND MLevel>=2
UPDATE __Merchandise__  SET G4=substring(G4,3,3) WHERE Id<>0 AND FPId= @pFPId AND MLevel>=3
UPDATE __Merchandise__  SET G5=substring(G5,3,3) WHERE Id<>0 AND FPId= @pFPId AND MLevel>=4
UPDATE __Merchandise__  SET G6=substring(G6,3,3) WHERE Id<>0 AND FPId= @pFPId AND MLevel>=5
UPDATE __Merchandise__  SET G7=substring(G7,3,3) WHERE Id<>0 AND FPId= @pFPId AND MLevel>=6
UPDATE __Merchandise__  SET Code=G1 WHERE MLevel=0 AND FPId= @pFPId
UPDATE __Merchandise__  SET Code=G1+G2 WHERE MLevel=1 AND FPId= @pFPId
UPDATE __Merchandise__  SET Code=G1+G2+G3 WHERE MLevel=2 AND FPId= @pFPId
UPDATE __Merchandise__  SET Code=G1+G2+G3+G4 WHERE MLevel=3 AND FPId= @pFPId
UPDATE __Merchandise__  SET Code=G1+G2+G3+G4+G5 WHERE MLevel=4 AND FPId= @pFPId
UPDATE __Merchandise__  SET Code=G1+G2+G3+G4+G5+G6 WHERE MLevel=5 AND FPId= @pFPId
UPDATE __Merchandise__  SET Code=G1+G2+G3+G4+G5+G6+G7 WHERE MLevel=6 AND FPId= @pFPId
commit tran A
```

افزایش یا کاهش طول کد تفصیلی شناور

اطلاعات تفصیلی های شناور در جدول __DetailAcc__ ذخیره می گردد و در واقع شامل سطح چهارم یا بیشتر کدینگ حسابداری می شود. و کد یک شناور براساس تجمیع اطلاعات موجود در فیلدهای T برای شماره سطح ساخته می شود. در سیستم تدبیر امکان گروهبندی شناور تا ۷ سطح وجود دارد و طول کد در هر سطح حداکثر ۹ رقم است ولی باید توجه داشته باشید از انجایی که در سیستم برای نمایش کد کامل شناور از نوع varchar(64) استفاده می شود امکان استفاده از تمامی سطوح با حداکثر طول وجود ندارد.

افزایش طول کد شناور (Tafsil_ADD)

- برای افزایش طول کد طبق روال زیر عمل خواهیم کرد
- در ابتدا از دیتا بیس بکاپ می گیریم
- براساس نظر کاربر مشخص می کنیم در کدام سطح یا سطوح افزایش داریم و رقم نهایی چند رقم خواهد بود
- از اسکریپت زیر می توانید به عنوان راهنما در انجام کار استفاده کنید

```
DECLARE @pFPId INT =1
--شناور اول سطح کد طول افزایش
UPDATE __CodeLength__ SET [Length]=5 WHERE LevelId=4
UPDATE __DetailAcc__ SET T1='00'+T1 WHERE FPId=@pFPId and AccLevel>=4 and Id<>0

--دوم سطح در شناور کد طول افزایش
UPDATE __CodeLength__ SET [Length]=4 WHERE LevelId=5
UPDATE __DetailAcc__ SET T2='00'+T2 WHERE FPId=@pFPId and AccLevel>=5 and Id<>0

--سوم سطح در شناور کد طول افزایش
UPDATE __CodeLength__ SET [Length]=4 WHERE LevelId=6
UPDATE __DetailAcc__ SET T3='00'+T3 WHERE FPId=@pFPId and AccLevel>=6 and Id<>0

--چهارم سطح در شناور کد طول افزایش
UPDATE __CodeLength__ SET [Length]=4 WHERE LevelId=7
UPDATE __DetailAcc__ SET T4='00'+T4 WHERE FPId=@pFPId and AccLevel>=7 and Id<>0

UPDATE AD SET DetFullId=(SELECT CASE AccLevel WHEN 4 THEN T1 WHEN 5 THEN T1+T2
                                              WHEN 6 THEN T1+T2+T3 WHEN 7 THEN T1+T2+T3+T4
                                              WHEN 8 THEN T1+T2+T3+T4 ELSE '0' END
                         FROM __DetailAcc__ WHERE Id=AD.DetId AND FPId=AD.FPId)
FROM __AccVsDetail__ AD
WHERE FPId=@pFPId
```

- تعداد صفر های قبل از سطح براساس تعداد جاری طول کد در آن سطح و تعداد مورد درخواست تعیین می گردد.

کاهش طول کد شناور(Tafsil_SUB)

برای کاهش ابتدا باید اطمینان حاصل کنیم که حذف تعداد رقم درخواستی کاربر از سمت چپ موجب ایجاد کد تکراری برای شناور نشود. در صورتی که این مهم را بررسی کردید و اطمینان حاصل کردید که تغییرات موجب ناسازگاری در سیستم نخواهد شد طبق روال زیر عمل خواهیم کرد

- در ابتدا از دیتا بیس بکاپ می گیریم
- براساس نظر کاربر مشخص می کنیم در کدام سطح یا سطوح کاهش داریم و رقم نهایی چند رقم خواهد بود
- از اسکریپت زیر می توانید به عنوان راهنما در انجام کار استفاده کنید

```
DECLARE @pFPId INT =1
--شناور اول سطح کد طول کاهش
UPDATE __CodeLength__ SET [Length]=2 WHERE LevelId=4
UPDATE __DetailAcc__ SET T1=RIGHT(T1,2) WHERE FPId=@pFPId and AccLevel>=4 and Id<>0

--دوم سطح در شناور کد طول کاهش
```

```
UPDATE __CodeLength__ SET [Length]=3 WHERE LevelId=5
UPDATE __DetailAcc__ SET T2=RIGHT(T2,3) WHERE FPId=@pFPId and AccLevel>=5 and Id<>0

--سوم سطح در شناور کد طول کاهش
UPDATE __CodeLength__ SET [Length]=3 WHERE LevelId=6
UPDATE __DetailAcc__ SET T3=RIGHT(T3,3) WHERE FPId=@pFPId and AccLevel>=6 and Id<>0

--چهارم سطح در شناور کد طول کاهش
UPDATE __CodeLength__ SET [Length]=2 WHERE LevelId=7
UPDATE __DetailAcc__ SET T4=RIGHT(T4,2) WHERE FPId=@pFPId and AccLevel>=7 and Id<>0

-----------------------------------------
UPDATE AD SET DetFullId=(SELECT CASE AccLevel WHEN 4 THEN T1 WHEN 5 THEN T1+T2
                                              WHEN 6 THEN T1+T2+T3 WHEN 7 THEN T1+T2+T3+T4
                                              WHEN 8 THEN T1+T2+T3+T4 ELSE '0' END
                         FROM __DetailAcc__ WHERE Id=AD.DetId AND FPId=AD.FPId)
FROM __AccVsDetail__ AD
WHERE FPId=@pFPId
```

مرتب سازی فرم های دریافت و پرداخت خزانه (SortPayRCV)

مرتب سازی فرم ها براساس نوع فرم به تفکیک صورت می گیرید و در بروزرسانی آرتیکل های سند مرتبط یا کاربر هر یک از فرم های دریافت و پرداخت نقدی خود را تک تک اصلاح و ثبت میزند تا شرح آرتیکل سند مرتبط براساس تنظیمات انجام شده در سیستم مجدد ایجاد شود، در صورت عدم تمایل کاربر به انجام این کار امکان بروزرسانی کلیه شرح ها به صورت استاندارد وجود دارد که می توانند شامل شماره فرم شرح فرم نام شخص طرح حساب فرم باشد که موارد کاملا براساس نظر کاربر چیده و کم یا زیاد می شود.

- برای انجام این کار ابتدا از دیتا بیس بکاپ می گیریم
- سپس دوره مالی مورد نظر کاربر را تعیین می کنیم
- نهایتا فرم ها براساس تاریخ مرتب سازی خواهد شد
- در صورت تمایل کاربر شرح آرتیکل سندهای مرتب با دریافت و پرداخت های نقدی بروز خواهند شد. در نظر داشته باشید فرم های دریافت و پرداخت می توانند شامل اطلاعات نقدی و چک باشند که بروزرسانی فقط برای فرم های نقدی صورت خواهد گرفت زیرا در شرح آرتیکل های مرتبط با چک شماره فرم درج نمی گردد و فقط وضعیت چک و طرف حساب حساب در شرح نمایان می شود در نتیجه تغییر شماره فرم مرتبط تاثیری در شرح آرتیکل مرتبط نخواهد داشت.
- جهت راهنما می تواند از اسکریپت زیر به عنوان راهنما در انجام کار استفاده کنید :

```
DECLARE @pFPId int =8

UPDATE __PayRcv__ SET Reference=PNo WHERE FPId=@pFPId AND Id<>0

UPDATE __PayRcv__ SET PNo=cte.pNON
FROM __PayRcv__ INNER JOIN (SELECT Id,PNo,dbo.MTOS(PDate) AS PDate,ROW_NUMBER()
over(order by PDate,Id) AS PNON FROM __PayRcv__
WHERE FPId=@pFPId AND PayType=0) AS cte ON cte.Id=__PayRcv__.Id
WHERE __PayRcv__.FPId=@pFPId AND __PayRcv__.PayType=0

UPDATE __PayRcv__ SET PNo=cte.pNON
FROM __PayRcv__ INNER JOIN (SELECT Id,PNo,dbo.MTOS(PDate) AS PDate,ROW_NUMBER()
over(order by PDate,Id) AS PNON FROM __PayRcv__
WHERE FPId=@pFPId AND PayType=1) AS cte ON cte.Id=__PayRcv__.Id
WHERE __PayRcv__.FPId=@pFPId AND __PayRcv__.PayType=1
```

```
UPDATE __PayRcv__ SET PNo=cte.pNON
FROM __PayRcv__ INNER JOIN (SELECT Id,PNo,dbo.MTOS(PDate) AS PDate,ROW_NUMBER()
over(order by PDate,Id) AS PNON FROM __PayRcv__
WHERE FPId=@pFPId AND PayType=2) AS cte ON cte.Id=__PayRcv__.Id
WHERE __PayRcv__.FPId=@pFPId AND __PayRcv__.PayType=2

----------نمونه عنوان به مرتبط سندهای آرتیکل شرح رسانی بروز
UPDATE A SET Adesc =(SELECT dbo.Verify1256Text('شماره دریافت سند طی دریافت بابت '
        +(SELECT dbo.ENumToFNum(PNO) FROM __PayRcv__ WHERE FPId=A.FPId AND Id=PA.PayId))
        +' '+(SELECT PDesc FROM __PayRcv__ WHERE FPId=A.FPId AND Id=PA.PayId)
        +' '+(SELECT Name FROM __DetailAcc__ WHERE FPId=A.FPId AND Id=A.FAccId)
        )
FROM __Article__ A INNER JOIN __PayRcvArticle__ PA ON A.CashTransId=PA.CashId AND A.FPId=PA.FPId
AND A.CashTransId<>0
WHERE A.FPId=@pFPId AND cAShtransId<>0 AND A.PayType=-1

UPDATE A SET Adesc =(SELECT dbo.Verify1256Text('شماره پرداخت سند طی پرداخت بابت '
        +(SELECT dbo.ENumToFNum(PNO) FROM __PayRcv__ WHERE FPId=A.FPId AND Id=PA.PayId))
        +' '+(SELECT PDesc FROM __PayRcv__ WHERE FPId=A.FPId AND Id=PA.PayId)
        +' '+(SELECT Name FROM __DetailAcc__ WHERE FPId=A.FPId AND Id=A.FAccId)
        )
FROM __Article__ A INNER JOIN __PayRcvArticle__ PA ON A.CashTransId=PA.CashId AND A.FPId=PA.FPId
AND A.CashTransId<>0
WHERE A.FPId=@pFPId AND CashTransId<>0 AND A.PayType=1
```

مرتب سازی فاکتور و بروزرسانی شرح های فرم های مرتبط (FactorNO&Descr)

مرتب سازی فاکتور ها به معنای شماره گذاری مجدد براساس تاریخ از کوچک به بزرگ است که بنا به درخواست کاربر انجام می گیرد. کاربر میتواند فقط یک نوع فاکتور و همه نوع های فاکتور را سورت کند. در مرتب سازی هر نوع فاکتور ملاحظاتی وجود دارد که باید قبل از مرتب سازی در نظر گرفته شوند.

در مرتب سازی فاکتور فروش و خرید این نکته حائز اهمیت است که ارتباط میان فاکتور فروش و فاکتور برگشت از فروش متناظر، براساس شماره فاکتور فروش تنظیم می گردد، از این جهت در صورتی که کاربر دارای فاکتور برگشتی باشد میبایست ارتباطات حفظ شود.

در مرتب سازی پیش فاکتور ها نکته حائز اهمیت ذخیره ارتباط پیش فاکتور با درخواست تحویل کالا به مشتری براساس شماره پیش فاکتور است و اگر کاربر از فرم های درخواست تحویل استفاده می کند میبایست ارتباطات بعد از مرتب سازی حفظ شوند. برای سورت فاکتور به شرح ذیل عمل می کنیم.

- ابتدا بکاپ می گیریم
- یک کپی از جدول فاکتورها ایجاد می کنیم
- اطلاعات جدول فاکتور را در جدول ایجاد شده میریزیم.
- فاکتورها را براساس روال درخواستی کاربر شماره گذاری می کنیم
- ارتباط فاکتورهای برگشتی را بروز می کنیم
- در صورت سورت پیش فاکتور ارتباط پیش فاکتور و درخواست تحویل را بروز می کنیم
- در سیستم برای اضافات مرتبط با فاکتور شرحی ثبت شده است که شماره فاکتور در آن نوشته شده است در صورتی که فاکتور دارای اضافات یا کسورات باشد، شرح در جدول مرتبط به صورت استاندارد بروز خواهد شد.
- اگر فاکتور ها دارای تسویه باشند شرح فرم تسویه و فرم دریافت یا پرداخت مرتبط نیز اصلاح می گردد.

- اگر به هر دلیلی کاربر امکان بازسازی کاردکس نداشته باشید می توان باساس یک شرح استاندارد شرح ردیف سند مرتبط در حسابداری و شرح فرم انبار دیتا بیسی بروز شود در غیر این صورت تریج بر انجام فرآیند بازسازی کاردکس است.
- می توانید از کد زیر به عنوان راهنما استفاده کنید:

```
declare @FPId as smallint
set @FPId=1

select * from __SPFactor__ where FactorType=1 and FPId=@FPId
select * from __SPFactor__ where FactorType=3 and FPId=@FPId
select * from __SPCost__ SC where FPId=@FPId and Id<>0
select * from __Article__ AD where invSPId<>0 and InvSPId in (select Id from __SPFactor__ where FactorType=1
and Id<>0 and FPId=@FPId)and FPId=@FPId
select * from __SPTasvie__ where SPId in (select Id from __SPFactor__ where FactorType=1 and Id<>0 and
FPId=@FPId) and FPId=@FPId

CREATE TABLE [dbo].[UD__SPFactor__](
        [Id] [int] NOT NULL,
        [FactorNo] [int] NOT NULL,
        [SPDate] [datetime] NOT NULL DEFAULT ((0.0)),
        [FactorType] [smallint] NOT NULL,
        [Committed] [smallint] NOT NULL DEFAULT ((0)),
        [SPReference] [int] NULL,
        [AccountId] [varchar](64) NOT NULL,
        [SPAccountId] [varchar](64) NOT NULL,
        [CustomerId] [int] NULL,
        [FAccId] [int] NULL,
        [PreAccId] [varchar](64) NULL DEFAULT ('0'),
        [PreAccPercent] [float] NULL DEFAULT ((0)),
        [Total] [float] NOT NULL,
        [Discount] [float] NULL DEFAULT ((0)),
        [Expense] [float] NULL DEFAULT ((0)),
        [SPDesc] [varchar](255) NULL,
        [CustomerName] [varchar](128) NULL,
        [CustomerPhoneNo] [varchar](64) NULL,
        [CustomerEcCode] [varchar](64) NULL,
        [CustomerAddress] [varchar](255) NULL,
        [BrokerId] [int] NULL DEFAULT ((0)),
        [BrokerShare] [float] NULL DEFAULT ((0)),
        [BrokerPercent] [float] NULL DEFAULT ((0)),
        [CurrencyVal] [float] NULL,
        [CurrencyId] [smallint] NULL,
        [LRes] [int] NULL DEFAULT ((0)),
        [DRes] [float] NULL DEFAULT ((0)),
        [TRes] [varchar](64) NULL DEFAULT ('1'),
        [UserId] [smallint] NOT NULL,
        [SecId] [smallint] NULL DEFAULT ((0)),
        [ETime] [datetime] NULL DEFAULT (getdate()),
        [EDate] [datetime] NULL DEFAULT (getdate()),
        [FPId] [smallint] NOT NULL,
        [CCId] [int] NULL DEFAULT ((0)),
        [PrjId] [int] NULL DEFAULT ((0)),
        [SPFAccId] [int] NULL DEFAULT ((0)),
        [SPCCId] [int] NULL DEFAULT ((0)),
        [SPPrjId] [int] NULL DEFAULT ((0)),
```

```
	[RONo] [int] NULL DEFAULT ((0)),
	[SPRefNo] [varchar](32) NULL,
	[ContractNo] [varchar](32) NULL,
	[ShipmentNo] [varchar](32) NULL,
	[Cancelled] [smallint] NULL DEFAULT ((0)),
	[SvcJobNo] [int] NULL DEFAULT ((0)),
	[SvcDesc] [varchar](255) NULL,
	[SvcCommPercent] [float] NULL DEFAULT ((0)),
	[SvcCommAmount] [float] NULL DEFAULT ((0)),
	[PreFAccId] [int] NULL DEFAULT ((0)),
	[PreCCId] [int] NULL DEFAULT ((0)),
	[PrePrjId] [int] NULL DEFAULT ((0)),
	[PRetFAccId] [int] NULL DEFAULT ((0)),
	[PRetCCId] [int] NULL DEFAULT ((0)),
	[PRetPrjId] [int] NULL DEFAULT ((0)),
	[TEnable] [int] NULL DEFAULT ((0)),
	[TValue] [float] NULL DEFAULT ((0)),
	[TRS] [int] NULL DEFAULT ((0)))
```

کپی اطلاعات در جدول موقت---

```
INSERT INTO [dbo].[UD__SPFactor__]
        ([Id],[FactorNo],[SPDate],[FactorType],[Committed],[SPReference],[AccountId]
                ,[SPAccountId],[CustomerId],[FAccId],[PreAccId],[PreAccPercent],[Total]
                ,[Discount],[Expense],[SPDesc],[CustomerName],[CustomerPhoneNo]
        ,[CustomerEcCode],[CustomerAddress],[BrokerId],[BrokerShare],[BrokerPercent]
,[CurrencyVal],[CurrencyId],[LRes],[DRes],[TRes],[UserId],[SecId],[ETime]
,[EDate],[FPId],[CCId],[PrjId],[SPFAccId],[SPCCId],[SPPrjId],[RONo],[SPRefNo]
,[ContractNo],[ShipmentNo],[Cancelled],[SvcJobNo],[SvcDesc],[SvcCommPercent]
,[SvcCommAmount],[PreFAccId],[PreCCId],[PrePrjId],[PRetFAccId],[PRetCCId]
                ,[PRetPrjId],[TEnable],[TValue],[TRS])
select * from __SPFactor__ where FPId=@pFPId AND Id<>0
```

سورت شماره فاکتور فروش براساس تاریخ و ساعت----

```
update SP set FactorNo=cte.FNoNew
from __SPFactor__ SP inner join (select Id,RONo,dbo.mtos(SPDate) as
SPDate,ETime,ROW_NUMBER()over(order by SPDate,ETime,id) as FNoNew from __SPFactor__
 where FPId=@pFPId  and FactorType=1) as cte on cte.Id=SP.Id
where SP.FPId=@pFPId and SP.FactorType=1
```

ایجاد ارتباط فاکتورهای برگشتی با فاکتور فروش مرتبط----

```
update SP set Expense =isnull((select factorno from __spfactor__ where Id=(select Id from  UD__spfactor__ where
factorno=sp1.Expense and FPId=sp1.FPId and FactorType=1)),0)
from __SPFactor__ SP inner join UD__spfactor__  SP1 on SP.Id=SP1.Id and SP.FPId=SP1.FPId
where SP.factortype=3 and SP.FPId=@pFPId
```

اصلاح شرح اضافات فاکتور----

```
update SC set SPCDesc=ltrim(rtrim(convert(char,(select FactorNo from __SPFactor__ where Id=SC.SPId and
FPId=@FPId and FactorType=1))))+(select Name from __Account__ where FullId=SC.CostAccount and
FPId=@FPId)+'-'+(select CustomerName from __SPFactor__ where Id=Sc.SPId and FPId=@FPId and
FactorType=1)+' شماره فاکتور طي '
from __SPCost__ SC where FPId=@FPId and Id<>0
```

اصلاح شرح تسویه و فرم دریافت مرتبط ----

```
update SPT set VDesc=' فاکتور تسویه بابت' + ltrim(rtrim(convert(char,(select FactorNo from __SPFactor__ where
Id=SPT.SPId and FPId=@FPId and FactorType=1)))) +' سند' +ltrim(rtrim((select PNo from __PayRcv__ where
Id=SPT.RefId)))
```

```
from __SPTasvie__ SPT
where SPId in (select Id from __SPFactor__ where FactorType=1 and Id<>0 and FPId=@FPId) and FPId=@FPId
and RefType=1 and RefId2='' --- hanooz baressi nakardam

update SPT set VDesc='شماره فاکتور تسویه بابت '+ ltrim(rtrim(convert(char,(select FactorNo from __SPFactor__
where Id=SPT.SPId and FPId=@FPId and FactorType=1)))) +'شماره چک '+ ltrim(rtrim((select CheckNo from
__Check__ where Id=SPT.RefId )))
from __SPTasvie__ SPT
where SPId in (select Id from __SPFactor__ where FactorType=1 and Id<>0 and FPId=@FPId) and FPId=@FPId
and RefType=2

update SPT set VDesc='فاکتور تسویه بابت ' + ltrim(rtrim(convert(char,(select FactorNo from __SPFactor__ where
Id=SPT.SPId and FPId=@FPId and FactorType=1)))) + ' دریافت '+ltrim(rtrim((select PNo,* from __PayRcv__
where Id=SPT.RefId)))
from __SPTasvie__ SPT
where SPId in (select Id from __SPFactor__ where FactorType=1 and Id<>0 and FPId=@FPId) and FPId=@FPId
and RefType=3

update RCV set PDesc='شماره فروش فاکتور تسویه بابت'+cast ((select factorNo from __SPFactor__ where
fpid=RCV.FPId and Id=(select spid from __SPTasvie__ where RefId=RCV.Id and RefType=3 and FPId=rcv.FPId))
as varchar)
from __PayRcv__ RCV where fpid=9 and id in (select RefId from __SPTasvie__ where fpid=9 and RefType=3) and
PayRcv=-1

update RCV set PDesc=dbo.Verify1256Text(PDesc)
from __PayRcv__ RCV where fpid=@FPId and id in (select RefId from __SPTasvie__ where fpid=@FPId and
RefType=3) and PayRcv=-1

update RCV set PDesc=dbo.ENumToFNum(PDesc)
from __PayRcv__ RCV where fpid=@FPId and id in (select RefId from __SPTasvie__ where fpid=@FPId and
RefType=3) and PayRcv=-1
در صورت عدم امکان بازسازی کاردکس میتوانید با الهام از اسکریپت زیر شرحهای آرتیکل های مرتبط را ---- بروز کنید
update A set ADesc= C.SPCDesc
from __Article__  A inner join __SPCost__ C on A.FPId=C.FPId and A.InvSPId=C.SPId and
A.AccountId=C.CostAccount and A.FAccId=C.CostFAccId
where InvSPId<>0 and InvSPId is not null and InvSPId in (select Id from __SPFactor__ where FactorType=1 and
Id<>0 and FPId=@FPId)
and A.FPId=@FPId

update AD  set ADesc= ltrim(rtrim(convert(char,(select FactorNo from __SPFactor__ where Id=AD.InvSPId and
FactorType=1 and FPId=@FPId))))
                        +' به فروش بابت '+(select top (1) Name from __Account__ where FullId=AD.AccountId
and FPId=@FPId)
                        +'-'+(select top (1)  CustomerName from __SPFactor__ where Id=AD.InvSPId)+' طي
شماره فاکتور '
from __Article__ AD
where invSPId<>0 and InvSPId in (select Id from __SPFactor__ where FactorType=1 and Id<>0 and FPId=@FPId)
and FPId=@FPId


update __Article__ set Adesc=dbo.ENumToFNum(Adesc)
```

```
where id<>0 and InvSPId<>0 and InvSPId in (select Id from __SPFactor__ where FactorType=1 and Id<>0 and
FPId=@FPId)and FPId=@FPId
```

مرتب سازی رسید و حواله انبار (Sort Strcptorder)

شماره گذاری رسید ها و حواله در سیستم تدبیر به صورت زیر صورت میگیرد:

مرتب سازی و شماره گذاری کلیه رسیدها و حواله ها براساس تاریخ و ساعت

```
declare @pFPId int
set @pFPId=7

update __StRcptOrder__
set RONo=cte.rononew

from __StRcptOrder__ inner join (select id,RONo,dbo.mtos(RODate) as rodate,RODate1,ROW_NUMBER()
over(order by rodate,RODate1,id ) as rononew from __StRcptOrder__
 where FPId=@pFPId  and ROType>100 ) as cte on cte.Id=__StRcptOrder__.Id
where __StRcptOrder__.FPId=@pFPId and __StRcptOrder__.ROType>100

update __StRcptOrder__
set RONo=cte.rononew
from __StRcptOrder__ inner join (select id,RONo,dbo.mtos(RODate) as rodate,RODate1,ROW_NUMBER()
over(order by rodate,RODate1,id ) as rononew from __StRcptOrder__
 where FPId=@pFPId  and ROType<100) as cte on cte.Id=__StRcptOrder__.Id
where __StRcptOrder__.FPId=@pFPId and __StRcptOrder__.ROType<100
```

مرتب سازی و شماره گذاری به تفکیک انبار براساس تاریخ و ساعت (شماره ها به ازای هر انبار از یک شروع می شود)

```
declare @pStock int,@pFPId int
set @pStock=1
set @pFPId=7

update __StRcptOrder__
set RONo=cte.rononew

from __StRcptOrder__ inner join (select id,RONo,dbo.mtos(RODate) as rodate,RODate1,ROW_NUMBER()
over(order by rodate,RODate1,id ) as rononew from __StRcptOrder__
 where FPId=@pFPId  and ROType>100 and StockRoomId=@pStock) as cte on cte.Id=__StRcptOrder__.Id
where __StRcptOrder__.FPId=@pFPId and __StRcptOrder__.ROType>100 and StockRoomId=@pStock

update __StRcptOrder__
set RONo=cte.rononew
from __StRcptOrder__ inner join (select id,RONo,dbo.mtos(RODate) as rodate,RODate1,ROW_NUMBER()
over(order by rodate,RODate1,id ) as rononew from __StRcptOrder__
 where FPId=@pFPId  and ROType<100 and StockRoomId=@pStock) as cte on cte.Id=__StRcptOrder__.Id
where __StRcptOrder__.FPId=@pFPId and __StRcptOrder__.ROType<100 and StockRoomId=@pStock
```

مرتب سازی و شماره گذاری به تفکیک عملیات براساس تاریخ و ساعت (شماره ها به ازای هر نوع فرم عملیاتی انبار از یک شروع می شود)

```
declare @pOPType int,@pFPId int
```

```
set @pOPType=1 ---(select Id from __StockIOType__)
set @pFPId=7

update __StRcptOrder__
set RONo=cte.rononew

from __StRcptOrder__ inner join (select id,RONo,dbo.mtos(RODate) as rodate,RODate1,ROW_NUMBER()
over(order by rodate,RODate1,id ) as rononew from __StRcptOrder__
 where FPId=@pFPId  and ROType>100 and ROType=@pOPType) as cte on cte.Id=__StRcptOrder__.Id
where __StRcptOrder__.FPId=@pFPId and __StRcptOrder__.ROType>100 and ROType=@pOPType

update __StRcptOrder__
set RONo=cte.rononew
from __StRcptOrder__ inner join (select id,RONo,dbo.mtos(RODate) as rodate,RODate1,ROW_NUMBER()
over(order by rodate,RODate1,id ) as rononew from __StRcptOrder__
 where FPId=@pFPId  and ROType<100 and ROType=@pOPType) as cte on cte.Id=__StRcptOrder__.Id
where __StRcptOrder__.FPId=@pFPId and __StRcptOrder__.ROType<100 and ROType=@pOPType
```

مرتب سازی به تفکیک انبار و عملیات

```
declare @pOPType int,@pFPId int,@pStock int
set @pOPType=1 ---(select Id from __StockIOType__)
set @pFPId=7
set @pStock=1

update __StRcptOrder__
set RONo=cte.rononew

from __StRcptOrder__ inner join (select id,RONo,dbo.mtos(RODate) as rodate,RODate1,ROW_NUMBER()
over(order by rodate,RODate1,id ) as rononew from __StRcptOrder__
 where FPId=@pFPId  and ROType>100 and ROType=@pOPType and StockRoomId=@pStock) as cte on
cte.Id=__StRcptOrder__.Id
where __StRcptOrder__.FPId=@pFPId and __StRcptOrder__.ROType>100 and ROType=@pOPType and
StockRoomId=@pStock

update __StRcptOrder__
set RONo=cte.rononew
from __StRcptOrder__ inner join (select id,RONo,dbo.mtos(RODate) as rodate,RODate1,ROW_NUMBER()
over(order by rodate,RODate1,id ) as rononew from __StRcptOrder__
 where FPId=@pFPId  and ROType<100 and ROType=@pOPType and StockRoomId=@pStock) as cte on
cte.Id=__StRcptOrder__.Id
where __StRcptOrder__.FPId=@pFPId and __StRcptOrder__.ROType<100 and ROType=@pOPType and
StockRoomId=@pStock
```

انجام نوع مرتب سازی براساس نیاز و درخواست کاربر خواهد بود و قبل از شروع فرآیند از کاربر پرسیده شود دقیقا به چه چیزی احتیاج دارد.

مرتب سازی فرم های ورود و خروج انبار(SortStockIO)

مرتب سازی فرم های ورود و خروج نیز می تواند به صورت کلی و یا به تفکیک انبار انجام شود. در مورد فرم های ریالی که ورود و خروج هم شامل می شود. میبایست بازسازی کاردکس انجام شود.

```
declare @pStock int,@pFPId int
set @pStock=1
set @pFPId=7

update __stockIo__ set STANo=cte.STANoN
from __stockIO__ inner join (select Id,STANo,dbo.MTOS(STADate) as STADate,ROW_NUMBER()
over(order by STADate,Id) as STANoN from __StockIO__
where FPId=@pFPId and STAType=104 and Stock=@pStock) as cte on cte.Id=__StockIO__.Id
where __StockIO__.FPId=@pFPId and __StockIO__.STAType=104 and stock=@pStock

update __stockIo__ set STANo=cte.STANoN
from __stockIO__ inner join (select Id,STANo,dbo.MTOS(STADate) as STADate,ROW_NUMBER()
over(order by STADate,Id) as STANoN from __StockIO__
where FPId=@pFPId and STAType=4 and Stock=@pStock) as cte on cte.Id=__StockIO__.Id
where __StockIO__.FPId=@pFPId and __StockIO__.STAType=4 and stock=@pStock
```

مرتب سازی فرم های انتقال بین انبار(SortStockTrans)

فرم های انتقال بیت انبار ریالی و مقداری در دو جدول متفاوت نگهداری می شود و پس از مرتب سازی انتقال بین انبار ریالی نیاز به بازسازی کاردکس است.

```
set @pFPId=16

update __stocktrans__ set STNo=cte.STNoN
from __stocktrans__ inner join (select Id,STNo,dbo.MTOS(STDate) as STADate,ROW_NUMBER()
over(order by STDate,Id) as STNoN from __stocktrans__
where FPId=@pFPId and STType=0) as cte on cte.Id=__stocktrans__.Id
where __stocktrans__.FPId=@pFPId and __stocktrans__.STType=0
```

جدول انتقال بین انبار ریالی __VStockTrans__ می باشد.

مرتب سازی فرم های رسید محصول ()

اطلاعات رسید محصول درهمان جدول انتقال بین انبار ها ذخیره شده است و به وسیله فیلد STType از هم تفکیک شده اند:

```
set @pFPId=16

update __stocktrans__ set STNo=cte.STNoN
from __stocktrans__ inner join (select Id,STNo,dbo.MTOS(STDate) as STADate,ROW_NUMBER()
over(order by STDate,Id) as STNoN from __stocktrans__
where FPId=@pFPId and STType=3) as cte on cte.Id=__stocktrans__.Id
where __stocktrans__.FPId=@pFPId and __stocktrans__.STType=3
```

اطلاعات رسید محصول های مقداری در جدول __VStockTrans__ ذخیره شده است.

بروزرسانی شرح آرتیکل اسناد مرتبط با سندهای انبار (SortStockArticleDesc)

در صورتی که به هر دلیلی پس از انجام عملیات سورت در فرم های انبار کاربر امکان بازسازی کاردکس نداشته باشد می تواند از اسکریپت زیر برای بروزرسانی شرح ها استفاده کنید:

```
declare @pFPId int =16

update ST set RODesc=(select STADesc from __StockIO__ where FPId=ST.FPId and id=ST.ROReference and ROType=ST.ROType)+' '
 + dbo.Verify1256Text( 'شماره ورود سند '+(select dbo.ENumToFNum(cast(STANo as varchar)) from __StockIO__ where FPId=ST.FPId and id=ST.ROReference and STAType=ST.ROType))
FROM __StRcptOrder__ ST
WHERE FPId=@pFPId AND Rotype=4 AND roreference<>-1

update ST set RODesc=(select STADesc from __StockIO__ where FPId=ST.FPId and id=ST.ROReference and ROType=ST.ROType)+' '
+dbo.Verify1256Text( 'شماره خروج سند '+(select dbo.ENumToFNum(cast(STANo as varchar)) from __StockIO__ where FPId=ST.FPId and id=ST.ROReference and STAType=ST.ROType))
FROM __StRcptOrder__ ST
WHERE FPId=@pFPId AND Rotype=104 AND roreference<>-1

update ST SET RODesc=dbo.Verify1256Text( 'شماره انتقال سند ـدیگر انبار از انتقال'  '+(select dbo.ENumToFNum(cast(STNo as varchar)) from __StockTrans__ where FPId=ST.FPId and id=ST.ROReference and STType=0))
FROM __StRcptOrder__ ST
WHERE FPId=@pFPId AND Rotype=3 AND roreference<>-1

update ST SET RODesc=dbo.Verify1256Text( 'شماره انتقال سند ـدیگر انبار به انتقال'  '+(select dbo.ENumToFNum(cast(STNo as varchar)) from __StockTrans__ where FPId=ST.FPId and id=ST.ROReference and STType=0))
FROM __StRcptOrder__ ST
WHERE FPId=@pFPId AND Rotype=103 AND roreference<>-1

update A set ADesc=S.STADesc +' '+  dbo.Verify1256Text( 'شماره ورود سند '+dbo.ENumToFNum(cast(STANo as varchar)))
FROM __Article__ A inner join __StockIO__ S ON A.FPId=S.FPId AND A.InvSTAId=S.Id
WHERE A.FPId=@pFPId AND A.InvSTAId<>0 AND S.STAType=4

update A set ADesc=S.STADesc +' '+  dbo.Verify1256Text( 'شماره خروج سند '+dbo.ENumToFNum(cast(STANo as varchar)))
FROM __Article__ A inner join __StockIO__ S ON A.FPId=S.FPId AND A.InvSTAId=S.Id
WHERE A.FPId=@pFPId AND A.InvSTAId<>0 AND S.STAType=104

update A set Adesc=STDesc+' '+dbo.Verify1256Text( 'شماره انبار بین انتقال '+dbo.ENumToFNum(cast(STNo as varchar)))
FROM __Article__ A inner join __StockTrans__ S ON A.FPId=S.FPId AND A.InvSTId=S.Id
WHERE S.STType=0 AND  A.FPId=@pFPId AND A.InvSTId<>0 AND S.STType=0

update ST SET RODesc=dbo.Verify1256Text( ' شماره مقداری انتقال سند ـدیگر انبار از انتقال '+(select dbo.ENumToFNum(cast(VSTNo as varchar)) from __VStockTrans__ where FPId=ST.FPId and Id=VST.VSTId and VSTType=0))
FROM __StRcptOrder__ ST inner join __VSTRO__ VST ON ST.Id=VST.ROId AND ST.FPId=VST.FPId
WHERE ST.FPId=@pFPId AND Rotype=3 AND roreference=-1
```

```
update ST SET RODesc=dbo.Verify1256Text( 'شماره مقداری انتقال سند ـدیگر انبار به انتقال '+(select
dbo.ENumToFNum(cast(VSTNo as varchar)) from __VStockTrans__ where FPId=ST.FPId and Id=VST.VSTId and
VSTType=0))
FROM __StRcptOrder__ ST inner join __VSTRO__ VST ON ST.Id=VST.ROId AND ST.FPId=VST.FPId
WHERE ST.FPId=@pFPId AND Rotype=103 AND roreference=-1
```