Procedury

Zmiana edycji silnika i operacje na bazach aplikacji.

- Zmiana edycji silnika MSSQL
- Wyciąganie pliku JPK w XML z bazy Optimy
- Wciągnięcie modyfikacji danych w księgach rozliczeniowych Optima

Zmiana edycji silnika MSSQL

Typ: procedura **Dotyczy: SQL Server 2016–2022, instalacja z nośnika instalacyjnego**

Czas: ~40 min

Efekt: instancja działa pod nową edycją licencyjną silnika (np. Enterprise zamiast Evaluation, albo Standard zamiast Enterprise), z zachowaniem wszystkich baz i konfiguracji.

Skróty użyte na tej stronie (1)

Skrót	Co znaczy
SQL	Structured Query Language: język zapytań do relacyjnych baz danych

Zanim zaczniemy

Z ŻYCIA WZIĘTE

Pan Zbigniew wraca z zarządu z pytaniem, czy da się zaoszczędzić na licencji, skoro sql01 stoi akurat na edycji Enterprise, a Always On i tak nie jest używany. Sprawdzamy, czy którakolwiek baza korzysta z funkcji dostępnych tylko w Enterprise, i trafiamy na jedną tabelę w CDN_Zielona podzieloną na partycje sprzed lat. Zanim zaczniemy downgrade do Standard, scalamy partycjonowanie tej tabeli, żeby instalator nie zablokował operacji regułą Cluster_EditionDownGradeCheck. Po zmianie edycji i restarcie usługi wszystkie bazy wracają online, a firma płaci za licencję odpowiednią do realnego użycia.

Wymaganie	Jak sprawdzić
Aktualna edycja i wersja silnika	`SELECT SERVERPROPERTY('Edition'), SERVERPROPERTY('ProductVersion');` — opisane na stronie Rozpoznanie instancji i baz
Pliki instalacyjne docelowej edycji dostępne lokalnie na serwerze	katalog z `setup.exe` odpowiedni dla nowej edycji
Pełny backup wszystkich baz na instancji	opisane na stronie Backup baz danych
Okno serwisowe — zmiana edycji wymaga restartu usługi silnika	potwierdzenie z właścicielem aplikacji

Uważaj: zmiana edycji z wyższej na niższą (downgrade, np. z Enterprise na Standard) jest operacją, która może zakończyć się niepowodzeniem, jeśli instancja korzysta z funkcji dostępnych wyłącznie w wyższej edycji — partycjonowanie tabel, kompresja danych na poziomie wierszy/stron w starszych wersjach silnika, `ONLINE` rebuild indeksów, Always On z więcej niż jedną repliką drugorzędną. Funkcje niedostępne w docelowej edycji trzeba wyłączyć albo usunąć przed zmianą, inaczej instalator odmówi kontynuacji.

Przebieg

1. Sprawdzenie obecnej edycji i wersji

```
SELECT
    SERVERPROPERTY('Edition') AS Edycja,
    SERVERPROPERTY('ProductVersion') AS WersjaProduktu,
    SERVERPROPERTY('ProductLevel') AS PoziomProduktu,
    SERVERPROPERTY('EngineEdition') AS EngineEdition;
```

```
Edycja                        WersjaProduktu   PoziomProduktu   EngineEdition
----------------------------- ---------------- ---------------- -------------
Enterprise Edition (64-bit)    15.0.4223.1      RTM              3


(1 rows affected)
```

Zapamiętujemy `EngineEdition` i pełną wersję — po zmianie porównujemy je z wynikiem uzyskanym po restarcie usługi.

2. Sprawdzenie użycia funkcji specyficznych dla edycji

Jeżeli planowany jest downgrade, sprawdzamy, czy którakolwiek baza na instancji korzysta z funkcji niedostępnych w docelowej edycji, na przykład partycjonowania tabel:

```
SELECT DB_NAME(database_id) AS NazwaBazy, OBJECT_NAME(object_id, database_id) AS NazwaTabeli
FROM sys.dm_db_partition_stats
WHERE partition_number > 1
GROUP BY database_id, object_id;
```

3. Uruchomienie zmiany edycji z wiersza poleceń

Z katalogu z plikami instalacyjnymi docelowej edycji, w konsoli uruchomionej jako administrator:

```
setup.exe /ACTION=EditionUpgrade /INSTANCENAME=NazwaInstancji /IACCEPTSQLSERVERLICENSETERMS
```

Dla downgrade'u, kiedy standardowe sprawdzenie reguł instalatora blokuje operację ze względu na wykryte różnice licencyjne, pomijamy tę konkretną regułę:

```
setup.exe /ACTION=EditionUpgrade /SkipRules=Cluster_EditionDownGradeCheck
/IACCEPTSQLSERVERLICENSETERMS
```

> (gdzie `Cluster_EditionDownGradeCheck` to nazwa reguły weryfikującej zgodność edycji przy downgrade w konfiguracji klastrowej — instalator opisuje w logu, która dokładnie reguła zablokowała instalację, jeśli innych reguł dotyczy problem)

Instalator poprosi o klucz licencyjny albo o wybór wariantu bezpłatnego (Express, Developer, Evaluation), w zależności od docelowej edycji.

4. Restart usługi silnika

Instalator restartuje usługę automatycznie po zakończeniu operacji. Jeśli tego nie zrobił, restartujemy ręcznie:

```
Restart-Service -Name "MSSQL`$NazwaInstancji"
```

Weryfikacja

```
SELECT
    SERVERPROPERTY('Edition') AS Edycja,
    SERVERPROPERTY('EngineEdition') AS EngineEdition;
```

```
Edycja                    EngineEdition
------------------------- -------------
Standard Edition (64-bit)   2


(1 rows affected)
```

Sprawdzamy też, że wszystkie bazy wróciły do stanu `ONLINE` po restarcie usługi:

```
SELECT name, state_desc FROM sys.databases WHERE state_desc <> 'ONLINE';
```

Pusty wynik oznacza, że wszystkie bazy uruchomiły się poprawnie.

Wycofanie zmian

Powrót do poprzedniej edycji wymaga powtórzenia tej samej procedury z plikami instalacyjnymi poprzedniej edycji — sama zmiana edycji nie jest jednym przełącznikiem do cofnięcia, tylko ponowną operacją `EditionUpgrade` w drugą stronę. Jeśli poprzednia edycja była wyższa (np. wracamy z Standard do Enterprise po nieudanym downgrade), operacja przebiega bez dodatkowych ograniczeń, bo idziemy w górę, a nie w dół.

Gdy coś nie zadziała

Objaw	Przyczyna	Naprawa
Instalator blokuje downgrade regułą `Cluster_EditionDownGradeCheck`	docelowa edycja nie obsługuje funkcji wykrytych na instancji	`/SkipRules=Cluster_EditionDownGradeCheck`, o ile funkcja została świadomie wyłączona wcześniej
Usługa nie startuje po zmianie edycji	niekompletna instalacja albo brak wymaganego klucza licencyjnego	log instalatora w `%ProgramFiles%\Microsoft SQL Server\...\Setup Bootstrap\Log`, ponowne uruchomienie `setup.exe`
Baza korzystająca z partycjonowania nie chodzi po downgrade do edycji niższej niż Standard	partycjonowanie tabel wymaga Enterprise (lub Standard od SQL Server 2016 SP1 z ograniczeniami)	scalenie partycji przed downgrade albo pozostanie przy wyższej edycji
`EngineEdition` po operacji nadal pokazuje starą wartość	usługa nie została zrestartowana po zakończeniu instalatora	ręczny restart usługi silnika

Powiązane: Rozpoznanie instancji i baz · Architektura MSSQL: instancja, baza, pliki MDF/NDF/LDF · Backup baz danych

Źródła: dokumentacja Microsoft Learn, sekcja „Upgrade to a different edition of SQL Server"

Wyciąganie pliku JPK w XML z bazy Optimy

Poziom: średni **Dotyczy: SQL Server 2016-2022, baza Comarch ERP Optima (CDN)**

Czas: ~15 min

Wyciągnięcie pliku JPK zapisanego jako plik binarny wewnątrz bazy Optimy i zapisanie go z powrotem jako plik XML na dysku.

Skróty użyte na tej stronie (6)

Skrót	Co znaczy
CDN	Content Delivery Network: sieć serwerów Red Hat, z której zarejestrowany system pobiera pakiety i aktualizacje
ERP	Enterprise Resource Planning: system planowania zasobów przedsiębiorstwa, łączący księgowość, magazyn i inne procesy firmy w jednej aplikacji
JPK	Jednolity Plik Kontrolny: ustandaryzowany plik XML, którym firma raportuje dane księgowe i podatkowe do urzędu skarbowego
SQL	Structured Query Language: język zapytań do relacyjnych baz danych
SSMS	SQL Server Management Studio: graficzna konsola MSSQL
XML	eXtensible Markup Language: format tekstowy do zapisu ustrukturyzowanych danych w znacznikach, tutaj używany przez PowerShell do zapisu zrzutu obiektów poleceniem Export-Clixml

Fiszka

TYLKO ODCZYT

Sprawdzenie struktury tabeli z plikami JPK

```
SELECT TOP (10) [JPK_JPKID], [JPK_KodOsoby], [JPK_Typ], [JPK_Nazwa],
    [JPK_DataUtworzenia], [JPK_CelZlozenia], [JPK_WariantFormularza],
```

```
    [JPK_XML], [JPK_RefNr], [JPK_Path], [JPK_Status], [JPK_StatusCode],
    [JPK_StatusOpis], [JPK_KodUrzedu], [JPK_DataWyslania], [JPK_DataWplyniecia],
    [JPK_KodOsobyOdbierajacej], [JPK_DataOdebrania], [JPK_SkrotDokumentu],
    [JPK_SkrotStruktury], [JPK_StrukturaLogiczna], [JPK_StempelCzasu],
    [JPK_SigningTime], [JPK_NazwaPodmiotu], [JPK_NIP1], [JPK_OkresOd], [JPK_OkresDo]
FROM [NazwaBazy].[CDN].[PlikiJPK];
```

Sprawdzamy zawartość tabeli przechowującej pliki JPK w bazie Optimy, zanim wyciągniemy z niej konkretny plik — kolumna `JPK_Nazwa` identyfikuje plik, a `JPK_XML` trzyma jego binarną zawartość. Schemat `CDN` jest stałym schematem bazy Optimy niezależnie od nazwy samej bazy danych.

Przykład:

```
JPK_JPKID   JPK_Typ   JPK_Nazwa                    JPK_DataUtworzenia        JPK_Status
----------- --------- ---------------------------- ------------------------
-----------
104821      JPK_VAT   JPK_VAT_202608_CDN_Firma      2026-09-20 10:12:03.000   Wyslany

(1 rows affected)
```

JPK = Jednolity Plik Kontrolny: ustandaryzowany plik XML, którym firma raportuje dane księgowe i podatkowe do urzędu skarbowego · **CDN** = Content Delivery Network: sieć serwerów Red Hat, z której zarejestrowany system pobiera pakiety i aktualizacje

TYLKO ODCZYT

Weryfikacja, które kolumny zawierają pliki binarne

```
SELECT t.TABLE_NAME, c.COLUMN_NAME, c.DATA_TYPE
FROM INFORMATION_SCHEMA.COLUMNS c
JOIN INFORMATION_SCHEMA.TABLES t ON t.TABLE_NAME = c.TABLE_NAME
WHERE c.DATA_TYPE IN ('varbinary', 'image', 'xml')
    AND (t.TABLE_NAME LIKE '%JPK%' OR c.COLUMN_NAME LIKE '%JPK%');
```

Sprawdzamy, w jakim typie danych trzymana jest zawartość pliku — w tabeli `PlikiJPK` kolumna `JPK_XML` jest typu `varbinary`, mimo nazwy sugerującej XML wprost, dlatego samego `SELECT JPK_XML` nie da się po prostu obejrzeć w SSMS jako tekst ani zapisać bezpośrednio jako plik.

Przykład:

```
TABLE_NAME   COLUMN_NAME   DATA_TYPE
------------ ------------ -----------
PlikiJPK     JPK_XML       varbinary

(1 rows affected)
```

XML = eXtensible Markup Language: format tekstowy do zapisu ustrukturyzowanych danych w znacznikach, tutaj używany przez PowerShell do zapisu zrzutu obiektów poleceniem Export-Clixml · **SSMS** = SQL Server Management Studio: graficzna konsola MSSQL · **join** = połącz dwa pliki tekstowe po wspólnej kolumnie, jak złączenie tabel w bazie danych · **JPK** = Jednolity Plik Kontrolny: ustandaryzowany plik XML, którym firma raportuje dane księgowe i podatkowe do urzędu skarbowego

ZMIENIA SYSTEM

Wyciągnięcie pliku JPK do XML na dysku: skrypt PowerShell

```
$server = "NazwaSerwera"
$db     = "NazwaBazy"
$nazwa  = "NazwaPlikuDocelowego"
$out    = "C:\SciezkaDocelowa\$nazwa.xml"
$tabela = "[$db].[CDN].[PlikiJPK]"

$conn = New-Object
System.Data.SqlClient.SqlConnection("Server=$server;Database=$db;Integrated
Security=True")
$conn.Open()
$cmd = $conn.CreateCommand()
$cmd.CommandText = "SELECT JPK_XML FROM $tabela WHERE JPK_Nazwa = @n"
$null = $cmd.Parameters.AddWithValue("@n", $nazwa)
$bytes = [byte[]]$cmd.ExecuteScalar()
$conn.Close()

[IO.File]::WriteAllBytes($out, $bytes)
Write-Host "Zapisano $($bytes.Length) bajtów do $out"
```

Uruchamiamy w PowerShell z konta, które ma połączenie autentykacją Windows do bazy danych. Skrypt odczytuje zawartość binarną kolumny `JPK_XML` dla wskazanego pliku po jego nazwie (`JPK_Nazwa`) i zapisuje ją jako plik `.xml` na dysku, dokładnie w takiej postaci, w jakiej trafiła kiedyś do bazy przy wysyłce deklaracji.

(gdzie `NazwaPlikuDocelowego` to wartość z kolumny `JPK_Nazwa` znaleziona w pierwszym zapytaniu Fiszki — musi się zgadzać dokładnie, łącznie z wielkością liter)

Przykład:

```
Zapisano 184320 bajtów do C:\SciezkaDocelowa\JPK_VAT_202608_CDN_Firma.xml
```

JPK = Jednolity Plik Kontrolny: ustandaryzowany plik XML, którym firma raportuje dane księgowe i podatkowe do urzędu skarbowego · **XML** = eXtensible Markup Language: format tekstowy do zapisu ustrukturyzowanych danych w znacznikach, tutaj używany przez PowerShell do zapisu zrzutu obiektów poleceniem Export-Clixml · **CDN** = Content Delivery Network: sieć serwerów Red Hat, z której zarejestrowany system pobiera pakiety i aktualizacje

Z życia wzięte

Z ŻYCIA WZIĘTE

Pan Wojtek podczas rocznego audytu prosi o kopię pliku JPK_VAT wysłanego do urzędu skarbowego pół roku wcześniej, bo w jego papierach brakuje jednej strony. Pani Halina nie pamięta, żeby Optima trzymała gdzieś osobno taki plik do pobrania, więc sprawdzamy bazę CDN_Zielona bezpośrednio w tabeli PlikiJPK. Kolumna JPK_XML okazuje się typu varbinary, więc zwykłe zaznaczenie w SSMS pokazuje tylko ciąg szesnastkowy zamiast czytelnego XML-a. Skrypt PowerShell zapisujący zawartość na dysk jako plik .xml rozwiązuje sprawę w dwie minuty, zanim pan Wojtek zdąży dopić kawę.

Po co to istnieje

Pojęcie: JPK. Jednolity Plik Kontrolny — ustandaryzowany format XML, w którym przedsiębiorstwa raportują dane księgowe i podatkowe do administracji skarbowej. System Comarch ERP Optima generuje pliki JPK i zapisuje ich pełną treść binarnie wewnątrz własnej bazy danych, w schemacie `CDN`, obok innych danych księgowych.

Do wyciągnięcia pliku sięgamy typowo wtedy, gdy trzeba dostarczyć jego kopię do urzędu poza standardowym mechanizmem wysyłki wbudowanym w aplikację Optima, albo gdy trzeba zweryfikować dokładną zawartość wysłanej wcześniej deklaracji bez logowania się do samej

aplikacji.

Częste błędy

Próba odczytania `JPK_XML` bezpośrednio jako tekstu w SSMS. Kolumna jest typu `varbinary`, więc bezpośredni podgląd w SSMS pokazuje ciąg szesnastkowy zamiast czytelnego XML-a — trzeba zapisać zawartość binarną do pliku, tak jak w skrypcie powyżej, i dopiero otworzyć plik `.xml` w edytorze tekstu.

Pomylenie nazwy pliku z identyfikatorem `JPK_JPKID`. Zapytanie w skrypcie filtruje po kolumnie `JPK_Nazwa`, nie po numerycznym identyfikatorze — podanie samego ID zamiast pełnej nazwy pliku zwraca pusty wynik i `$bytes` o długości zero.

Powiązane: Rozpoznanie instancji i baz · Wciągnięcie modyfikacji danych w księgach rozliczeniowych Optima

Źródła: dokumentacja Comarch ERP Optima, moduł JPK

Wciągnięcie modyfikacji danych w księgach rozliczeniowych Optima

Poziom: średni **Dotyczy: SQL Server 2016-2022, baza Comarch ERP Optima (CDN)**

Czas: ~20 min

Wyszukanie zapisów księgowych z zamkniętego okresu rozliczeniowego, które zostały mimo to utworzone albo zmodyfikowane później, po jego formalnym zamknięciu.

Skróty użyte na tej stronie (3)

Skrót	Co znaczy
CDN	Content Delivery Network: sieć serwerów Red Hat, z której zarejestrowany system pobiera pakiety i aktualizacje
ERP	Enterprise Resource Planning: system planowania zasobów przedsiębiorstwa, łączący księgowość, magazyn i inne procesy firmy w jednej aplikacji
SQL	Structured Query Language: język zapytań do relacyjnych baz danych

Fiszka

OSTROŻNIE · NIEODWRACALNE

Wyszukanie ruszonych zapisów ze starego okresu księgowego

```
DECLARE @Od      date = '2026-01-01'; -- okno akcji: początek roku
DECLARE @Do      date = '2026-07-08'; -- koniec półotwarty = do 7 lipca włącznie
DECLARE @StareOd date = '2025-01-01'; -- "stare księgi": rok 2025
DECLARE @StareDo date = '2026-01-01';
```

```
IF OBJECT_ID('tempdb..#baza') IS NOT NULL DROP TABLE #baza;

SELECT
    n.DeN_DeNId,
    n.DeN_Dziennik,
    n.DeN_NrDziennika,
    n.DeN_DataDok,
    n.DeN_Bufor,
    n.DeN_TS_Zal,
    n.DeN_TS_Mod,
    CASE WHEN n.DeN_TS_Zal >= @Od AND n.DeN_TS_Zal < @Do
         THEN 'Nowy wpis z datą starego okresu'
         ELSE 'Modyfikacja starego wpisu' END AS TypZmiany,
    CASE WHEN n.DeN_TS_Zal >= @Od AND n.DeN_TS_Zal < @Do
         THEN n.DeN_TS_Zal ELSE n.DeN_TS_Mod END AS AkcjaTS
INTO #baza
FROM CDN.DekretyNag n
WHERE n.DeN_DataDok >= @StareOd AND n.DeN_DataDok < @StareDo
    AND ( (n.DeN_TS_Zal >= @Od AND n.DeN_TS_Zal < @Do)
       OR (n.DeN_TS_Mod >= @Od AND n.DeN_TS_Mod < @Do) );
```

Wyszukujemy nagłówki dekretów księgowych (`CDN.DekretyNag`), które dotyczą dokumentów z okresu uznanego już za zamknięty (`@StareOd`–`@StareDo`), ale zostały utworzone albo zmodyfikowane w oknie czasowym, które nas interesuje (`@Od`–`@Do`). Kolumna `DeN_TS_Zal` trzyma znacznik czasu założenia zapisu, `DeN_TS_Mod` — znacznik ostatniej modyfikacji; rozróżnienie w kolumnie `TypZmiany` mówi, czy mamy do czynienia z całkiem nowym wpisem dopisanym do starego okresu, czy z edycją wpisu, który już wcześniej istniał.

Zbieramy wynik do tabeli tymczasowej `#baza`, żeby dwa kolejne zapytania — listę szczegółową i podsumowanie — mogły z niego korzystać bez powtarzania tej samej, złożonej logiki filtrowania.

□□ **CDN** = Content Delivery Network: sieć serwerów Red Hat, z której zarejestrowany system pobiera pakiety i aktualizacje · **declare** = zadeklaruj zmienną powłoki z określonym typem, na przykład jako liczbę całkowitą albo tablicę

□□ **TYLKO ODCZYT**

Lista szczegółowa zmienionych zapisów

```
SELECT
    b.DeN_Dziennik,
    b.DeN_NrDziennika,
    b.DeN_DataDok AS DataKsiegowania,
    b.AkcjaTS AS KiedyRuszone,
    b.TypZmiany,
    b.DeN_Bufor,
    e.DeE_KontoWn,
    e.DeE_KontoMa,
    e.DeE_Kwota,
    e.DeE_Kategoria,
    e.DeE_Dokument
FROM #baza b
JOIN CDN.DekretyElem e ON e.DeE_DeNId = b.DeN_DeNId
ORDER BY b.AkcjaTS, b.DeN_Dziennik, b.DeN_NrDziennika, e.DeE_Lp;
```

Rozwijamy każdy znaleziony nagłówek dekretu do jego pozycji księgowych (`CDN.DekretyElem`) — konta Winien/Ma, kwotę, kategorię i numer dokumentu źródłowego. Ten widok trafia zwykle do księgowości jako lista do ręcznej weryfikacji, które konkretnie zapisy wymagają wyjaśnienia.

Przykład:

```
DeN_Dziennik   DeN_NrDziennika   DataKsiegowania   KiedyRuszone
TypZmiany                          DeE_KontoWn   DeE_KontoMa   DeE_Kwota
-------------- ----------------- ----------------- -------------------------
---------------------------------- ------------ ------------ -----------
KA             184               2025-11-14        2026-02-03 09:12:00.000   Modyfikacja
starego wpisu          201-01        702-01        4520.00
KA             912               2025-12-29        2026-01-08 16:44:12.000   Nowy wpis z
datą starego okresu    100-01        201-01        1200.00


(2 rows affected)
```

CDN = Content Delivery Network: sieć serwerów Red Hat, z której zarejestrowany system pobiera pakiety i aktualizacje · **join** = połącz dwa pliki tekstowe po wspólnej kolumnie, jak złączenie tabel w bazie danych

OSTROŻNIE · NIEODWRACALNE

Podsumowanie miesięczne

```
SELECT
    DATEFROMPARTS(YEAR(b.AkcjaTS), MONTH(b.AkcjaTS), 1) AS MiesiacAkcji,
    b.TypZmiany,
    COUNT(*) AS LiczbaDokumentow,
    SUM(CASE WHEN b.DeN_Bufor = 1 THEN 1 ELSE 0 END) AS WBuforze
FROM #baza b
GROUP BY DATEFROMPARTS(YEAR(b.AkcjaTS), MONTH(b.AkcjaTS), 1), b.TypZmiany
ORDER BY MiesiacAkcji, b.TypZmiany;

DROP TABLE #baza;
```

Agregujemy znalezione zapisy do jednej liczby na miesiąc i typ zmiany — pierwszy widok, jaki pokazujemy przy pytaniu „ile w ogóle było takich ruchów i w którym miesiącu było ich najwięcej", zanim ktokolwiek zacznie przeglądać listę szczegółową wiersz po wierszu. Na końcu sprzątamy po sobie tabelę tymczasową.

Przykład:

```
MiesiacAkcji   TypZmiany                          LiczbaDokumentow   WBuforze
-------------- ---------------------------------- ------------------ ----------
2026-01-01     Nowy wpis z datą starego okresu     14                 2
2026-02-01     Modyfikacja starego wpisu           6                  0
2026-03-01     Modyfikacja starego wpisu           3                  1

(3 rows affected)
```

Z życia wzięte

□□ Z ŻYCIA WZIĘTE

Po zamknięciu roku obrachunkowego pan Wojtek pyta, czy ktoś w księgowości poprawiał zapisy z zamkniętego już okresu, bo w jego zestawieniu jedna kwota się nie zgadza. Uruchamiamy zapytanie po tabelach DekretyNag i DekretyElem, filtrując dokumenty z zeszłego roku zmodyfikowane już po dacie zamknięcia. Wynik pokazuje jeden dekret z grudnia, poprawiony w lutym przez panią Halinę, wciąż leżący w buforze jako niezatwierdzony. Ten jeden wiersz wystarcza, żeby wyjaśnić rozbieżność, zanim audytor

> zdąży napisać zastrzeżenie do raportu.

Po co to istnieje

> **Pojęcie: dekret księgowy.** W terminologii Optimy pojedynczy zapis w dzienniku księgowym — nagłówek (`DekretyNag`) reprezentuje jedną operację księgową, a jej pozycje (`DekretyElem`) to poszczególne zapisy na kontach Winien/Ma, które razem muszą się bilansować. Rozróżnienie nagłówka od pozycji jest konieczne przy każdym zapytaniu analitycznym po księgach — sam nagłówek nie mówi nic o kwotach ani kontach, dopiero złączenie z pozycjami daje pełny obraz operacji.

Zapytanie z tej strony odpowiada na pytanie zadawane zwykle przez księgowość albo audyt po zamknięciu okresu rozliczeniowego: czy ktoś, po zamknięciu ksiąg za dany rok, wprowadził do nich jeszcze jakieś zmiany — czy to nowy wpis z datą z zamkniętego okresu, czy modyfikację wpisu, który istniał już wcześniej. `DeN_Bufor` mówi dodatkowo, czy taki wpis wciąż siedzi w buforze (jeszcze niezaksięgowany na stałe) czy jest już zatwierdzony.

Częste błędy

Filtrowanie tylko po `DeN_TS_Zal`, z pominięciem `DeN_TS_Mod`. Pomija to modyfikacje istniejących wcześniej zapisów — dokument założony rok wcześniej, ale edytowany w interesującym nas oknie czasowym, nie pojawi się na liście, jeśli sprawdzimy tylko datę założenia.

Pozostawienie tabeli tymczasowej `#baza` bez sprzątania w skrypcie uruchamianym wielokrotnie w tej samej sesji. Kolejne uruchomienie bez wcześniejszego `DROP TABLE` kończy się błędem `Msg 2714, Level 16, State 6, There is already an object named '#baza' in the database` — stąd warunkowy `DROP` na początku i jawny na końcu skryptu.

Powiązane: Wyciąganie pliku JPK w XML z bazy Optimy · Sesje, zapytania i blokady

Źródła: dokumentacja Comarch ERP Optima, struktura bazy danych modułu Księga Handlowa