DBS_UPDATE

Akcia DB_UPDATE a DBS_UPDATE

Funkcia

Modifikácia existujúceho riadku (riadkov) v databáze.

Deklarácia

```
 DB_UPDATE handleIdent_Int, rowIdent, retCodeIdent_Int [WHERE
strExpression_Str [BINDIN varIdent1, varIdent2, ... ]] [ORAHINT
hintIdent_Str]

 DB_UPDATE handleIdent_Int, rowIdent, retCodeIdent_Int [WHERE
strExpression_Str [BINDIN structRowIdent]] [ORAHINT hintIdent_Str]

 DBS_UPDATE dbObjIdent, rowIdent, retCodeIdent_Int [WHERE
strExpression_Str [BINDIN varIdent1, varIdent2, ... ]] [TRANS
transHandle_Int] [ORAHINT hintIdent_Str]

 DBS_UPDATE dbObjIdent, rowIdent, retCodeIdent_Int [WHERE
strExpression_Str [BINDIN structRowIdent]] [TRANS transHandle_Int]
[ORAHINT hintIdent_Str]
```

alebo

```
DB_UPDATE handleIdent_Int, structIdent, retCodeIdent_Int [ORAHINT
hintIdent_Str]

DBS_UPDATE dbObjIdent, structIdent, retCodeIdent_Int [TRANS
transHandle_Int] [ORAHINT hintIdent_Str]
```

Parametre

handleIdent_Int	in	Identifikátor typu *Int - (handle) spojenia s databázou (DB_CONNECT).*
dbObjIdent	in	Odkaz na objekt typu Tabuka.
rowIdent	in	Identifikátor jedného riadku štruktúry (modifikovaný riadok).
structIdent	in	Identifikátor celej štruktúry (modifikované riadky).
retCodeIdent_Int	out	Návratová hodnota typu *Int* - úspešnos akcie.
strExpression_Str	in	Výraz typu *String*, ktorý identifikuje modifikované riadky. Ak je výraz parametrizovaný, tak kúové slovo **BINDIN** a hodnoty parametrov (*structRowIdent* alebo *varIdent1*, *varIdent2* , ...) sú povinné.
varIdent1, varIdent2, ...	in	zoznam objektov, konštánt alebo lokálnych premenných, ktoré špecifikujú hodnoty parametrov parametrizovaného SQL výrazu *strExpression_Str*.
structRowIdent	in	Odkaz na riadok lokálnej premennej typu *Record* alebo na riadok štruktúrovanej premennej. Hodnoty riadkov špecifikujú hodnoty parametrov parametrizovaného SQL výrazu *strExpression_Str*.
transHandle_Int	in	Identifikátor Pripojenia na databázu.
hintIdent_Str	in	Výraz typu *String*, ktorý definuje Oracle SQL hint. Ten je použitý ako inštrukcia pre optimalizátor výkonu SQL príkazu. Zapisuje hodnotu bez obaujúcich znakov */*+ <*orahint* > **/*. Príklad použitia je uvedený tu.

Návratový kód

Hodnota parametra *retCodeIdent_Int* - pozri tabuku chybových stavov. Je možné získa rozšírenú informáciu o chybe.

Popis

Databáza musí by otvorená s prístupom *_DB_MODIFY*. Podobne ako pri akcii DB_READ (DBS_READ), môžu by modifikované riadky urené dvoma spôsobmi:

1. Nastavením kúových položiek v riadku štruktúry (`rowIdent`), resp. v celej štruktúre (`structIdent`). Nastavenie musí by realizované pred akciou **DB_UPDATE**. Položky musia by oznaené ako kú pri definícii objektu typu *Tabuka*. Potom sa v databáze hadá riadok, ktorý má zhodnú hodnotu pre kúové položky a tento je modifikovaný. Vkladaný riadok alebo štruktúra musia ma korektný typ štruktúry. Hodnoty všetkých kúových položiek v `rowIdent` musia by platné, inak sa modifikovanie neuskutoní, skoní s chybou a %GetLastExtErrorCode() vráti chybu 667.
2. Hodnotou výrazu `strExpression_Str`, ktorý predstavuje klauzulu **WHERE** pre SQL príkaz **UPDATE**, ktorý vykonáva zmenu databázy. V tomto prípade je hodnota kúovej položky v štruktúre nepoužitá.

Pokia nie je zadaný (alebo je prázdny) výraz `strExpression_Str` ani nie je zadefinovaný kú pri definícii objektu typu *Tabuka*, tak akcia vráti chybu _ERR_TRANS_ERROR (11) a premenná _ERR_NR_TRANS_EX bude ma hodnotu BAD_CONDITION.

Pozn: odporúame používa verziu bez výrazu `strExpression_Str` - Oracle pri použití výrazu krátkodobo zamkne aktualizovanú tabuku ako aj všetky tabuky, do ktorých má aktualizovaná tabuka *foreign index*. Toto môže spôsobi zablokovanie, ak iná prebiehajúca transakcia aktualizovala nejaké riadky v týchto tabukách. Zablokovanie potrvá až kým nebude prebiehajúca transakcia ukonená.

Výhodou akcie **DBS_UPDATE** pri práci s tabukou je možnos vynecha jej otvorenie a zatvorenie (jednoduchší zápis).

Pre D2000 verziu 5: Nevýhoda akcie **DBS_UPDATE** je v rýchlosti. Každé volanie akcie **DBS_UPDATE** má za následok, v rámci jej obsluhy v procese D2000 DBManager, nutnos databázu otvori a následne zatvori, o môže by asovo náronáinnos a predstavuje z databázového hadiska pomerne neštandardný postup. Nutnos otvárania a zatvárania databázy je možné potlai v rámci transakného spracovania tak, že za príkaz uvedieme parameter **TRANS**.
Pre D2000 verziu 6 a vyššie: Optimalizácia procesu D2000 DBManager (recyklovanie spojení, predpripravené spojenia) spôsobila, že akcia **DBS_UPDATE** sa vykoná rovnako rýchlo ako akcia **DB_UPDATE** a navyše sa ušetrí as potrebný na spracovanie explicitného príkazu na otvorenie databázy **DB_CONNECT**.

Potreba otvorenia a zatvorenia databázy sa môže eliminova v rámci transakného spracovania tak, že za príkazom nasleduje parameter **TRANS.**

Príklad

Príklad práce s databázou (akcie DB_...).

Súvisiace odkazy

DB_CONNECT
DB_DELETE
DB_DISCONNECT
DB_INSERT
DB_INSUPD
DB_READ
DB_READ_BLOB
DB_UPDATE_BLOB

DB_TRANS_OPEN
DB_TRANS_COMMIT
DB_TRANS_ROLLBACK
DB_TRANS_CLOSE

Všetky databázové akcie

Súvisiace stránky:

Akcie v skriptoch