Modernisation IBM i – Nouveautés 2014-2015

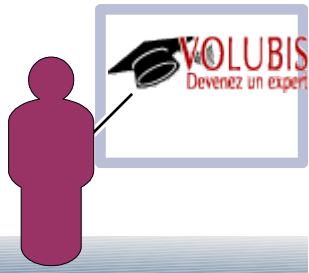
19 et 20 mai 2015 – IBM Client Center, Bois-Colombes

Volubis.fr

Conseil et formation sur OS/400, I5/OS puis IBM *i* depuis 1994 !

Dans nos locaux, vos locaux ou par Internet

Christian Massé - cmasse@volubis.fr



Modernisation IBM i – Nouveautés 2014-2015

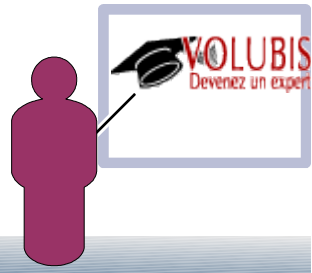
19 et 20 mai 2015 – IBM Client Center, Bois-Colombes

Volubis.fr

Base de connaissance depuis 1995 (plus de 500 cours)

Nouveau *Cours en ligne (accessibles en mode « replay »)*

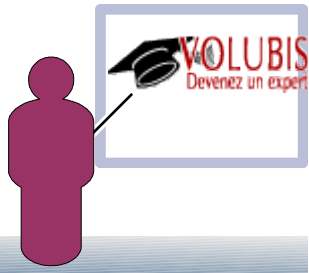
*Accédez à tout cela **gratuitement** pendant trois semaines cet été !*



Modernisation IBM i – Nouveautés 2014-2015

19 et 20 mai 2015 – IBM Client Center, Bois-Colombes

Session 14 : R C A C



Version 7.2

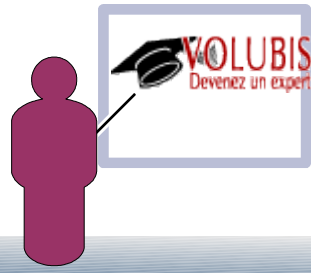
Nouvelle clause VIOLATION sur les Check constraint

- **ON INSERT VIOLATION SET column-name = DEFAULT**
- **L'erreur n'est pas signalée, la valeur par défaut est insérée**

- **ON UPDATE VIOLATION PRESERVE column-name**
  L'erreur n'est pas signalée, la valeur précédente est conservée

- **Exemple**

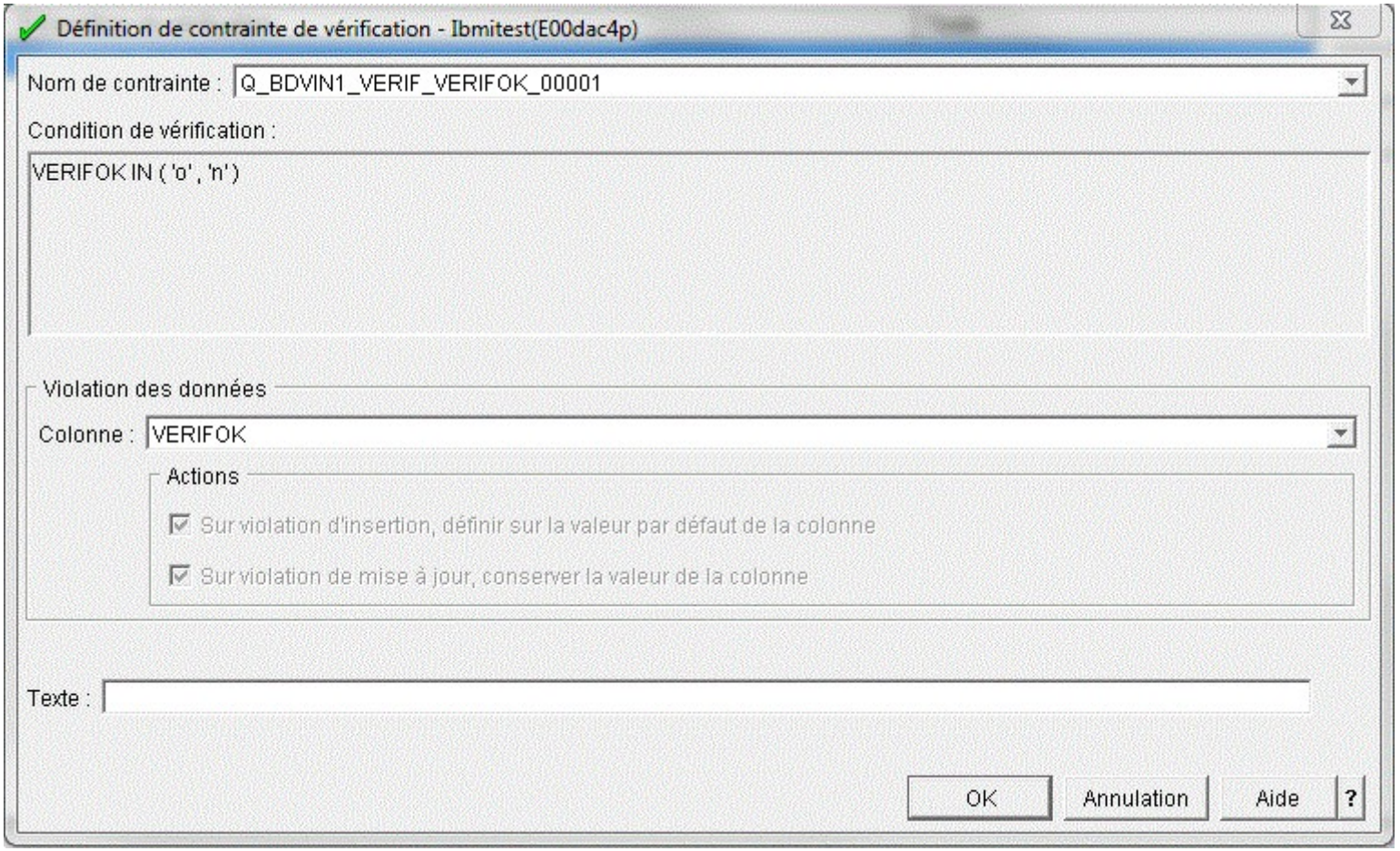
```
create table bdvin1/verif (
 cle int as identity,
 libelle char(50),
 verifOK char(1) default 'o' check (verifok in ('o' , 'n') )
                 on insert violation set verifOK = DEFAULT
                 ON UPDATE VIOLATION PRESERVE verifOK
)
La table VERIF a été créée dans BDVIN1.
```

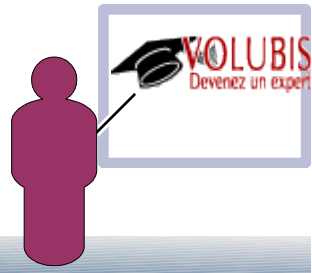


Version 7.2

Nouvelle clause VIOLATION sur les Check constraint

- **Vu pas System i Navigator**





Version 7.2

Nouvelle clause VIOLATION sur les Check constraint

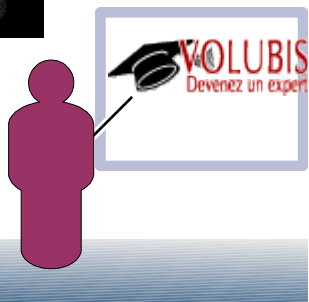
•suite à deux INSERT, dont l'un ne renseignant pas la colonne Verifok, nous avons bien les valeurs attendues

```
select * from VERIF
CLE    LIBELLE                                                    VERIFOK
  1    test1                                                         o
  2    test1                                                         o
```

Mais suite à cet INSERT qui aurait du être refusé

```
INSERT INTO BDVIN1/VERIF (LIBELLE, VERIFOK) VALUES('test2', 'x')
1 lignes insérées dans VERIF de BDVIN1.
```

```
CLE    LIBELLE                                                    VERIFOK
  1    test1                                                         o
  2    test1                                                         o
  4    test2                                                         o
```



Version 7.2

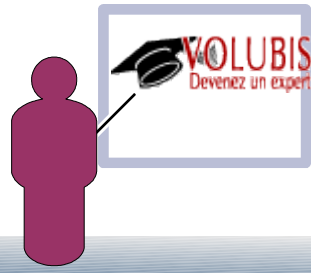
Nouvelle clause VIOLATION sur les Check constraint

- **Enfin, suite à cet UPDATE, lui aussi invalide**

```
update bdvin1/verif set verifok = ' '
 where cle = 1
1 lignes mises à jour dans VERIF de BDVIN1.
```

CLE	LIBELLE	VERIFOK
1	test1	o
2	test1	o
4	test2	o

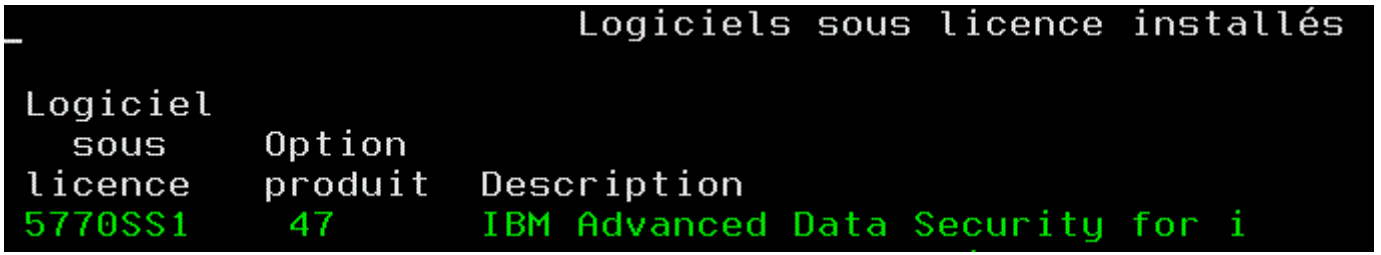
- **NB : Aucun message dans la LOG pour signaler les "remplacement" de valeur**



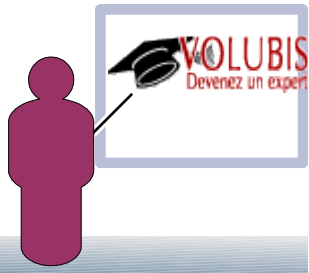
Version 7.2

L' option 47 de 5770SS1 (non facturable) apporte RCAC

- Row and Column Access Control

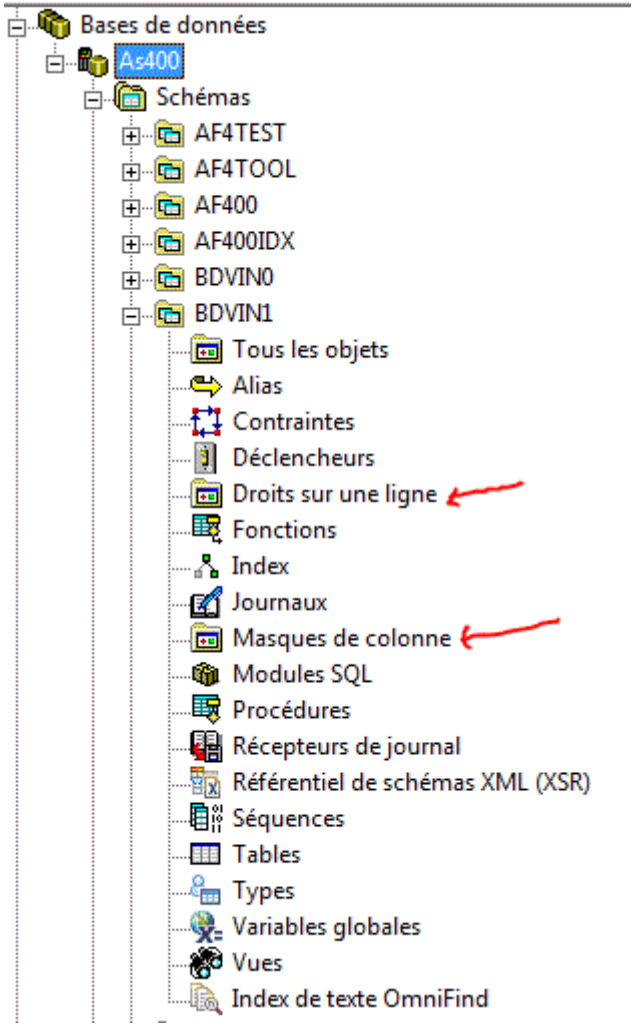


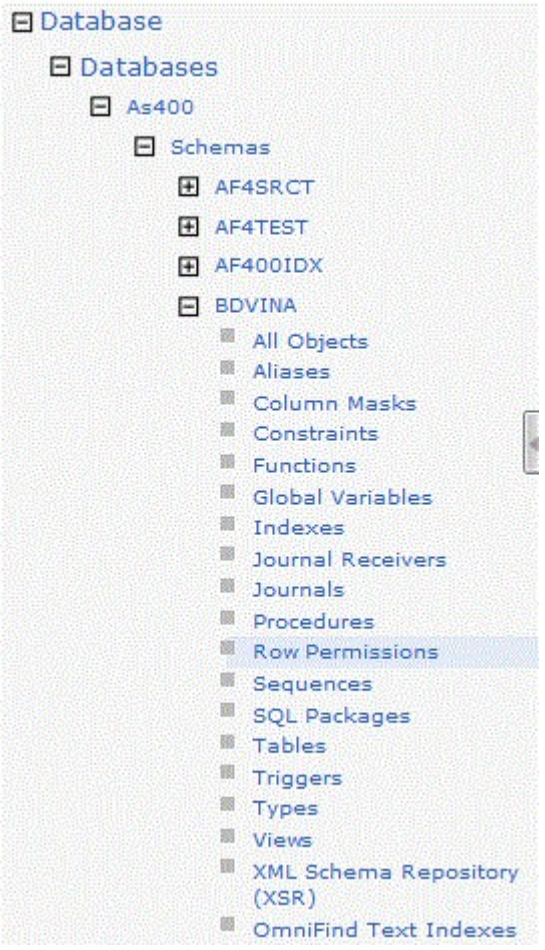
Il s'agit de pouvoir indiquer des « droits » à la colonne ou à la ligne qui s'appliquent y compris aux personnes ayant les droits d'administrateur

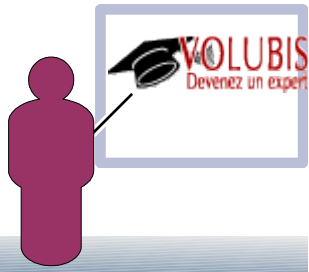


RCAC

Ces deux nouvelles options sont accessibles via
System i Navigator ou Navigator for i (version Web).



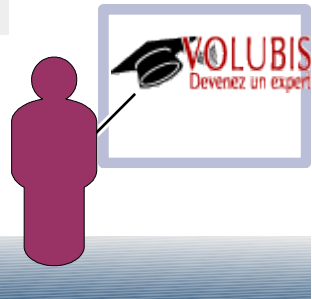




RCAC

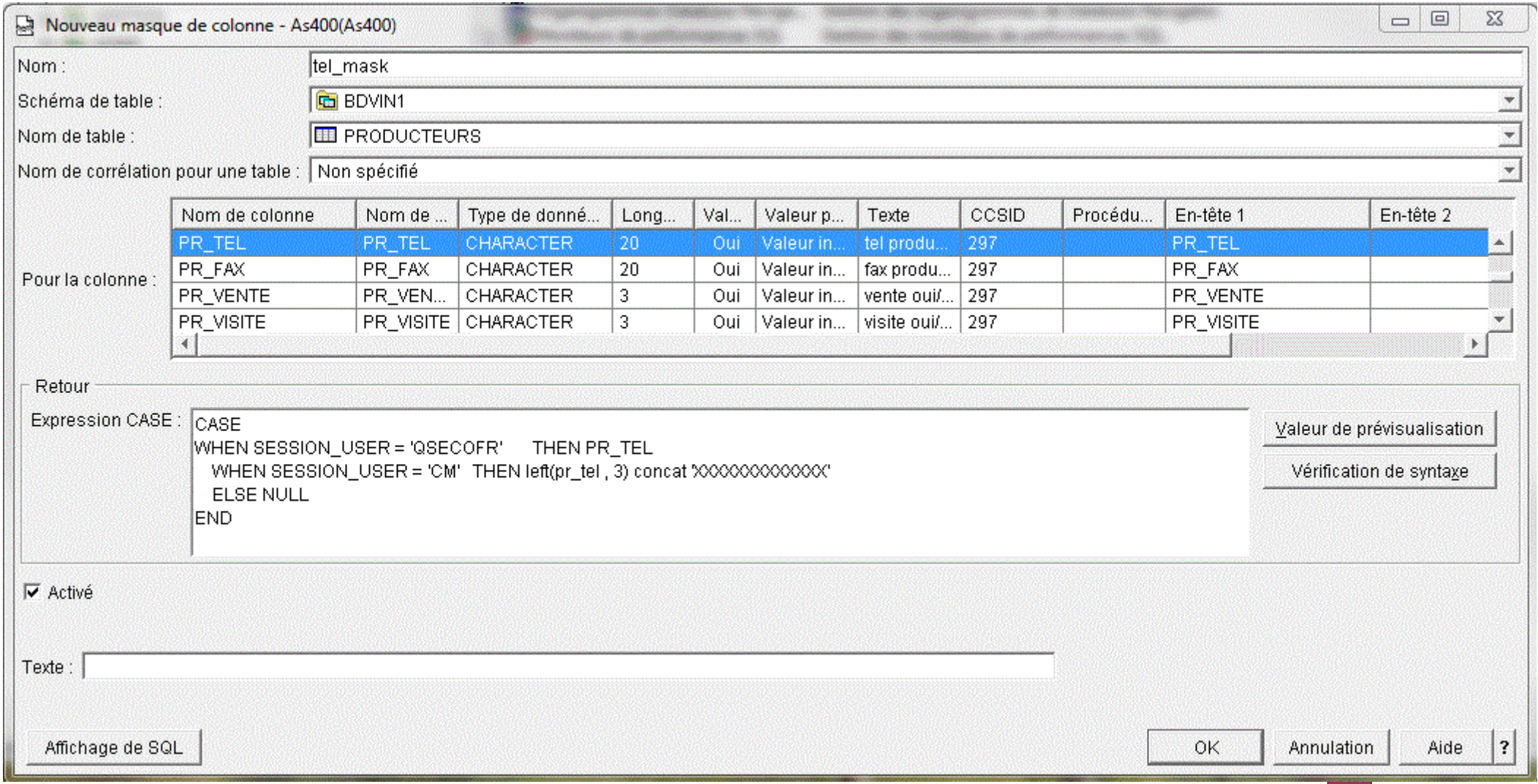
CREATE MASK indique si une colonne est retournée tel que ou totalement/partiellement masquée ('xxx-xxx-xxx-1234' pour un n° de CB)

```
CREATE [or REPLACE] MASK tel_MASK ON bdvin1/producteurs
 FOR COLUMN pr_tel RETURN
  CASE
    WHEN SESSION_USER = 'QSECOFR'
                              THEN PR_TEL
    WHEN SESSION_USER = 'CM'
                              THEN left(pr_tel , 3) concat 'XXXXXXXXXXXX'
    ELSE NULL
END
ENABLE
```



RCAC

Sous System i Navigator

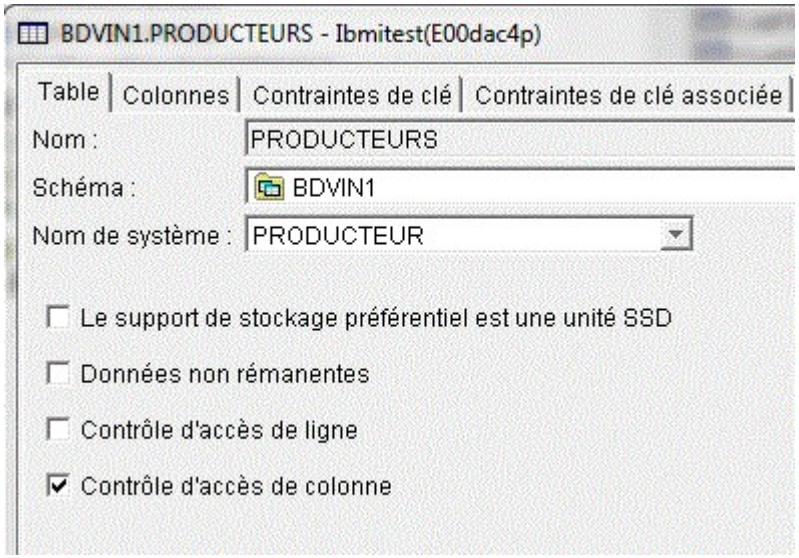


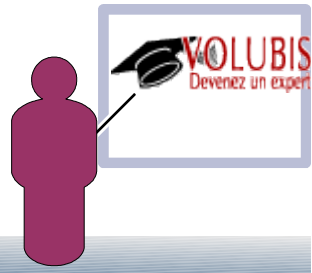
RCAC

Puis

```
ALTER TABLE bdvin1/producteurs
  ACTIVATE COLUMN ACCESS CONTROL
```

ou





RCAC

Attention vous devez avoir les droits QIBM_DB_SECADM (même QSECOFR !)

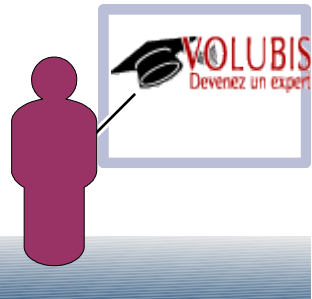
•sinon vous recevrez SQL0552 Non autorisé à utiliser CREATE MASK.

pour donner ce droit : WRKFCNUSG

```
                    Gestion de l'utilisation de fonctions

Indiquez vos options, puis appuyez sur ENTREE.
  2=Modifier l'utilisation   5=Utilisation

Opt  ID fonction                    Nom de la fonction
 _   QIBM_DIRSRV_ADMIN              IBM Tivoli Directory Server Administrator
 _   QIBM_ACCESS_ALLOBJ_JOBLOG      Accès à l'historique de travail du trava
 _   QIBM_ALLOBJ_TRACE_ANY_USER     Trace any user
 _   QIBM_WATCH_ANY_JOB            Watch any job
 _   QIBM_DB_DDMDRDA                DDM & DRDA Application Server Access
 _   QIBM_DB_SECADM                 Administrateur sécurité base de données
 _   QIBM_DB_SQLADM                 Administrateur de base de données
 _   QIBM_DB_SYSMON                 Informations de base de données
```



RCAC

Option 2 pour modifier

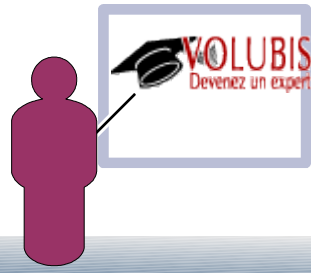
```
                  Modifier utilisation fonction (CHGFCNUSG)

Indiquez vos choix, puis appuyez sur ENTREE.

ID fonction  . . . . . . . . . . > QIBM_DB_SECADM
Utilisateur  . . . . . . . . . .   __________      Nom
Utilisation  . . . . . . . . . .   _______         *ALLOWED, *DENIED, *NONE
Droit par défaut . . . . . . . .   *DENIED         *SAME, *ALLOWED, *DENIED
Droit spécial *ALLOBJ  . . . . .   *NOTUSED        *SAME, *USED, *NOTUSED
```

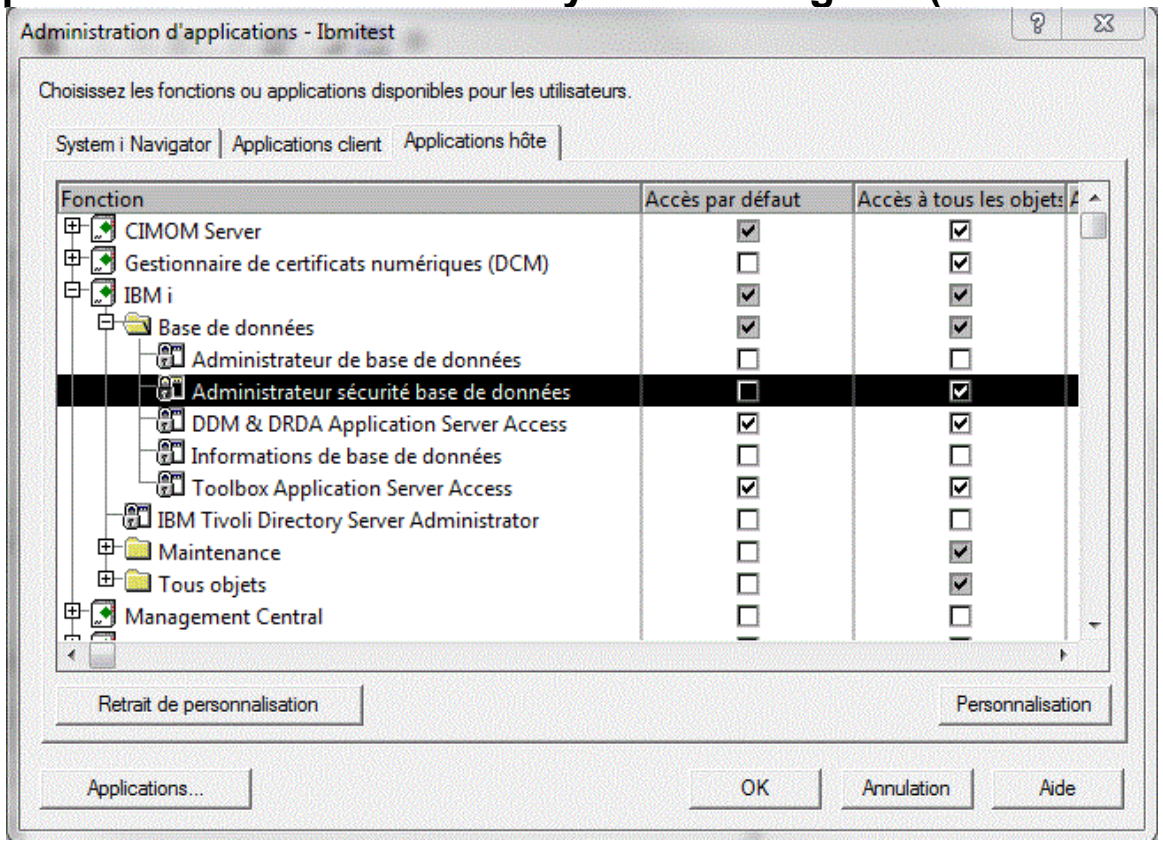
Indiquez :

- ***USED face à *Droits spécial *ALLOBJ* pour que QSECOFR puisse manipuler ces notions**
- ***ALLOWED face à *Droits par défaut* pour que TOUT LE MONDE puisse manipuler ces notions (déconseillé !)**
- **sinon indiquez un profil à ajouter (Utilisateur) et *ALLOWED (Utilisation) pour autoriser un profil ou un groupe**

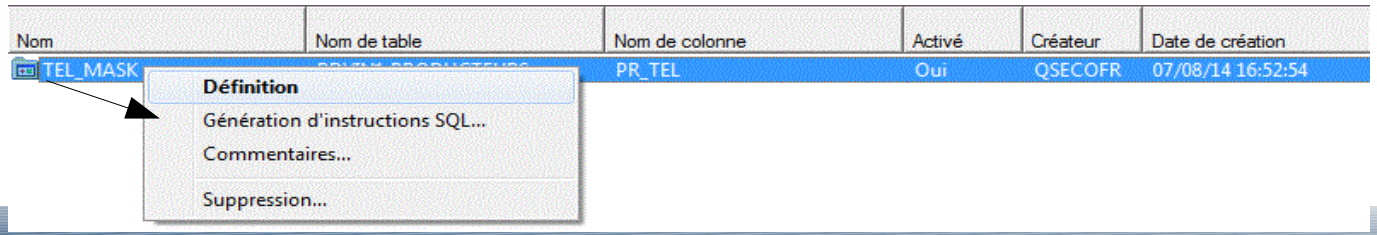


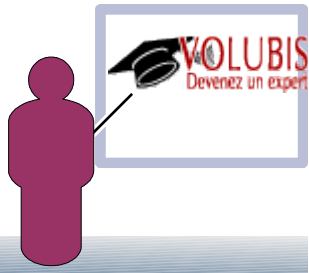
RCAC

Vous pouvez aussi le faire avec System i Navigator (administration d'applications)



la restriction étant posée, vous pouvez la modifier, retrouver le source et la supprimer



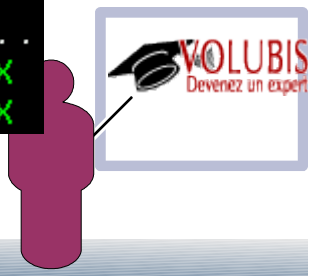


RCAC

Les restrictions RCAC s'appliquent dans tous les contextes :

DSPPFM sous QSECOFR

DSPPFM sous CM



RCAC

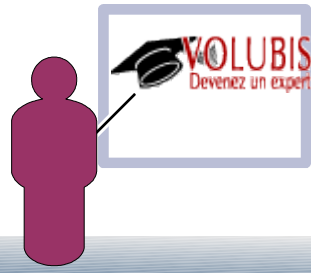
Bien sur la valeur retournée doit être compatible

- **1er test refusé le prix est numérique**

```
Le type de données du littéral *N n'est pas compatible avec la colon
create or replace MASK prixMASK ON bdvin1/ma_cave
       FOR COLUMN cav_prix RETURN
     CASE
       WHEN SESSION_USER = 'QSECOFR'
        THEN cav_prix
       WHEN SESSION_USER = 'CM'
        THEN 'xxxxxxxxxx'
       ELSE NULL
     END
 ENABLE
```

- **2eme, OK, on retourne 0**

```
> create or replace MASK prixMASK ON bdvin1/ma_cave
         FOR COLUMN cav_prix RETURN
       CASE
         WHEN SESSION_USER = 'QSECOFR'
          THEN cav_prix
         WHEN SESSION_USER = 'CM'
          THEN 0
         ELSE NULL
       END
   ENABLE
  Instruction CREATE MASK terminée pour PRIXMASK de BDVIN1.
```



RCAC

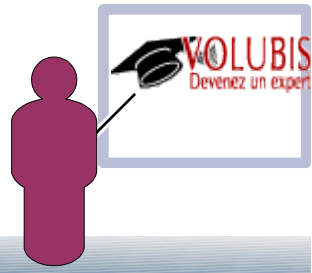
Un MASK n'empêche pas les insertions

(par contre vous ne retrouvez pas forcement la donnée telle que vous l'avez insérée, mais masquée)

•Le problème se pose éventuellement lors des mises à jour :

Exemple , le pgm suivant lit et modifie le producteur 1 en RPG
(fichier qui possède un MASK sur PR_TEL)

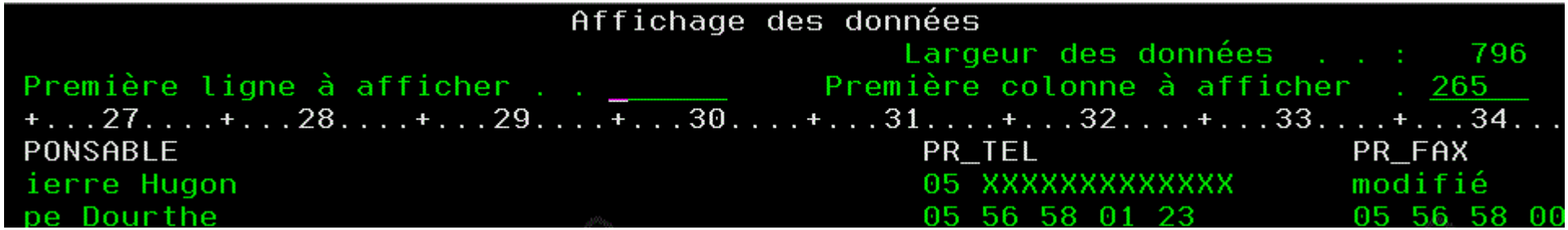
```
ctl-opt alwnull(*usrctl);
dcl-F producteur disk usage(*update);
chain 1 producteur;
pr_fax = 'modifié';
update prodf ;
*inLR = *on;
```

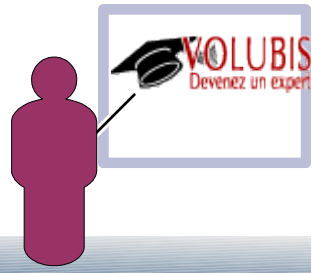


RCAC

lors de l'Update RPG, il met à jour la ligne suivant les données qu'il a lui même reçu.

- **Suite à un CALL par CM (c'est QSECOFR qui regarde le contenu de la table) :**

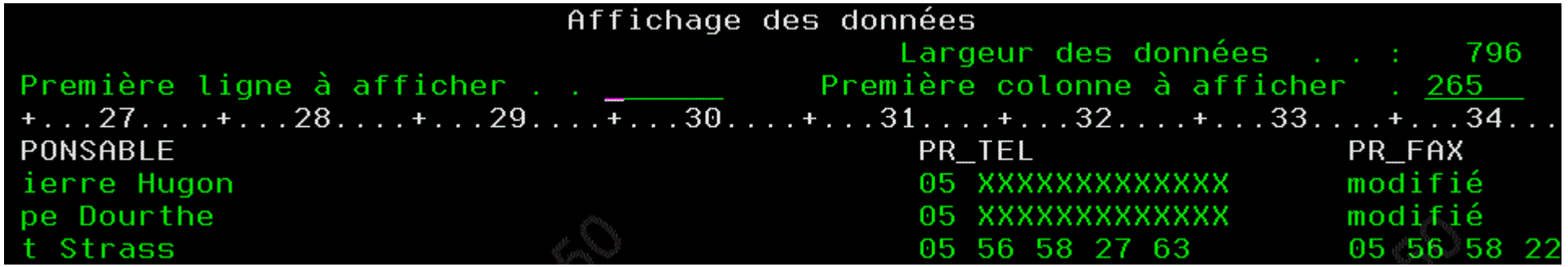


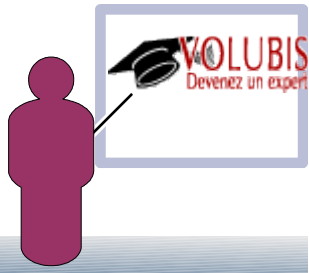


RCAC

Y compris en utilisant le fonction %fields()

```
ctl-opt alwnull(*usrctl);
dcl-F producteur disk usage(*update);
chain 2 producteur;
pr_fax = 'modifié';
update prodf %fields(pr_fax) ;
*inLR = *on;
```





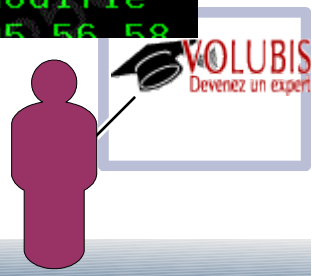
RCAC

Seul un UPDATE SQL (toujours réalisé par CM) , ne met à jour que certains champs (donc ne touche pas aux autres)

```
                          Entrée d'instructions SQL

Saisissez l'instruction SQL, puis appuyez sur ENTREE.
   > UPDATE BDVIN1/PRODUCTEURS SET PR_FAX = 'Modifié' WHERE pr_code = 3
     1 lignes mises à jour dans PRODUCTEUR de BDVIN1.
===> _
```

```
                          Affichage des données
                                          Largeur des données  . . :   796
Première ligne à afficher . . _____     Première colonne à afficher  . 265
+...27....+...28....+...29....+...30....+...31....+...32....+...33....+...34.
PONSABLE                                PR_TEL                  PR_FAX
ierre Hugon                             05 XXXXXXXXXXXXX        modifié
pe Dourthe                              05 XXXXXXXXXXXXX        modifié
t Strass                                05 56 58 27 63          Modifié
t Vanderhaeden                          05 56 58 80 04          05 56 58
```



RCAC

La documentation conseille dans le cas d'entrées/sorties « natives » :

1/ de faire un trigger, qui rétablisse l'ancienne valeur

IF substr(new.PR_TEL, 4, 13) = 'XXXXXXXXXXXXX' THEN
new.PR_TEL = old.PR_TEL;

... /...

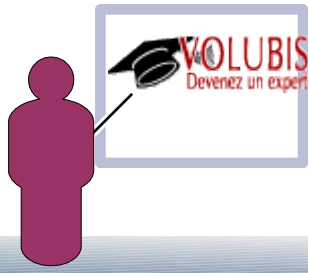
2/ de faire une contrainte qui refuse la valeur produite par le MASK

CHECK substr(PR_TEL, 4, 13) <> 'XXXXXXXXXXXXX'

Ou **mieux**, faites une contrainte qui utilise la nouvelle clause
ON UPDATE VIOLATION

CHECK substr(PR_TEL, 4, 13) <> 'XXXXXXXXXXXXX'
ON UPDATE VIOLATION PRESERVE PR_TEL

L'ancienne valeur sera alors replacée automatiquement,
sans message d'erreur



RCAC

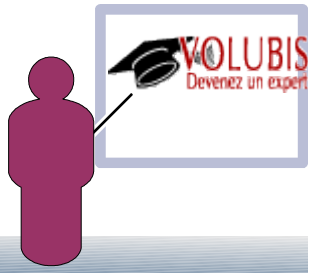
Enfin, les données sont masquées, juste avant l'affichage,
c'est à dire après jointure et GROUP BY

par exemple, le SELECT suivant, qui donne le nombre de clients par indicatif téléphonique

```
SELECT LEFT(PR_TEL, 5) , count(*) as
nombre
  FROM producteurs
GROUP BY LEFT(PR_TEL, 5)
```

affiche

si la colonne n'est pas masquée		si la colonne est masquée	
05 55	3	05 xx	3
05 56	7	05 xx	7
05 58	4	05 xx	4



RCAC

CREATE PERMISSION indique la(les) règles(s) qui font qu'une ligne peut être vue

•Toute ligne ne correspondant pas à la règle n'est pas retournée :

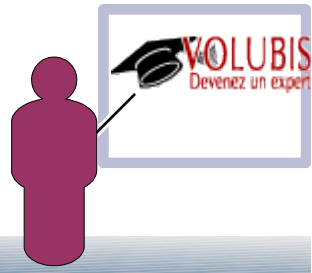
Exemple CM ne doit pas voir l'appellation 13

```
CREATE [or REPLACE] PERMISSION VINS_ROW_ACCESS ON bdvin1/vins
FOR ROWS
 WHERE
    SESSION_USER <> 'CM'
       OR (SESSION_USER = 'CM'
             and (appel_code <> 13 or appel_code IS NULL)
           )
ENFORCED FOR ALL ACCESS
ENABLE
```

rappelez vous, on indique ce qui peut être vu (une affirmation, donc)

Puis

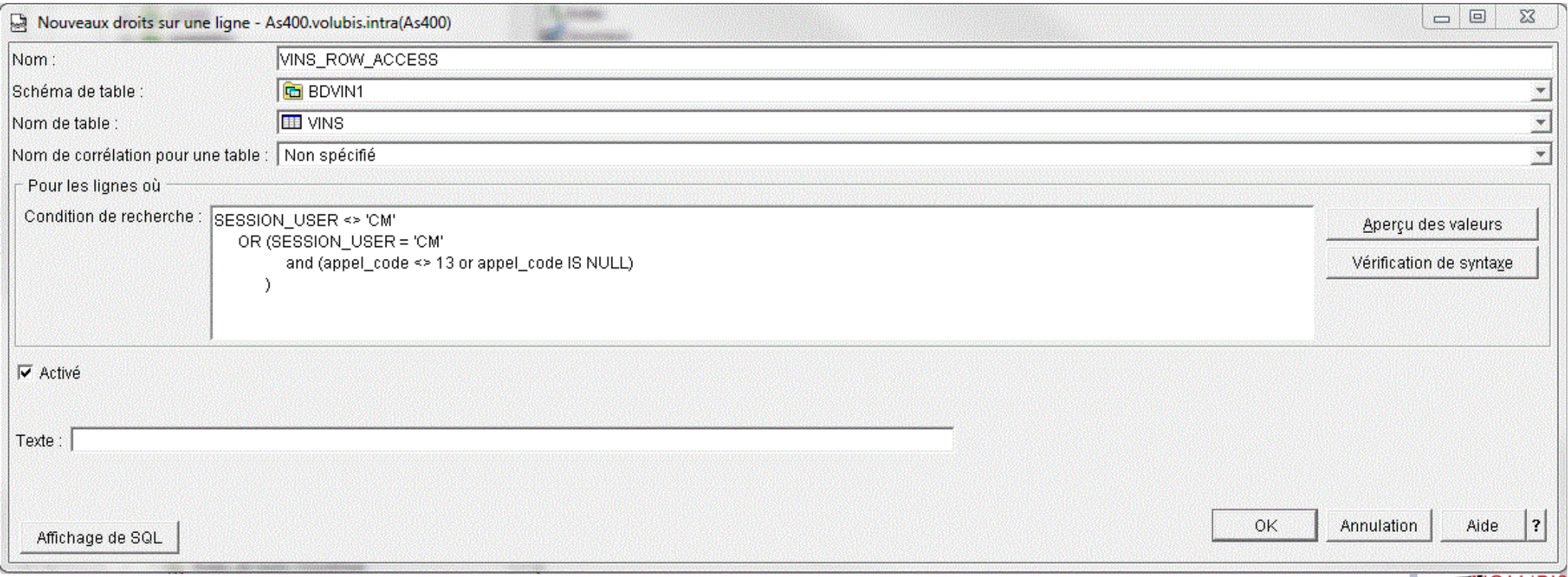
```
ALTER TABLE bdvin1/vins
  ACTIVATE ROW ACCESS CONTROL
```



RCAC

CREATE PERMISSION indique la(les) règles(s) qui font qu'une ligne peut être vue

- **Sous System i Navigator**



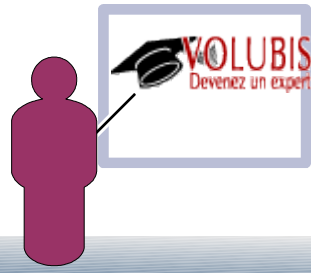
RCAC

On aurait aussi pu faire deux permissions

```
CREATE PERMISSION VINS_ROW_ACCESS1 ON bdvin1/vins
FOR ROWS
 WHERE
    SESSION_USER <> 'CM'

ENFORCED FOR ALL ACCESS
ENABLE ;
-------------------------------------------------------------------------------
CREATE PERMISSION VINS_ROW_ACCESS2 ON bdvin1/vins
FOR ROWS
 WHERE
    SESSION_USER = 'CM'
              and (appel_code <> 13 or appel_code IS NULL)
ENFORCED FOR ALL ACCESS
ENABLE ;
```

un peu moins performant en temps de réponse...



RCAC

Le système ajoute alors une permission implicite QIBM_DEFAULT_nomdetable_schema où la permission est si 0=1, donc toujours fausse.

- **seules les permissions explicites autorisent des lignes à être vues (en bref tout ce qui n'est pas autorisé est interdit)**

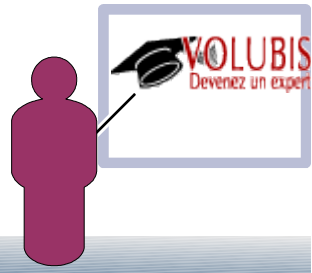
- **Donc, Attention, si vous enlever la permission sans désactiver ROW ACCESS CONTROL plus aucune ligne ne peut être extraite (le fichier apparaît toujours comme vide !)**

Avec notre PERMISSION "VINS_ROW_ACCESS" :

SELECT COUNT(*) from VINS , sous QSECOFR affiche 25.221

SELECT Count(*) from VINS WHERE appel_code = 13 indique un nombre de 811 vins pour cette appellation

SELECT COUNT(*) from VINS , sous CM affiche 24.410



RCAC

Une PERMISSION, peut empêcher une insertion ou une mise à jour, qui ne respecte pas la règle

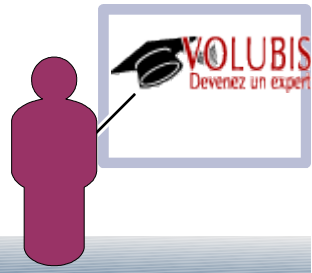
•Exemple avec APPEL_CODE à 13 (vous recevez SQ20471)

```
> INSERT INTO BDVIN1/VINS VALUES(999999, 1, 'test', 'Cabernet', null,
  null, null, null, null, null, 13, 1, null)
  Instruction INSERT ou UPDATE non conforme aux droits de ligne.
```

•Administration :

- **pour modifier : ALTER MASK | PERMISSION**
- **pour retirer : DROP MASK | PERMISSION**

tant que vous n'avez pas activé les droits par ALTER TABLE,
les MASK et les PERMISSIONS sont inopérants



RCAC

nouvel ordre ALTER TRIGGER en version 7.2

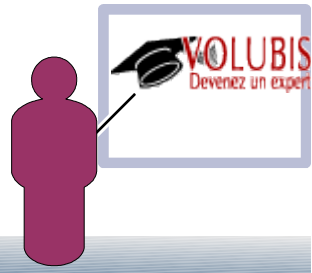
pour modifier certains paramètres d'un TRIGGER :

ENABLE, le trigger est actif (dft)
DISABLE, le trigger n'est plus actif

SECURED ce trigger est sécurisé (compatible) avec les droits RCAC
NOT SECURED ce trigger n'est pas compatible avec les droits RCAC (le défaut)

Il est impossible de modifier cet attribut quand des droits RCAC sont actifs

- **il est impossible de créer un trigger NOT SECURED quand des droits RCAC sont actifs**



RCAC

```
> create trigger bdvin1.prdtrg
   AFTER INSERT ON BDVIN1.PRODUCTEUR
  BEGIN
    call QCMDEXC('sndmsg msg(''producteur ajouté'') tousr(cm)');
  END
  PRDTRG de BDVIN1 ne peut pas être utilisé pour le contrôle d'accès d
```

Détail du message SQ20470

```
ID message . . . . . :   SQ20470


Message . . . . :   PRDTRG de BDVIN1 ne peut pas être utilisé pour le contrôle
  d'accès de ligne ou de colonne.

Cause . . . . . : PRDTRG de BDVIN1 type TRIGGER ne peut pas être créé ou
  modifié pour l'une des raisons suivantes car PRODUCTEUR de BDVIN1 type FILE
  dépend de celui-ci pour le contrôle d'accès de ligne ou de colonne.
    -- Une fonction définie par l'utilisateur doit être sécurisée si elle est
  référencée dans un DROIT ou un MASQUE ou dans une vue elle-même référencée
  par un DROIT ou un MASQUE.
    -- Un déclencheur doit être sécurisé s'il est défini sur une table avec un
  DROIT ou un MASQUE actif.
    -- Un déclencheur doit être sécurisé s'il est défini sur une vue basé sur
  une table avec un DROIT ou un MASQUE actif.
```

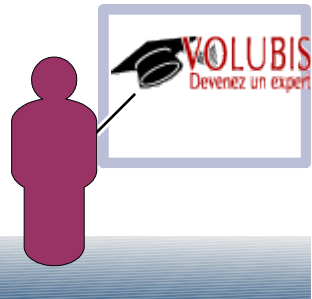
RCAC

Par contre, cet ordre fonctionne

```
> create trigger bdvin1.prdtrg
   AFTER INSERT ON BDVIN1.PRODUCTEUR SECURED
  BEGIN
    call QCMDEXC('sndmsg msg(''producteur ajouté'') tousr(cm)');
  END
  Déclencheur PRDTRG créé dans BDVIN1.
```

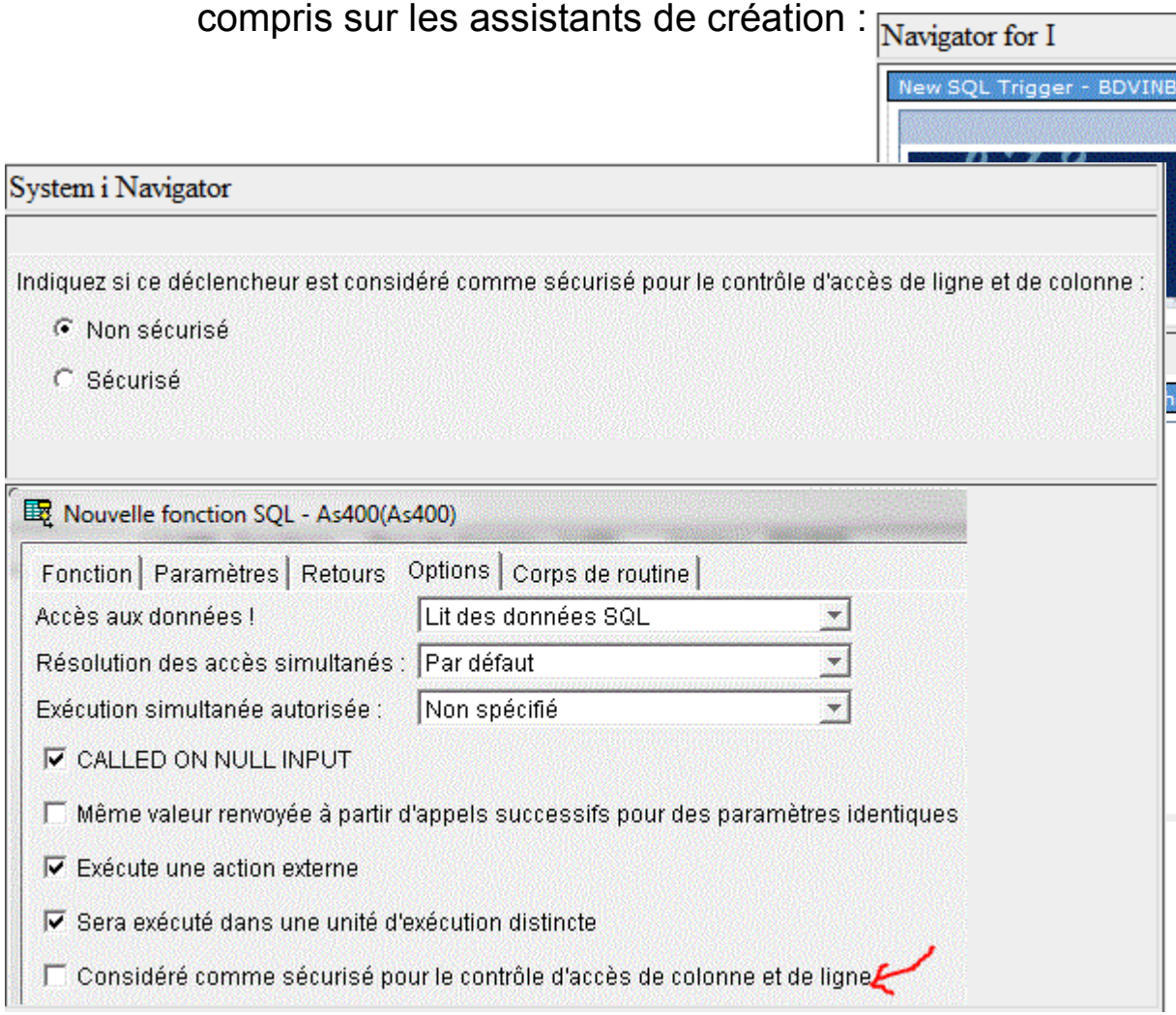
Lors d'un insert, le trigger est lancé sans problème

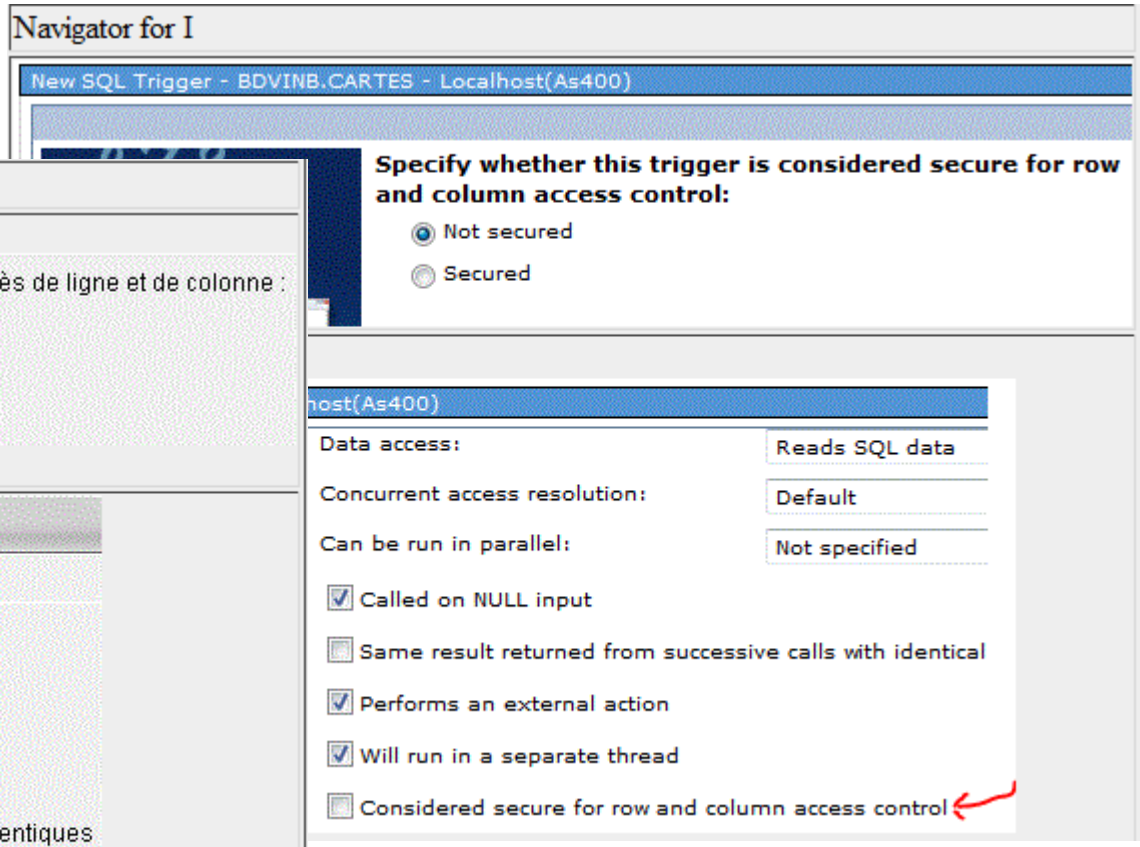
```
                    Messages ne nécessitant pas de réponse
_     producteur ajouté
        De     . . :   QSECOFR         23/07/14   16:02:07
```

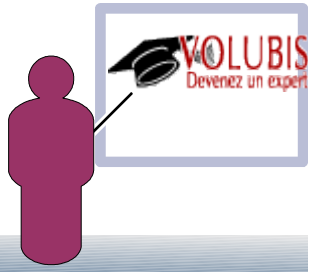


RCAC

Les mêmes paramètres sont proposés sur CREATE TRIGGER / CREATE FUNCTION, y compris sur les assistants de création :







RCAC

La mise en place ou le retraits des droits RCAC :

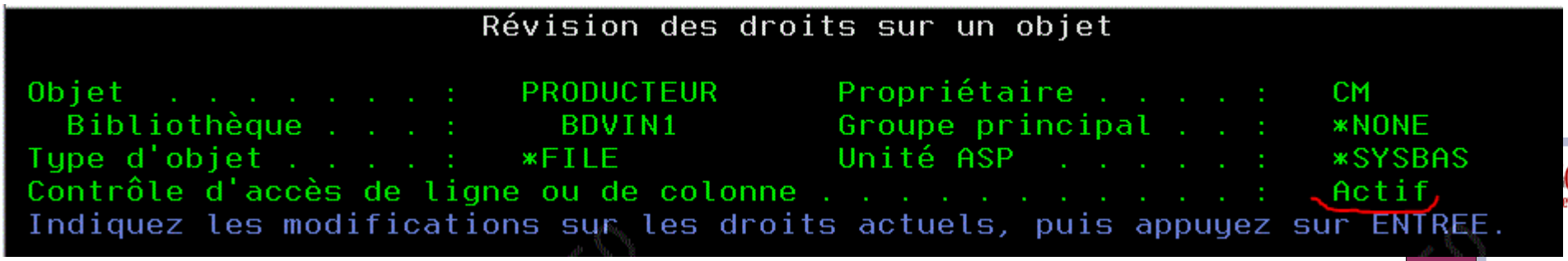
impose une allocation exclusive de la table

provoque un enregistrement AX dans le journal d'AUDIT.

et des enregistrements dans le journal de la table, de code D, type :

- M1 : CREATE MASK
- M2 : DROP MASK
- M3 : ALTER MASK
- P1 : CREATE PERMISSION
- P2 : DROP PERMISSION
- P3 : ALTER PERMISSION

Les droits RCAC sont visibles (mais non modifiables) par EDTOBJAUT/DSPOBJAUT

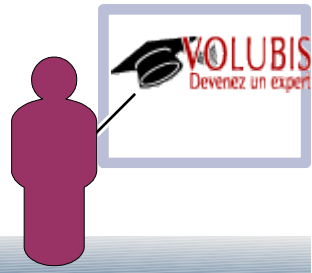


RCAC

Pour voir la liste des droits RCAC, regarder SYSCONTROLS et SYSCONTROLSDEP

• SysControls de QSYS2

RCAC_SCHEMA	*VARCHAR(128)*
RCAC_NAME	*VARCHAR(128)*
RCAC_OWNER	*VARCHAR(128)*
TABLE_SCHEMA	*VARCHAR(128)*
TABLE_NAME	*VARCHAR(128)*
TBCORRELATION	*VARCHAR(128)*
COLUMN_NAME	*VARCHAR(128)*
SYSTEM_COLUMN_NAME	*CHAR(10)*
SYSTEM_TABLE_NAME	*CHAR(10)*
SYSTEM_TABLE_SCHEMA	*CHAR(10)*
CONTROL_TYPE	*CHAR(1)* **M=Mask | R=Row permission**
ENFORCED	*CHAR(1)*
IMPLICIT	*CHAR(1)*
ENABLE	*CHAR(1)*
CREATE_TIME	*TIMESTAMP*
LAST_ALTERED	*TIMESTAMP*
IASP_NUMBER	*SMALLINT*
LABEL	*VARCHAR(50)*
LONG_COMMENT	*VARCHAR(2000)*
RULETEXT.	*DBCLOB* -> **c'est ici qu'est la règle,** en clair.



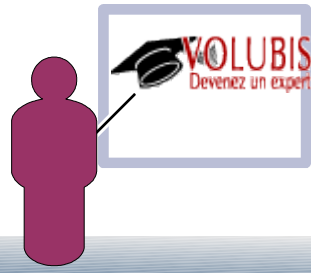
RCAC

SYSCONTROLSDEP affiche les éléments dépendants, par exemple :

```
CREATE TABLE USEROK
(username char(10), appel_code int ) ;
-------------------------------------------------------------------------
-- l'utilisateur CM ne doit voir que les appellations 13 et 144

INSERT INTO USEROK VALUES('CM', 13);
INSERT INTO USEROK VALUES('CM', 144);
```

```
CREATE PERMISSION PROD_ROW_ACCESS1 ON producteurs
FOR ROWS
WHERE
appel_code in (select appel_code from USEROK
                    where username = SESSION_USER)
ENFORCED FOR ALL ACCESS
ENABLE ;
-------------------------------------------------------------------------
ALTER TABLE producteurs
ACTIVATE ROW ACCESS CONTROL
ENFORCED FOR ALL ACCESS
ENABLE ;
```

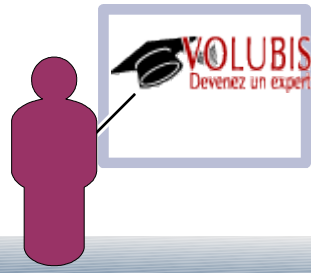


RCAC

SYSCONTROLSDEP contient alors :

```
SELECT SYSCONTROLSDEP.RCAC_NAME, SYSCONTROLSDEP.OBJECT_NAME, SYSCONTROLSDEP.OBJECT_TYPE,
SYSCONTROLSDEP.COLUMN_NAME
  FROM QSYS2.SYSCONTROLSDEP AS SYSCONTROLSDEP;
```

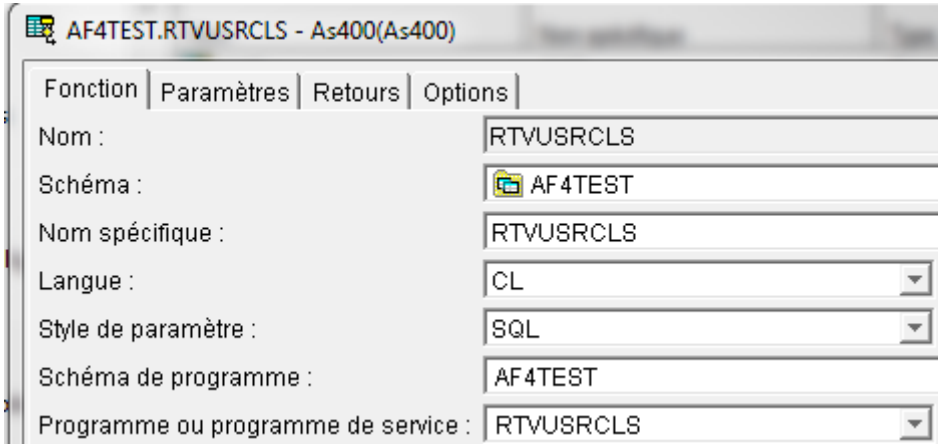
RCAC_NAME	OBJECT_NAME	OBJECT_TYPE	COLUMN_NAME
PROD_ROW_ACCESS1	USEROK	TABLE	
PROD_ROW_ACCESS1	USEROK	COLUMN	APPEL_CODE
PROD_ROW_ACCESS1	USEROK	COLUMN	USERNAME

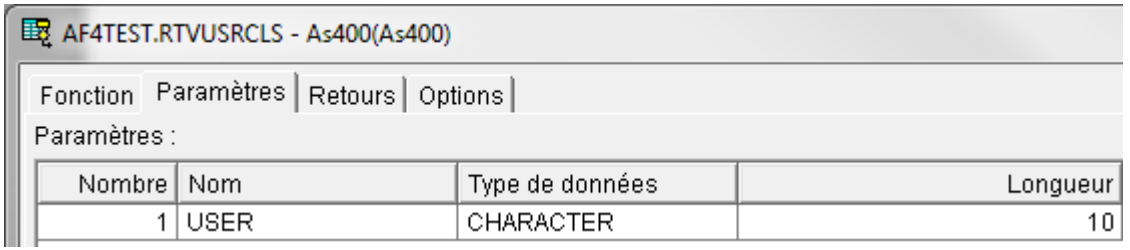


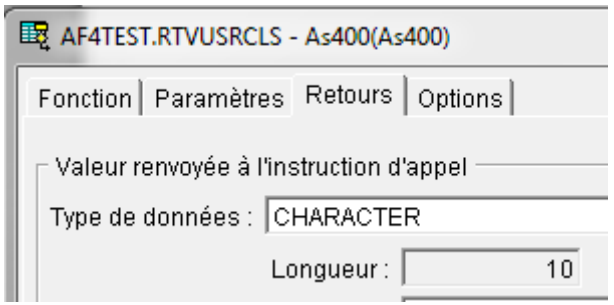
RCAC

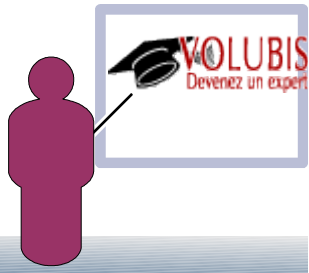
N'hésitez pas à créer vos propres fonctions pour retourner des informations complexes

Ex RTVUSRCLS









RCAC

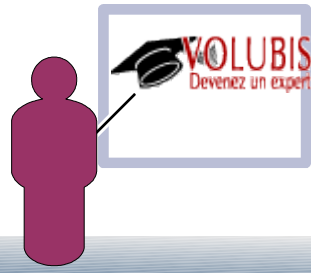
N'hésitez pas à créer vos propres fonctions pour retourner des informations complexes

Ex RTVUSRCLS

```
PGM parm(&profil  &CLS &profilind &CLSind +
         &SQLSTATE &functionQ &function &message)
DCL         VAR(&PROFIL) TYPE(*CHAR) LEN(10)
DCL         VAR(&CLS) TYPE(*CHAR) LEN(10)
DCL         VAR(&PROFILIND) TYPE(*INT) LEN(2)
DCL         VAR(&CLSIND) TYPE(*INT) LEN(2)
DCL         VAR(&SQLSTATE) TYPE(*CHAR) LEN(5)
DCL         VAR(&FUNCTIONQ) TYPE(*CHAR) LEN(519)
DCL         VAR(&FUNCTION) TYPE(*CHAR) LEN(130)
DCL         VAR(&MESSAGE) TYPE(*CHAR) LEN(1000)

RTVUSRPRF  USRPRF(&PROFIL) USRCLS(&CLS)
 MONMSG CPF0000 EXEC(DO)
  CHGVAR &CLSIND -1
 ENDDO
ENDPGM
```

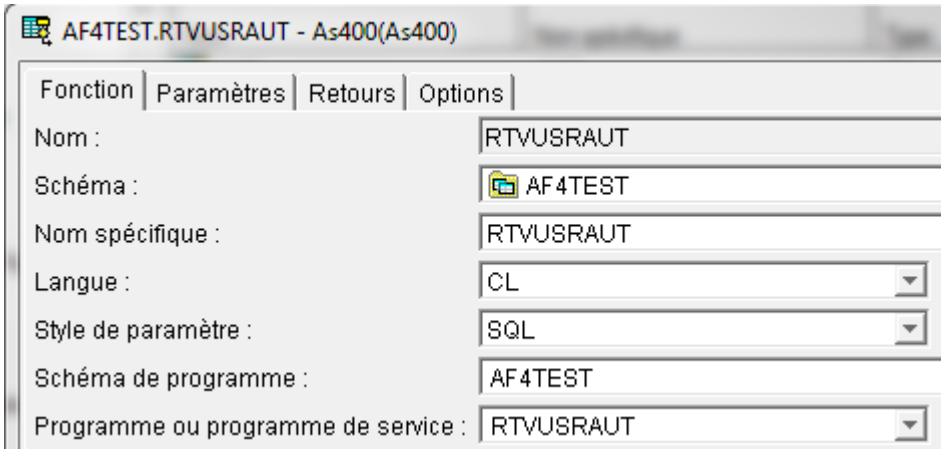
Retourne le paramètre USRCLS ou NULL en cas de problème

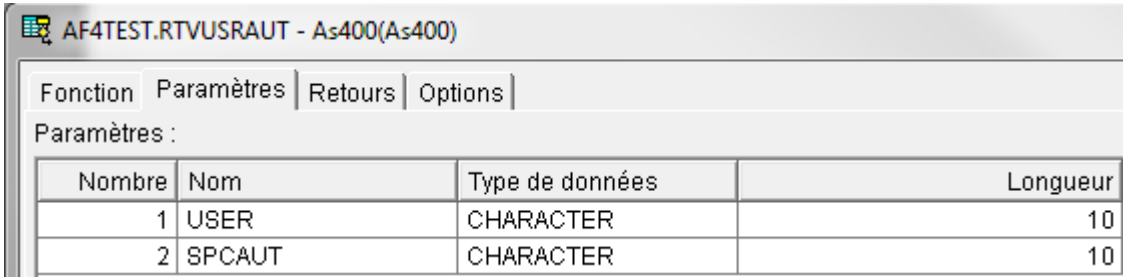


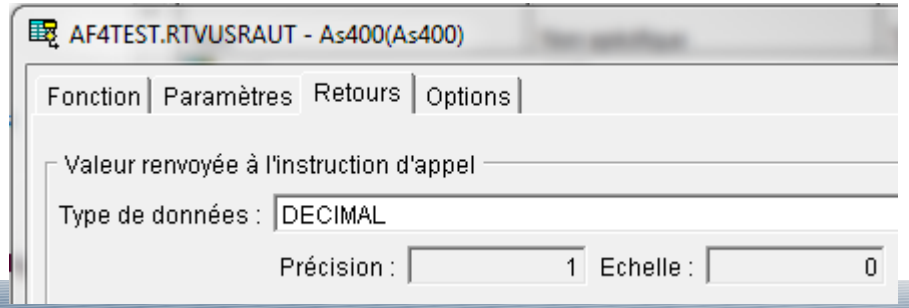
RCAC

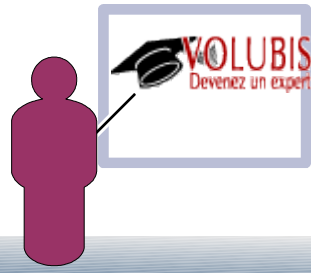
N'hésitez pas à créer vos propres fonctions pour retourner des informations complexes

Ex RTVUSRAUT









RCAC

N'hésitez pas à créer vos propres fonctions pour retourner des informations complexes

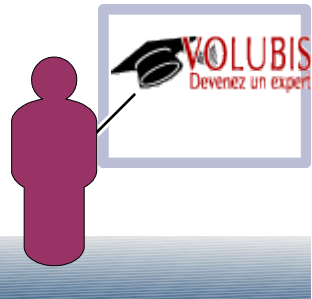
Ex RTVUSRAUT

```
PGM parm(&profil  &aut &YES &profilind &autind &yesind +
         &SQLSTATE &functionQ &function &message)
DCL         VAR(&PROFIL) TYPE(*CHAR) LEN(10)
DCL         VAR(&AUT) TYPE(*CHAR) LEN(10)
DCL         VAR(&YES) TYPE(*DEC) LEN(1 0)
DCL         VAR(&PROFILIND) TYPE(*INT) LEN(2)
DCL         VAR(&AUTIND) TYPE(*INT) LEN(2)
DCL         VAR(&YESIND) TYPE(*INT) LEN(2)
DCL         VAR(&SQLSTATE) TYPE(*CHAR) LEN(5)
DCL         VAR(&FUNCTIONQ) TYPE(*CHAR) LEN(519)
DCL         VAR(&FUNCTION) TYPE(*CHAR) LEN(130)
DCL         VAR(&MESSAGE) TYPE(*CHAR) LEN(1000)

DCL         VAR(&SPCAUT) TYPE(*CHAR) LEN(100)

RTVUSRPRF  USRPRF(&PROFIL) SPCAUT(&SPCAUT)
 MONMSG CPF0000 EXEC(DO)
  CHGVAR &YESIND -1
  RETURN
 ENDDO
```

Retourne NULL en cas de problème



RCAC

N'hésitez pas à créer vos propres fonctions pour retourner des informations complexes

Ex RTVUSRAUT

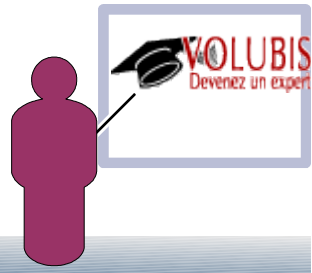
```
/* PROFIL EXISTE, LE DROIT SPÉCIAL EST-IL ATTRIBUÉ ? */
IF          COND(%SCAN(&AUT &SPCAUT) > 0) THEN(DO)
   CHGVAR      VAR(&YES) VALUE(1)
ENDDO
ELSE DO
   CHGVAR      VAR(&YES) VALUE(0)
enddo
ENDPGM
```

Retourne 1 si l'utilisateur à le droit spécial demandé

Permettant CASE

When RTVUSRCLS(session_user) = '*SECOFR '
Or RTVUSRAUT(session_user, '*ALLOBJ') = 1
…/...

*Notez la fonction **%scan** disponible aujourd'hui en CL …. avec plein d'autres !*



RCAC

Impact sur les requêtes SQL :

Autant les masques ne s'appliquent que lors de l'affichage (n'ont donc pas d'impact sur les jointures, par exemple)

Autant les permissions sont plus déterminantes :

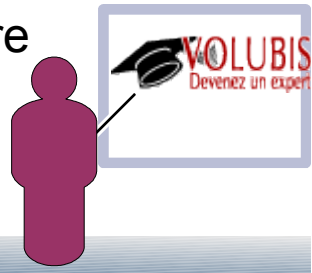
-la jointure n'a lieu qu'avec les lignes autorisées.

-INSERT into cible (SELECT * from SOURCE)

vous subissez les MASK !
vous ne copiez que les lignes autorisés par les permissions

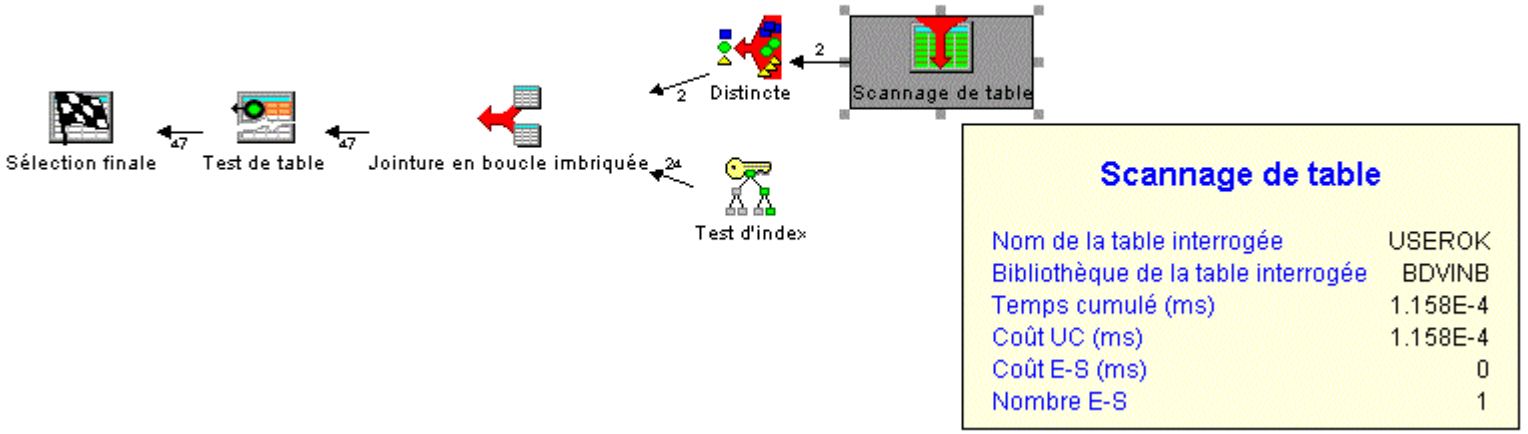
+ les lignes copiées doivent être valides dans la table cible
(si elle même possède des droits RCAC)

SQE fait une jointure, le cas échéant : si le droit RCAC référence une autre table (cas d'utilisation de notre table USEROK)



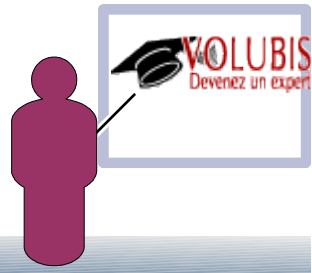
RCAC

ici, sous VISUAL Explain, mais aussi dans les moniteurs SQL



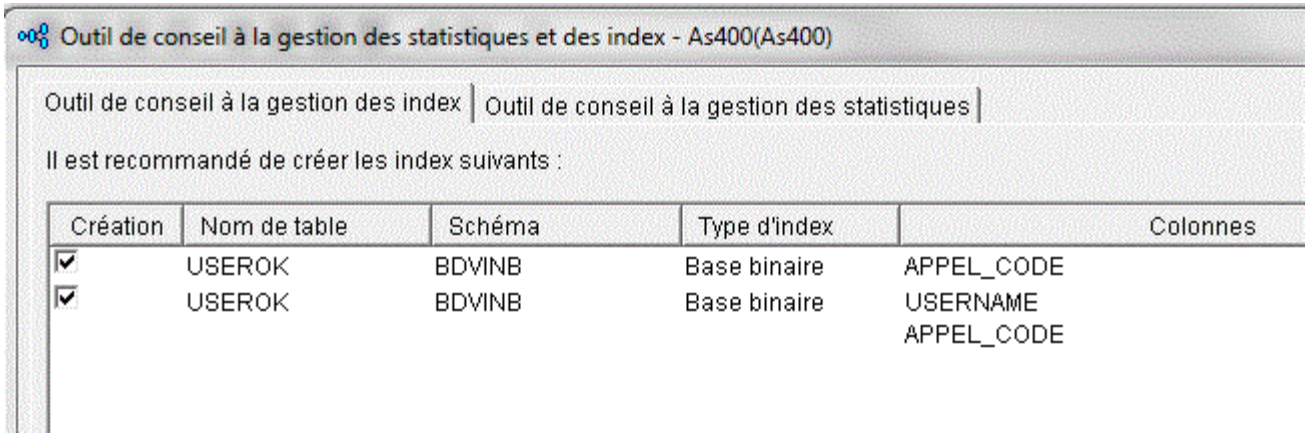
Il signale qu'il y a des droits RCAC

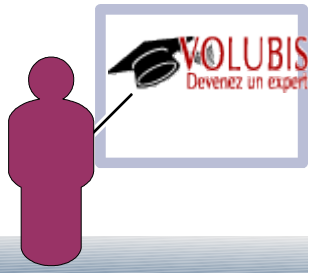
AQP utilisé dans le plan d'accès	Non
Itération du plan d'accès AQP	1
Contrôle d'accès	Colonne



RCAC

VISUAL Explain peut aussi proposer des index



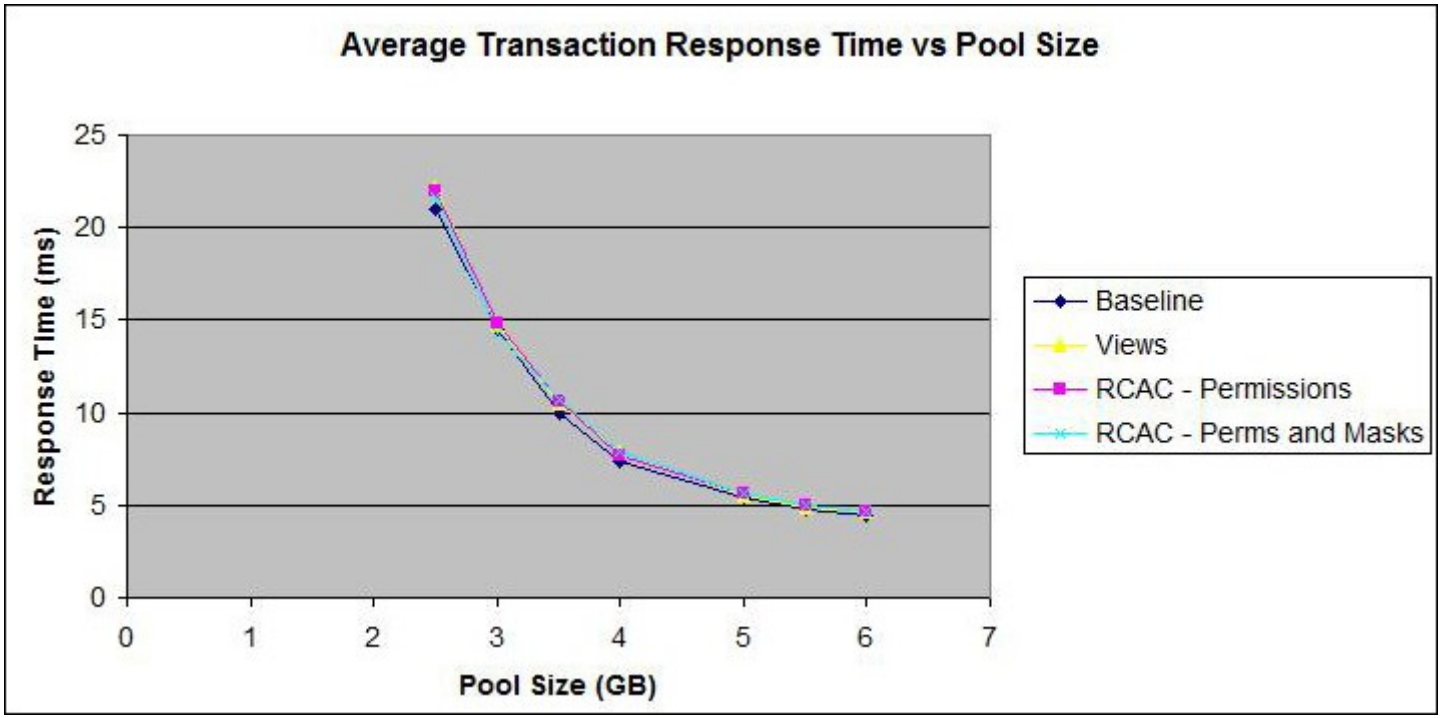


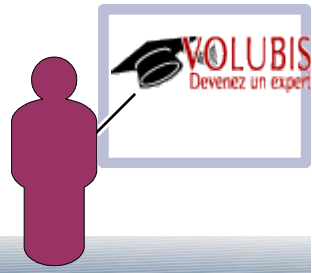
RCAC

Developper Work's signale un surplus en CPU

- Négligeable (< à 1 %) par rapport à une vue réalisant les mêmes services

- Moins de 5 %, par rapport à une lecture sans contrôle





RCAC

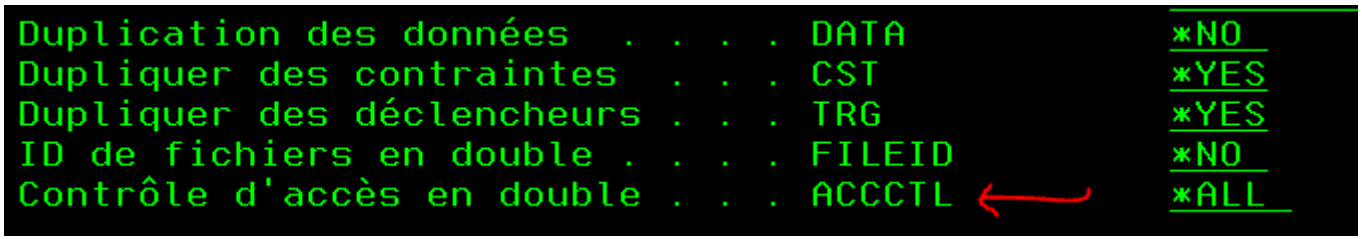
Commandes système

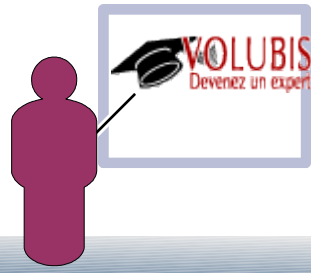
- les droits RCAC sont stockés dans la table elle même, ils sont donc :

  sauvegardés par SAVLIB, SAVOBJ
  déplacés par MOVOBJ
  dupliqués (par défaut) par CRTDUPOBJ

CRTDUPOBJ et CPYLIB

Quand les Data sont dupliquées, elles le sont à l'identique (en clair) , les droits RCAC étant aussi dupliqués, mais il y a un nouveau paramètre sur CRTDUPOBJ





RCAC

Commandes système

- CPYF

Seules les données étant copiées (pas les droits RCAC), ne sont dupliqués que les lignes autorisées, et éventuellement masquées.

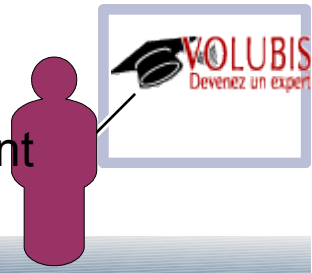
- Réplication manuelle de fichiers, ou ETL :

Le problème est le même que CPYF, il faut donc explicitement autoriser l'utilisateur OU subir les effets de RCAC !

- Attention aux droits

Les données dans le journal et dans une MQT sont en clair (la réplication basée sur la fonction journal se passera bien)

Mais il faut limiter le droit de lire le récepteur, et le droit de lire explicitement une MQT, sinon vous aurez une faille de sécurité !



RCAC

ATTENTION !

une table (ou fichier physique) avec des droits RCAC ne peux pas être sauvegardée pour version précédente.

une table (ou fichier physique) avec des droits RCAC, restaurée sur un système ne possédant pas l'option 47 ne peux plus être **ouverte.**

Merci !