Power Week 2025

#pw2025

18 - 19 - 20 novembre 2025
IBM Innovation Studio Paris

S09- Sécurisez vos données avec Row and Column Access Control (RCAC)

18 novembre 13:30 - 14:30

Birgitta Hauser
Birgitta Hauser – Modernization – Education – Consulting on IBM I
eMail: Hauser@ModEdCon.com / Hauser@SSS-Software.de
Web: https://ModEdCon.com

IBM

common FRANCE

1

Landsberg am Lech

2

Agenda

Row And Column Access Control (RCAC) – Aperçu

- Rôles et séparation des fonctions
- Registres spéciaux pour RCAC
- Fonctions scalaires pour RCAC

Mettre en œuvre et activer les Row Permissions

Mettre en œuvre et activer les Column Masks

- Modifier les lignes avec Column Masks

Restrictions et Risques

3

Row and Column Access Control (RCAC)

4

RCAC – qu'est-ce que c'est?

RCAC = Row and Column Access Control

- **Un niveau Sécurité des données supplémentaire**(disponible avec Db2)
  - Les commandes CL pour sécuriser les objets (base de données) peuvent être utilisées en complément.
- **En plus** de la **sécurité des objets**
- **Limiter** l'accès aux **données autorisées exclusivement**
  - Contrôler l'accès à une table au niveau **des lignes** et/ou **des colonnes**
  - Même les **utilisateurs** avec l'autorisation ***ALLOBJ ne peuvent plus accéder à toutes les données.**

Offre deux méthodes différentes

- Autorisations pour accéder **aux lignes** → `CREATE PERMISSION`
- Masques pour le **contenu des colonnes** → `CREATE MASK`

IBM Advanced Data Security feature for i

- **Doit être installé** → feature gratuit (no-charge), option 47
- Requis sur les **systèmes de développement et de production**

5

Pourquoi utiliser RCAC?

Méthodes actuelles pour limiter l'accès aux données

- En définissant et en utilisant les **vues SQL**
- Les règles d'accès sont intégrées dans la **logique de l'application** → programmation nécessaire

Les restrictions d'accès peuvent être détournées

- En **accédant** aux **tables directement**
  - Avec SQL, Db2-WebQuery, Query/400, JDBC, ODBC native I/O, UPDDTA etc.
- Les utilisateurs avec **l'autorisation de l'objet** (par exemple *ALLOBJ) peuvent toujours voir **toutes les données**.

RCAC permet de contrôler l'accès aux toutes les données au niveau des lignes et des colonnes.

- **Indépendant** de la **méthode d'accès** est utilisée par exemple SQL, native I/O, CL, ODBC
- **Indépendant** de la logique **d'application** → **Transfert de la logique de gestion dans la base de données**
- Facilite la **multi-location (multi-tenancy)**
  - **Les données de plusieurs clients**/entreprises indépendants peuvent **être stockées dans la même table** sans qu'ils ne sachent les uns des autres
  - garantit que chaque utilisateur **ne voit que les données des lignes** et **colonnes** qu'il est **vraiment autorisé**

6

Rôles et séparation des responsabilités

7

Rôles et séparation des fonctions

Situation actuelle

- Les **rôles d'accès aux données** sont définis de **façon binaire,** c'est-à-dire **tout ou rien**
  - **Accès à TOUTES les données** d'une table (= object authority) ou
  - **AUCUN accès à AUCUNE donnée** de la table (= sans object authority)
- **Autorité d'accès COMPLET:** **Tous les profils utilisateur** associés à la **classe utilisateur** ***SECOFR**

  Tous les profils utilisateur avec **l'autorisation spéciale** ***ALLOBJ**

  → **AUCUNE EXCEPTION**

Malheureusement, cela pourrait **ne pas être conforme aux critères de l'organisation** en ce qui concerne **la restriction d'accès** aux données ou la **séparation des fonctions**

8

Rôles et séparation des fonctions

Function Usage IDs

- Permettre la réalisation de contrôles de sécurité détaillés
- Sans donner des autorisations spéciales puissantes aux utilisateurs

Function Usage ID - Commandes CL administratives

- **WRKFCNUSG** Travailler avec Function Usage
- **CHGFCNUSG** Modifier Function Usage
- **DSPFCNUSG** Afficher Function Usage

Nouveau Function Usage Ids

- **QIBM_DB_SECADM** Fonction d'administrateur de sécurité
- **QIBM_DB_SQLADM** Fonction d'administrateur de la base de données
- **QIBM_DB_SYSMON** Fonction d'information sur la base de données
- **QIBM_DB_ZDA** Accès au serveur d'applications Toolbox
- **QIBM_DB_DDMDRDA** Accès au serveur d'applications DDM et DRDA®

9

Rôles et séparation des fonctions – Commande CL WRKFNCUSG

```
                        Work with Function Usage

Type options, press Enter.
  2=Change usage   5=Display usage

Opt  Function ID                    Function Name
 _   QIBM_DIRSRV_ADMIN              IBM Tivoli Directory Server Administrator
 _   QIBM_ACCESS_ALLOBJ_JOBLOG      Access job log of *ALLOBJ job
 _   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                Database Security Administrator
 _   QIBM_DB_SQLADM                Database Administrator
 _   QIBM_DB_SYSMON                Database Information
 _   QIBM_DB_ZDA                   Toolbox Application Server Access
 _   QIBM_QYAS_SERVICE_DISKMGMT     Disk units
 _   QIBM_SERVICE_DISK_WATCHER     DISK WATCHER
 _   QIBM_SERVICE_DUMP             Service dump
                                                                    More...
Parameters for option 2 or command
===> _____
F3=Exit    F4=Prompt    F5=Refresh    F9=Retrieve    F12=Cancel    F17=Top
F18=Bottom
```

- **Nouveau Database Function Usage Ids**

10

Services IBM for Function Usage

Vue FUNCTION_INFO

- Détails sur les Function Usage Identifiers

```
Select Function_Id, Trim(Function_Name_Message_Text) Function_Name_Message_Text,
       Function_Type, Default_Usage, Function_Group_Id
  from Function_INFO a
  Where Function_Id like 'QIBM_DB%';
```

FUNCTION_ID	FUNCTION_NAME_MESSAGE_TEXT	FUNCTION_TYPE	DEFAULT_USAGE	FUNCTION_GROUP_ID
QIBM_DB	Database	GROUP	<NULL>	*NONE
QIBM_DB_SQLADM	Database Administrator	ADMINISTRABLE	DENIED	QIBM_DB
QIBM_DB_SYSMON	Database Information	ADMINISTRABLE	ALLOWED	QIBM_DB
QIBM_DB_SECADM	Database Security Administrator	ADMINISTRABLE	DENIED	QIBM_DB
QIBM_DB_DDMDRDA	DDM & DRDA Application Server Access	ADMINISTRABLE	ALLOWED	QIBM_DB
QIBM_DB_ZDA	Toolbox Application Server Access	ADMINISTRABLE	ALLOWED	QIBM_DB
QIBM_DB_GENCOL_OVERRIDE	Override Database Generated Values	ADMINISTRABLE	DENIED	QIBM_DB
QIBM_DB2_MIRROR	Db2 Mirror Administrator	ADMINISTRABLE	DENIED	QIBM_DB

Vue FUNCTION_USAGE

- Contient les détails de la Function Usage Configuration

```
Select * from Function_Usage
Where Function_Id like 'QIBM_DB_SECADM%';
```

FUNCTION_ID	USER_NAME	USAGE	USER_TYPE
QIBM_DB_SECADM	RPGPGM	ALLOWED	USER
QIBM_DB_SECADM	HAUSERPER	ALLOWED	USER

User Defined Function SQL_CHECK_FUNCTION_USAGE()

- Vérifier si l'utilisateur actuel est autorisé à utiliser le Function Usage Identifier spécifié
  - 0= L'utilisateur n'est pas autorisé / 1= L'utilisateur est autorisé

```
Values(qsys2.SQL_Check_Function_Usage('QIBM_DB_SECADM'));
```

00001
0

Rôles et séparation des fonctions QIBM_DB_SECADM

```
CHGFCNUSG FCNID(QIBM_DB_SECADM)
          USER(DBSECOFR)
          USAGE(*ALLOWED)
```

- **Enregistrer le profil utilisateur DBSECOFR pour qu'il puisse gérer les autorisations d'accès aux données**

Seuls le QSECOFR ou un utilisateur avec l'autorisation *SECADM

- peuvent **attribuer** le function usage **QIBM_DB_SECADM** à un utilisateur ou un group

Profil utilisateur enregistré pour QIBM_DB_SECADM peut

- Donner/révoquer des **autorisations**, modifier **la propriété (ownership)**, modifier le **groupe principal**
- Donner accès en **lecture aux tables Db2** à **d'autres utilisateurs**

→ Les utilisateurs (même) avec **l'autorisation *ALLOBJ peuvent être exclus** de **l'accès aux données** dans des tables particulières

- **Le profil utilisateur enregistré ne peut pas lire les données de quelque table Db2 que ce soit**

Afficher Function Usage

```
                          Display Function Usage

Function ID  . . . . . :   QIBM_DB_SECADM
Function name  . . . . :   Database Security Administrator

Description  . . . . . :   Database Security Administrator Functions


Product  . . . . . . . :   QIBM_BASE_OPERATING_SYSTEM
Group  . . . . . . . . :   QIBM_DB

Default authority  . . . . . . . . . . . . :   *DENIED
*ALLOBJ special authority  . . . . . . . . :   *NOTUSED

User        Type          Usage          User        Type          Usage
HAUSERPER   User          *ALLOWED
RPGPGM      User          *ALLOWED


F3=Exit    F12=Cancel    F17=Top    F18=Bottom
(C) COPYRIGHT IBM CORP. 1980, 2021.
```

- Les utilisateurs **HAUSERPER** et **RPGPGM** sont enregistrés pour le function usage **QIBM_DB_SECADM**

```
Select * from Function_Usage
Where Function_Id like 'QIBM_DB_SECADM%';
```

FUNCTION_ID	USER_NAME	USAGE	USER_TYPE
QIBM_DB_SECADM	RPGPGM	ALLOWED	USER
QIBM_DB_SECADM	HAUSERPER	ALLOWED	USER

13

Profil utilisateur enregistré pour la fonction QIBM_DB_SECADM

```
                    Change User Profile (CHGUSRPRF)

Type choices, press Enter.

User profile . . . . . > HAUSERPER     Name
User password  . . . . .   *SAME       Character value, *SAME, *NONE
Set password to expired    *NO         *SAME, *NO, *YES
Status . . . . . .         *ENABLED    *SAME, *ENABLED, *DISABLED
User class . . . . . .     *USER       *SAME, *USER, *SYSOPR...
Assistance level . . . .   *SYSVAL     *SAME, *SYSVAL, *BASIC...
Current library  . . . .   *CRTDFT     Name, *SAME, *CRTDFT
Initial program to call    *NONE       Name, *SAME, *NONE
  Library  . . . . . .                 Name, *LIBL, *CURLIB
Initial menu . . . . . .   MAIN        Name, *SAME, *SIGNOFF
  Library  . . . . . .       *LIBL     Name, *LIBL, *CURLIB
Limit capabilities . . .   *NO         *SAME, *NO, *PARTIAL, *YES
Text 'description' . . .   'Birgitta Hauser - FncUsr RCAC'
```

- Profil utilisateur HAUSERPER – enregistré pour QIBM_DB_SECADM
  - User Class: ***USER**
  - **Sans** aucune autorité particulière

```
F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this d
F24=More keys
```

```
                    Change User Profile (CHGUSRPRF)

Type choices, press Enter.

                          Additional Parameters

Special authority  . . . . . .   *NONE       *SAME, *USRCLS, *NONE...
                + for more values
Special environment  . . . . .   *SYSVAL     *SAME, *SYSVAL, *NONE, *S36
Display sign-on information  . .  *SYSVAL     *SAME, *NO, *YES, *SYSVAL
```

14

Registres spéciaux et fonctions scalaires

15

Registres spéciaux importants pour le RCAC

Registres spéciaux	Description	Définition
`User`	Profil utilisateur d'exécution qui détermine les autorisations de l'objet pour la connexion/le travail en cours	`VARCHAR(18)`
`Session_User`		`VARCHAR(128)`
`Current_User`	Profil utilisateur d'exécution qui détermine les autorisations de l'objet pour la connexion/le travail en cours.	
	Considère aussi les autorités adoptées	
	Programme/routine SQL créé avec **`USER=*OWNER`**	
	`Current_User` renvoie le profil **`*OWNER`** au moment de l'exécution.	
`System_User`	Profil utilisateur qui a établi la connexion au serveur	
	Prestarted Jobs : se connectent initialement au serveur avec un profil utilisateur par défaut, puis passent à un autre profil utilisateur	
	for example **`QUSER`** pour un job **`QZDASOINIT`**	

16

Fonction scalaire VERIFY_GROUP_FOR_USER

```
VERIFY_GROUP_FOR_USER (Current User,
                       User/Group Profiles)
```

Vérifier les autorisations d'accès de l'utilisateur actuel

- D'abord pour être utilisé avec les **RCAC Row Permissions** et **Column Masks**, mais peut aussi être utilisé dans **des autres instructions SQL**

Paramètres

- ***Current User:*** Registres spéciaux **SESSION_USER, USER** ou **CURRENT_USER**
- ***User/Group Profiles:*** un seul profil utilisateur ou une liste de profils ou profils Groupe

Renvoie un Integer-Value

- **1** = Le registre spécial **se trouve dans** la liste spécifiée des profils d'utilisateurs ou de groupes.
- **0** = Le registre spécial **ne se trouve pas dans** la liste spécifiée des profils d'utilisateurs ou de groupes.

Fonction scalaire VERIFY_GROUP_FOR_USER - Exemple

```
Values(Verify_Group_For_User(Current_User, 'QPGMR')),
      (Verify_Group_For_User(Current_User, 'Hauser', 'MEIER',   'SCHMIDT')),
      (Verify_Group_For_User(Current_User, 'Meier',  'QSYSOPR', 'Sales'))
```

00001
1
1
0

- **Current_User = 'HAUSER'**
- **Le profil utilisateur 'HAUSER' est membre du profil de groupe 'QPGMR'**

Fonction scalaire VERIFY_GROUP_FOR_USER - Exemple

```
Select s.*
  From Sales s
  Where CustNo = Case When Verify_Group_For_User(Current_User, 'HAUSERB') > 0
                      Then '10001'
                      When Verify_Group_For_User(Current_User, 'QPGMR') > 0
                      then '10002'
                      Else '10003' End
```

CUSTNO	ITEMNO	ITEM	SALESDATE	AMOUNT
10001	5100	King,Stephen - Es	2008-11-01	55,00
10001	5100	King,Stephen - Es	2008-12-23	60,00
10001	5200	King,Stephen - Drei	2009-01-30	160,00

- Current_User = **HAUSERB** → Vérifié explicitement
- Ne renvoie que des lignes pour **CUSTNO = 10001**

```
Select s.*
  From Sales s
  Where CustNo = Case When Verify_Group_For_User(Current_User, 'HAUSERB') > 0
                      Then '10001'
                      When Verify_Group_For_User(Current_User, 'QPGMR') > 0
                      then '10002'
                      Else '10003' End
```

- Current_User est **HAUSER**

CUSTNO	ITEMNO	ITEM	SALESDATE	AMOUNT
10002	5100	King,Stephen - Es	2008-11-15	1350,00
10002	5200	King,Stephen - Drei	2009-06-22	20,00
10002	5400	King,Stephen - Shining	2009-07-21	250,00

- Current_User = **HAUSER** → Membre de **QPGMR**
- Ne renvoie que des lignes pour **CUSTNO = 10002**

20

Vue: Vérifier Current_User à l'aide de VERIFY_GROUP_FOR_USER

```
Create or Replace View HSCOMMON10.SalesVRCAC
as Select s.*
      From Sales s
      Where CustNo = Case When Verify_Group_For_User(Current_User, 'HAUSERB') > 0
                          Then '10001'
                          When Verify_Group_For_User(Current_User, 'QPGMR')   > 0
                          then '10002'
                          Else '10003' End
```

```
Select Current_User, CustNo, Sum(Amount) Total
   from SalesVRCAC
   Group By Current_User, CustNo
```

00001	CUSTNO	TOTAL
HAUSERB	10001	3031,14

- Current_User: **HAUSERB**
- CUSTNO: **10001**

```
Select Current_User, CustNo, Sum(Amount) Total
   from SalesVRCAC
   Group By Current_User, CustNo
```

00001	CUSTNO	TOTAL
HAUSER	10002	2986,25

- Current_User: **HAUSER**
- Group Profile: **QPGMR**
- CUSTNO: **10002**

22

VERIFY_GROUP_FOR_USER Example

```
Select Current_User CurrUser, EmployeeNo, LastName, CostCenter,
       Case When Verify_Group_For_User(Current_User, 'HAUSERB') > 0
            Then Case When CostCenter = 344                Then Salary Else 0 End
            When Verify_Group_For_User(Current_User, 'OPGMR') > 0
            Then Case When CostCenter Between 100 and 199 Then Salary Else 0 End
            When Verify_Group_For_User(Current_User, 'HAUSERHR') > 0
            Then Salary
            Else 0
            End DspSalary
  From comrcac.EmployeeCpy a;
```

CURRUSER	EMPLOYEENO	LASTNAME	COSTCENTER	DSPSALARY
HAUSERB	1000	Fischer	344	55000,00
HAUSERB	1010	Meier	344	70000,00
HAUSERB	1020	Bauer	344	55000,00
HAUSERB	2000	Schmidt	100	0,00
HAUSERB	2100	Gerber	111	0,00
HAUSERB	1030	Moser	344	30000,00

CURRUSER	EMPLOYEENO	LASTNAME	COSTCENTER	DSPSALARY
HAUSER	1000	Fischer	344	0,00
HAUSER	1010	Meier	344	0,00
HAUSER	1020	Bauer	344	0,00
HAUSER	2000	Schmidt	100	150000,00
HAUSER	2100	Gerber	111	45000,00
HAUSER	1030	Moser	344	0,00

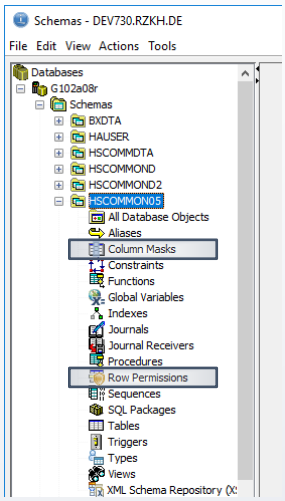
CURRUSER	EMPLOYEENO	LASTNAME	COSTCENTER	DSPSALARY
HAUSERHR	1000	Fischer	344	55000,00
HAUSERHR	1010	Meier	344	70000,00
HAUSERHR	1020	Bauer	344	55000,00
HAUSERHR	2000	Schmidt	100	150000,00
HAUSERHR	2100	Gerber	111	45000,00
HAUSERHR	1030	Moser	344	30000,00

CURRUSER	EMPLOYEENO	LASTNAME	COSTCENTER	DSPSALARY
HAUSERPER	1000	Fischer	344	0,00
HAUSERPER	1010	Meier	344	0,00
HAUSERPER	1020	Bauer	344	0,00
HAUSERPER	2000	Schmidt	100	0,00
HAUSERPER	1030	Moser	344	0,00
HAUSERPER	2100	Gerber	111	0,00

23

24

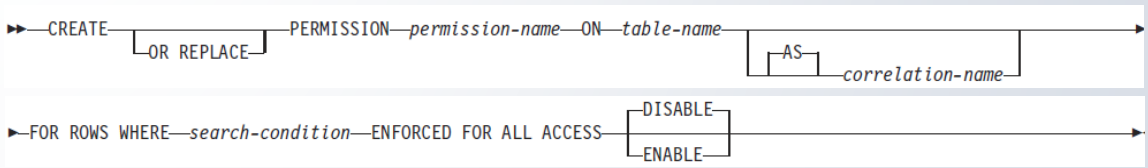
Access Client Solutions – Schéma - Row and Column Access Control (RCAC)



- **Column Masks**
- **Row Permission**

25

Row Permission - Create Permission Statement



CREATE PERMISSION = Définir Row Permission

- *Permission-Name:* **Nom** de la RCAC **row permission**
- *Table-Name:* **Table** sur laquelle la row permission est créée
- FOR ROWS: **Row** Access Control
- *Seach Condition:* **Conditions WHERE** qui renvoient **VRAI, FAUX** ou **INCONNU**
- ENFORCED FOR ALL ACCESS → **ENABLE/DISABLE:**
  Si la Permission est **initialement activée** ou **pas**

26

Alter Table – Pour activer le Row and Column Access Control

Une Row Permission n'est PAS activée avec sa création

- Même si <<**ENFORCED FOR ALL ACCESS>> est activé (enabled)**

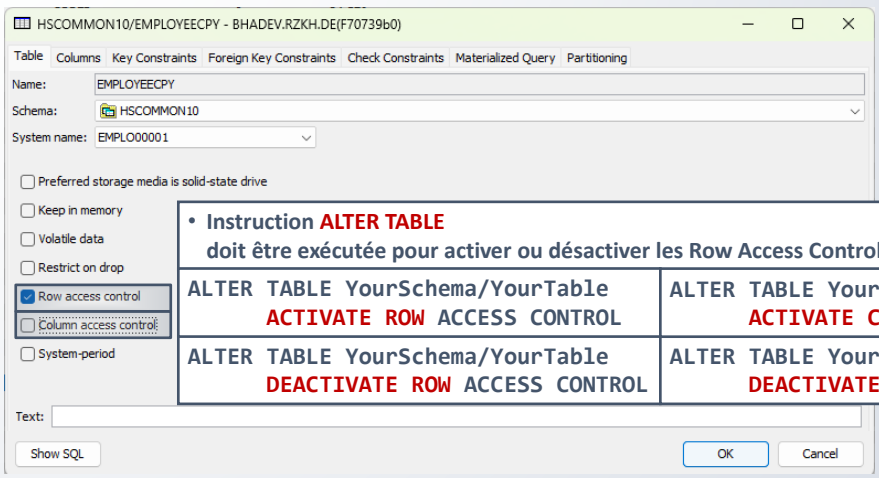
Une Row Permission doit être explicitement activée pour la table

- Pour **activer explicitement** une Row Permission, une instruction **ALTER TABLE** avec **ACTIVATE ROW ACCESS CONTROL** est nécessaire

```
ALTER TABLE HSCOMMON10.MYADDRESS
          ACTIVATE ROW ACCESS CONTROL ;
```


27

Alter Table - Activer - Row and Column Access Control



- **Instruction ALTER TABLE**
  doit être exécutée pour activer ou désactiver les Row Access Control et/ou Column Access Control

`ALTER TABLE YourSchema/YourTable ACTIVATE ROW ACCESS CONTROL`	`ALTER TABLE YourSchema/YourTable ACTIVATE COLUMN ACCESS CONTROL`
`ALTER TABLE YourSchema/YourTable DEACTIVATE ROW ACCESS CONTROL`	`ALTER TABLE YourSchema/YourTable DEACTIVATE COLUMN ACCESS CONTROL`

28

Accès aux données avec Row Permission - Exemple

```
CREATE OR REPLACE PERMISSION HSCOMMON10.MYADDRESS_PERMCUSTNO ON HSCOMMON10.MYADDRESS AS MACUSTNO
    FOR ROWS
        WHERE Substr(CustNo, 1, 1)
                between case When Verify_Group_for_User(Session_User, 'HAUSERB') = 1 Then '0'
                             When Verify_Group_for_User(Session_User, 'QPGMR')   = 1 Then '5'
                        End
                    and case When Verify_Group_for_User(Session_User, 'HAUSERB') = 1 Then '4'
                             When Verify_Group_for_User(Session_User, 'QPGMR')   = 1 Then '7'
                        End
    ENFORCED FOR ALL ACCESS
    ENABLE  ;
```

- Session_User = **HAUSERB**
- Peut accéder à tous les no client commençant avec **0,1,2,3,4**

```
Select * from HSCOMMON10.MyAddress
```

CUSTID	CUSTNO	CUSTNAME1
45	00100	Pallhuber und Söhne
46	00110	Bahnleitner Gemischtwaren
47	00120	Ebäcko Nordrhein Westfalen
48	00130	Deutscher-Paket-Dienst

- Session_User = **HAUSER**
- Membre du profil group **QPGMR**
- Peut accéder à tous les no client commençant avec **5,6,7**

```
Select * from HSCOMMON10.MyAddress
```

CUSTID	CUSTNO	CUSTNAME1
31	56453	GWINNER WOHNDESIGN GMBH
23	63820	FIRMA MAYER GMBH
29	63899	HELLSTERN GMBH
33	66215	GÜNTHER NETZER GMBH
36	66588	ARBURG GMBH

29

Column Masks

30

Permission de colonne – Instructions Grant / Revoke

Instruction GRANT

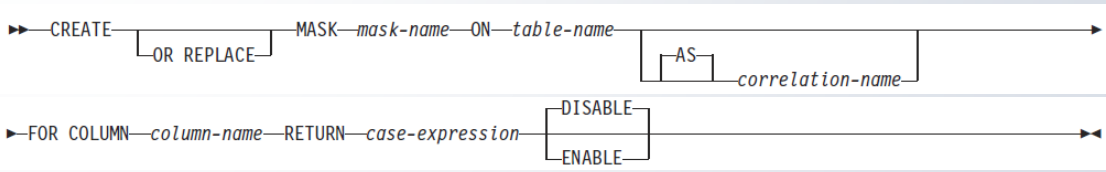
- donne des **autorisations** sur des **tables** ou des **vues** à un **utilisateur** spécifique, à un **profil de groupe** ou au **PUBLIC** (valeur spéciale)
- Attribue le privilège de **modifier seulement les colonnes explicitement listées** dans **l'instruction GRANT**
  - Les autres colonnes de la table ou de la vue qui **ne sont pas explicitement listées ne peuvent pas être modifiées** par l'utilisateur spécifié, le profil de groupe ou **PUBLIC**

```
GRANT SELECT ,
      UPDATE ( BIRTHDAY   , CITY   , COSTCENTER , COUNTRY   ,
               DEPARTMENT , EMAIL  , EMPLOYEENO , EXITDATE ,
               FIRSTNAME  , "ID"   , JOBSPEC    , LASTNAME ,
               MOBILE     , PHONE  , STREET     , TITLE    ,
               ZIPCODE )
   ON HSCOMMON10.EMPLOYEE
   TO HAUSERB ;
```

- **Profil utilisateur HAUSERB peut**
  - **Lire les données de la table EMPLOYEE**
  - **Modifier seulement les colonnes listées**

31

Column Masks - Create Mask Statement



CREATE MASK – Masquer le contenu des colonnes

- *Mask-Name* **Nom du masque** pour column access control
- *Table-Name* **Table** sur laquelle le column mask est créée
- *Correlation-Name* **Nom de corrélation** facultatif pouvant être utilisé dans **l'expression case**
- FOR COLUMN *Column-Name* **Colonne** à laquelle le masque s'applique
- RETURN *Case-Expression* **Case-Expression** à évaluer

  Les **types de données** du résultat/du masque et de la colonne doivent être compatibles
- Enable/Disable Si le masque est **initialement activé (enable)** ou **pas (disable)**

32

Alter Table - Activate Row and Column Access Control

Un Column Mask n'est PAS activé à sa création

- Même si **ENABLED** est spécifié (enabled)

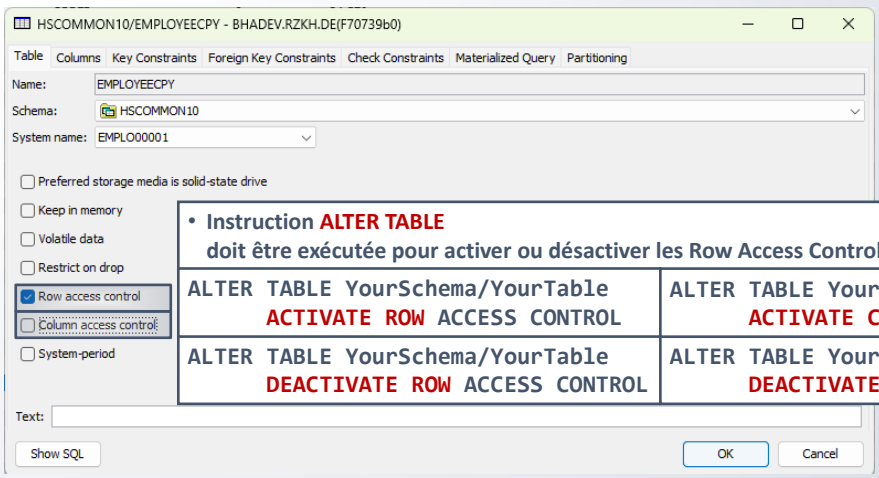
Un Column Mask doit être activé sur la table

- Pour **activer explicitement** un Column Mask, une instruction **ALTER TABLE** avec **ACTIVATE COLUMN ACCESS CONTROL** est nécessaire

```
ALTER TABLE HSCOMMON10.EMPLOYEE
            ACTIVATE COLUMN ACCESS CONTROL ;
```

33

Alter Table - Activer - Row and Column Access Control



- **Instruction ALTER TABLE**
  doit être exécutée pour activer ou désactiver les Row Access Control et/ou Column Access Control

`ALTER TABLE YourSchema/YourTable ACTIVATE ROW ACCESS CONTROL`	`ALTER TABLE YourSchema/YourTable ACTIVATE COLUMN ACCESS CONTROL`
`ALTER TABLE YourSchema/YourTable DEACTIVATE ROW ACCESS CONTROL`	`ALTER TABLE YourSchema/YourTable DEACTIVATE COLUMN ACCESS CONTROL`

34

Accès aux données avec Column Masks Exemple

```
CREATE or Replace MASK COMRCAC.PEREMPLSALARY
        ON COMRCAC.EMPLOYEE
        FOR COLUMN SALARY
  RETURN Case When Verify_Group_For_User(Session_User, 'HAUSERB')    = 1
              Then Case When salary < 100000 Then Salary Else -999 End
              When   Verify_Group_For_User(Session_User, 'QPGMR')    = 1
                 and CostCenter = 344       Then Salary
              When   Verify_Group_For_User(Session_User, 'HAUSERHR') = 1
              Then Salary Else -999 End
   ENABLE;
```

```
Select Id, EmployeeNo, CostCenter,
       Salary, LastName, FirstName
  from Employee
  Order By Salary Desc
```

ID	EMPLOYEENO	COSTCENTER	SALARY
4	2000	100	-999,00
2	1010	344	70000,00
1	1000	344	55000,00
3	1020	344	55000,00
5	2100	111	45000,00
6	1030	344	30000,00

- Session_User = **HAUSERB**
- Peut voir tous les salaires **moins de 100000**

```
Select Id, EmployeeNo, CostCenter,
       Salary, LastName, FirstName
  from Employee
  Order By Salary Desc
```

ID	EMPLOYEENO	COSTCENTER	SALARY
4	2000	100	-999,00
2	1010	344	70000,00
1	1000	344	55000,00
3	1020	344	55000,00
5	2100	111	-999,00
6	1030	344	30000,00

- Session_User = **HAUSER**
- Member du profil du groupe **QPGMR**
- ne voit que les salaires du **cost center 344**

35

Requêtes SQL, Totaux et Column Masks

Requêtes et fonctions d'agrégation avec Column Masks

- Valeurs des colonnes dépendant des autorisations d'accès de l'utilisateur

- Valeur du Masque = -9999

```
Select Sum(Salary) Total
  From Employee;
```

TOTAL
208002,00

- Session_User = **HAUSER**
- Membre du profil de group **QPGMR**
- ne peux voir que les salaires du centre de coûts **344**

```
Select Sum(Salary) Total
  From Employee;
```

TOTAL
254001,00

- Session_User = **HAUSERB**
- Ne peut accéder qu'aux salaires moins de 100000 Euros

```
Select Sum(Salary) Total
  From Employee;
```

TOTAL
-5994,00

- Session_User = **HAUSERPER**
- ne vois aucun salaire

36

Table des Employés et Cartes de crédit- Exemple

```
Select EmployeeNo, EMCCID "CredCard Id",
       LastName, FirstName, Street, ZipCode, City
from EmplCred
Order By EMCCID;
```

- Table des Employés

EMPLOYEENO	CredCard Id	LASTNAME	FIRSTNAME	STREET	ZIPCODE	CITY
2100	2	Gerber	Kim	Am Bach 3	85051	Ingolstadt
1000	5	Fischer	Fritz	Oberfeldstr. 16	76149	Karlsruhe
1010	11	Meier	Anna	Frankfurter Str. 55	63128	Dietzenbach
1020	15	Bauer	Stefan	An der Havel 234	10785	Berlin
1030	17	Moser	Ben	Waldstr 1	77880	Sasbach
2000	19	Schmidt	Anton	Seestr. 7	17192	Waren/Mueritz

```
Select CCID "CredCard Id", CredCardNo, CVV,
       ExpirYear, ExpirMonth
  from CREDCARD;
```

- Table des Cartes de Crédit

CardId	CREDCARDNO	CVV	EXPIRYEAR	EXPIRMONTH
1	6011276617038831	765	2022	3
2	5140103430432676	221	2022	9
3	5294479349873539	396	2021	5
4	5302985801465861	324	2022	9
5	5487796361329240	323	2024	9
6	4013987007326163	373	2021	1
7	4042051746869564	614	2021	7
8	4047043193104590	207	2021	2
9	4070199796656947	793	2021	12
10	4092983562516516	460	2022	7
11	4108554684486359	210	2024	11
12	4131077380803421	833	2022	10
13	4134875449544952	376	2021	10
14	4500484439016885	602	2022	3
15	4502245604298774	263	2024	3
16	4508969691406214	586	2021	7
17	4529749847757685	127	2023	3
18	4539759236009026	716	2021	5
19	4563063840841158	503	2024	11

- Table des Employés avec l'Id de la Table des Cartes de Crédit
- Le masquage (partiel) du numéro de carte de crédit dépend des informations de la table des employés (centre de coûts).

37

Table des Employés et Cartes de crédit- Exemple

```
Create Or Replace Mask COMRCAC.COLM_CREDCARD_CREDCARDNO
    On COMRCAC.CREDCARD For Column CREDCARDNO
       Return Case When     Verify_Group_For_User(Session_User, 'HAUSERHR') = 1
                   Then CredCardNo
                   When     Verify_Group_For_User(Session_User, 'HAUSERB')  = 1
                        and (Select CostCenter
                             From EmplCred
                             Where EMCCID = CCID
                             Fetch First Row Only) between 100 and 200
                   Then CredCardNo
              When     Verify_Group_For_User(Session_User, 'QPGMR') = 1
                   and (Select CostCenter
                        from EmplCred
                        Where emccid = CCID
                              Fetch First Row Only) = 344
                   Then CredCardNo
                   Else Repeat('*', Length(Trim(CredCardNo)) - 4) concat Right(Trim(CredCardno), 4)
              End

       Enable;
```

- Le centre de coûts de la table des employés est vérifié
- Les 4 derniers chiffres du numéro de carte de crédit sont toujours affichés.

```
ALTER TABLE COMRCAC.CREDCARD
            ACTIVATE COLUMN ACCESS CONTROL ;
```

- Activer Column Access Control

38

Table des Employés et Cartes de crédit- Exemple

```
Select CCID "CredCard Id", CredCardNo, CVV,
       ExpirYear, ExpirMonth
   from CREDCARD;
```

CredCard Id	CREDCARDNO	CVV	EXPIRYEAR	EXPIRMONTH
1	**********8831	765	2022	3
2	**********2676	221	2022	9
3	**********3539	396	2021	5
4	**********5861	324	2022	9
5	5487796361329240	323	2024	9
6	**********6163	373	2021	1
7	**********9564	614	2021	7
8	**********4590	207	2021	2
9	**********6947	793	2021	12
10	**********6516	460	2022	7
11	4108554684486359	210	2024	11
12	**********3421	833	2022	10
13	**********4952	376	2021	10
14	**********6885	602	2022	3
15	4502245604298774	263	2024	3
16	**********6214	586	2021	7
17	4529749847757685	127	2023	3
18	**********9026	716	2021	5
19	**********1158	503	2024	11

Update avec Column Masks

Que se passe-t-il si une ligne complète est modifiée mais que le buffer de ligne contient des valeurs masquées ?

- Exemple: L'adresse postale doit être modifiées dans la table des Employés.
  La date de naissance est une autre colonne dans la même table.
  Il y a un **column mask** sur la colonne de la date de naissance.
  L'utilisateur qui doit modifier l'adresse postale n'est **pas autorisé** à voir la date de la naissance, mais **voit** à place **le masque**.
  Lorsqu'une ligne est modifiée avec **Native I/O, le Buffer de ligne** ne contient **pas la valeur d'origine** mais **la valeur masquée**

Comment conserver la valeur originale?

- Ajouter une **contrainte de vérification (check)** avec la clause **ON VIOLATION**
- Ajouter un **SECURED Before Insert/Update Trigger**

Vérifier la contrainte avec la clause ON VIOLATION

```
Alter Table EMPLOYEE
      Add Constraint ChkCst_Employee_Birthday
          Check(BIRTHDAY > '0001-01-01')
          On Insert Violation Set      BIRTHDAY = Default
          On Update Violation Preserve BIRTHDAY;
```

- **ON INSERT VIOLATION**
- **Si une valeur masquée est passée, elle est remplacée par la valeur Défault**

- **ON UPDATE VIOLATION**
- **Si une valeur masquée est passée, la valeur originale est préservée**

Attribut SECURED

SECURED doit être indiqué pour des Vues, Fonctions, Triggers

- Un **trigger** ou une **fonction SECURED** est considérée comme **sécurisée** pour RCAC
  - Doit être spécifié pour un **trigger** basé sur **une table avec RCAC**
  - Doit être spécifié pour un **trigger Instead Of** créé pour **une vue** où une ou plusieurs **tables utilisant RCAC**
  - Doit être spécifié lorsqu'**une fonction** est référencée dans un **RCAC Column Mask**

Before Trigger avec l'attribut SECURED

```
Create or Replace Trigger HSCOMMON10.REPLACE_MASK_BIRTHDAY
       Before Insert Or Update On HSCOMMON10.EMPLOYEE
       Referencing New Row as N
                   Old Row as O
       For Each Row
       Mode DB2ROW
       Secured
       When (N.BIRTHDAY = '0001-01-01')
   Begin
     If     Inserting Then Set N.BIRTHDAY = Default;
     ElseIf Updating  Then Set N.BIRTHDAY = o.BIRTHDAY;
     End If;
   End;
```

- SECURED Before Insert/Update Trigger pour préserver les valeurs originales
- Si une valeur masquée est passée lors de l'insertion, elle est remplacée par la valeur par défaut.
- Si une valeur masquée es passée lors de la modification, la valeur précédente est préservée

43

RCAC et Native I/O – Exemple - sans Column Mask

```
DSPLY  Programming   344 Bauer      Stefan      70000.00
DSPLY  Programming   344 Fischer    Fritz       55000.00
DSPLY  Programming   344 Meier      Anna       105000.00
DSPLY  Programming   344 Moser      Ben         30000.00
DSPLY  Programming   344                       260000.00
DSPLY  Programming                             260000.00
DSPLY  Sales         100 Schmidt    Anton      150000.00
DSPLY  Sales         100                       150000.00
DSPLY  Sales         111 Gerber     Kim         45000.00
DSPLY  Sales         111                        45000.00
DSPLY                                          455000.00
```

- Program Salary02:
  - Accumuler le salaire annuel par centre de coûts et par département

44

Ajouter des Column Masks pour le salaire et la date de naissance

```
CREATE or Replace MASK COMRCAC.COLM_Employee_Salary
         ON COMRCAC.EMPLOYEE
         FOR COLUMN SALARY
  RETURN Case When Verify_Group_For_User(Session_User, 'HAUSERHR')    = 1
              Then Salary
              When Verify_Group_For_User(Session_User, 'HAUSERB')     = 1
              Then Case When salary < 100000 Then Salary Else -999 End
              When   Verify_Group_For_User(Session_User, 'QPGMR')     = 1
                   and CostCenter = 344       Then Salary
             Else -999 End
   ENABLE;
```

- **Attention: Si les valeurs par défaut numériques sont définies sur une valeur différente de *Zeros!!!**

```
            Create Or Replace Mask COMRCAC.COLM_EMPLOYEE_BIRTHDAY
                On COMRCAC.EMPLOYEE For Column BIRTHDAY
                   Return Case When     Verify_Group_For_User(Session_User, 'QPGMR') = 1
                                   And COSTCENTER = 344
                               Then BIRTHDAY
                               When Verify_Group_For_User(Session_User, 'HAUSERHR') = 1
                               Then BIRTHDAY
                               Else '0001-01-01' End
                    Enable;
```


RCAC et Native I/O – Avec Column Mask - Exemple

DSPLY	Programming	344	Bauer	Stefan	70000.00	70000.00	70000.00
DSPLY	Programming	344	Fischer	Fritz	55000.00	55000.00	55000.00
DSPLY	Programming	344	Meier	Anna	105000.00	-999.00	105000.00
DSPLY	Programming	344	Moser	Ben	30000.00	30000.00	30000.00
DSPLY	Programming	344			260000.00	154001.00	260000.00
DSPLY	Programming				260000.00	154001.00	260000.00
DSPLY	Sales	100	Schmidt	Anton	-999.00	-999.00	150000.00
DSPLY	Sales	100			-999.00	-999.00	150000.00
DSPLY	Sales	111	Gerber	Kim	-999.00	45000.00	45000.00
DSPLY	Sales	111			-999.00	45000.00	45000.00
DSPLY					258002.00	198002.00	455000.00

- **Le programme est exécuté par différents utilisateurs → Totaux Différentes**
- **HAUSER:** Masquer les salaires de tous les employés qui ne sont pas dans le centre de coûts 344
- **HAUSERB:** Masquer tous les salaires plus de 100 000 Euros
- **HAUSERHR:** voit tous les salaires

Ajouter Check Constraints pour la date de naissance et le salaire

```
Alter Table EMPLOYEE
      Add Constraint ChkCst_Employee_Birthday
          Check(BIRTHDAY > '0001-01-01')
          On Insert Violation Set       BIRTHDAY = Default
          On Update Violation Preserve BIRTHDAY;
```

```
Alter Table EMPLOYEE
     Add Constraint ChkCst_Employee_Salary
         Check(Salary >= 0)
         On Insert Violation Set       Salary = Default
         On Update Violation Preserve Salary;
```

- Sans Check Constraint avec violation ON INSERT et ON UPDATE ou un BEFORE INSERT OR UPDATE trigger approprié, **les valeurs masquées sont écrites**!!!

RCAC et Native I/O – Avec Column Mask Update Exemple

- Update Record avec Native I/O (inclut toutes les colonnes)

```
DSPLY  User: HAUSER
DSPLY  Fischer Fritz 86916 Kaufering
DSPLY  CostCenter:344 Birthday: 1958-05-15 Salary: 55000.0
DSPLY  Fischer Fritz 76149 Karlsruhe
DSPLY  CostCenter:344 Birthday: 1958-05-15 Salary: 55000.0
DSPLY  Schmidt Anton 63303 Langen
DSPLY  CostCenter:100 Birthday: 0001-01-01 Salary: -999.00
DSPLY  Schmidt Anton 17192 Waren/Mueritz
DSPLY  CostCenter:100 Birthday: 0001-01-01 Salary: -999.00
```

- **HAUSER:**
  ne peux voir que les anniversaires et les salaires des employés du centre de coûts 344

```
DSPLY  User: HAUSERHR
DSPLY  Fischer Fritz 86916 Kaufering
DSPLY  CostCenter:344 Birthday: 1958-05-15 Salary: 55000.0
DSPLY  Fischer Fritz 76149 Karlsruhe
DSPLY  CostCenter:344 Birthday: 1958-05-15 Salary: 55000.0
DSPLY  Schmidt Anton 63303 Langen
DSPLY  CostCenter:100 Birthday: 1965-01-31 Salary: 150000.
DSPLY  Schmidt Anton 17192 Waren/Mueritz
DSPLY  CostCenter:100 Birthday: 1965-01-31 Salary: 150000.
```

- **HAUSERHR:**
  Peut voir et modifier les dates d'anniversaire et les salaires de tous les employés

Restrictions

Fichiers pour lesquels RCAC ne peut pas être utilisé

- Fichiers distribués
- Fichiers décrits en interne/dans le programme
- Fichiers logiques multiformats
- Fichiers avec ICU 2.6.1 Séquence de tris
- Fichiers ou Tables avec READ triggers

L'accès aux données doit être exécuté avec le SQE (SQL Query Engine)

- Depuis Release 7.4 tout accès aux données même avec des interfaces non-SQL doit être exécuté avec le SQE
  - Non-SQL Interfaces: Native I/O (RPG or Cobol), Query400, OPNQRYF, RUNQRY, QQQQRY API

RCAC active – Copier des données avec la commande CL CPYF

Copier des données avec la commande CL CPYF (Copy File)

- **Seules** les données auxquelles **l'utilisateur a accès** sont copiées
  - Les données **ne sont pas toujours complètement copiées**
  - Si des **column masks** sont inclus, les valeurs **masquées** sont copiées!
- Si une **nouvelle table** est générée, les définitions **RCAC ne sont pas copiées**
- Si les tables « De » et « À » ont des **définitions RCAC différentes**, cela peut **provoquer une erreur**
- **Attention:** Des **implémentations RCAC incorrectes** peuvent provoquer des **interruptions du programme** ou, encore pire, les données sont **copiées incomplètement sans aucun message d'erreur**

Considérations supplémentaires

Transfert de données d'une table avec RCAC vers une table sans RCAC

- Seules les lignes auxquelles **l'utilisateur est autorisé** sont transférées
- Les valeurs **masquées** sont **transférées**

• **Possibilité de perte de données**

Transfert de données d'une table sans RCAC vers une table avec RCAC

- Seules les données auxquelles **l'utilisateur est autorisé** sont transférées

• **Les données peuvent être copiées incomplètement**

Définitions différentes du RCAC dans les deux tables

- Seules les lignes que **l'utilisateur est autorisé** peuvent être **insérées** dans la nouvelle table
- Les valeurs **masquées** sont **refusées**

• **Le transfert peut échouer**

RCAC actif – Créer un objet dupliqué avec la commande CL CRTDUPOBJ

Créer un objet dupliqué avec la commande CL CRTDUPOBJ

- **Nouvelle Option:** (Duplicate access control)
  - ***ALL** Toutes les définitions RCAC sont copiées dans la nouvelle table(= Default)
  - ***ROW** Seule les Row Permissions sont copiées
  - ***COL** Seule les Column Masks sont copiées
  - ***NONE** Aucune définition RCAC n'est copiée
- Option **DATA = *YES**
  - **All data** Toutes les données (les lignes et valeurs de colonnes) sont copiées
  - **DATA** = ***YES** → **ACCCTL** doit être fixé à ***ALL** sinon, le double ne peut pas être créé

Attention: Avec **CPYF**, seules **les données auxquelles l'utilisateur a accès** sont copiées
Avec **CRTDUPOBJ**, **l'objet complet** avec **toutes les données et règles** est copié

CRTDUPOBJ – Create duplicate Object - Enhancement

```
                    Create Duplicate Object (CRTDUPOBJ)

Type choices, press Enter.

From object  . . . . . > EMPLOYEE     Name, generic*, *ALL
From library . . . . . > COMRCAC      Name, *LIBL, *CURLIB
Object type  . . . . . > *FILE        *ALL, *ALRTBL, *AUTL...
             + for more values
To library . . . . . .   *FROMLIB     Name, *FROMLIB, *SAME...
New object . . . . . .   *OBJ         Name, *OBJ, *SAME
From ASP device  . . . .   *          Name, *, *CURASPGRP, *SYSBAS
To ASP device  . . . . .   *ASPDEV     Name, *ASPDEV, *...
Duplicate data . . . . .   *Yes        *NO, *YES
Duplicate constraints  .   *YES       *YES, *NO
Duplicate triggers . . .   *YES       *YES, *NO
Duplicate file identifiers .   *NO    *NO, *YES
Duplicate access control . .   *ALL    *ALL, *ROW, *COL, *NONE

F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
F24=More keys
```

- Si Duplicate Data (Données en double) est défini sur *YES
  Duplicate Access Control (ACCCTL) doit être défini sur *ALL

Considérations supplémentaires

RCAC Timing

- Le masquage des colonnes s'effectue **après le traitement complet** de **TOUTES les requêtes**
- Local Selection, Joins, Group Bys and Sorts **basé** sur les **valeurs masquées based**

RCAC et Field Procedures

- Field Procedure Masking a lieu lorsque les valeurs des colonnes **sont lues/écrites**

  → **Avant Query Processing** au contraire au RCAC masquage de colonne

RCAC et Transfert des données

- Pour les processus de transfert de données (copie/ sauvegarde), **autorisation d'accéder à TOUTES les données sans aucun masquage**

→ **Essentiel! Doit être soigneusement conçu et planifié.**

RCAC et Journal Receivers

- RCAC **n'est pas appliqué** à l'accès **Journal Receiver Access,**

  → c'est-à-dire que **toutes les transactions indépendantes du RCAC** se trouvent dans les journal receivers

Des questions?

56

Biographie brève: Birgitta Hauser

Birgitta Hauser
Diplom-Betriebswirt (BA)
Database and Software Architect

Diplômée en gestion d'entreprise, Birgitta Hauser a d'abord travaillé plusieurs années dans le contrôle de gestion avant de se tourner vers la programmation (RPG) sur AS/400. Aujourd'hui, elle travaille encore quelque fois comme programmeur sur l' IBM i. Son travail se concentre toutefois sur la modernisation et l'optimisation des applications IBM i existantes, en particulier des bases de données, ainsi que sur l'intégration de nouvelles technologies.

Depuis 2020, Birgitta travaille à son compte et assiste ses clients dans des projets de modernisation d'applications et de bases de données et de l'optimisation des performances SQL sur l'IBM i et Db2 for i.

De plus, Birgitta donne régulièrement des cours pour des programmeurs IBM i (RPG/CL) et des spécialiste Db2 for i et des utilisateurs SQL.

Depuis 2002, Birgitta intervient régulièrement lors de conférences des COMMON User Groups en Allemagne, dans d'autres pays européens, ainsi qu'aux États-Unis et au Canada.

Birgitta est co-auteur de 2 IBM Redbooks, ainsi que de plusieurs articles spécialisés pour IBM DeveloperWorks and IT-Jungle. Elle écrit régulièrement des articles spécialisés (RPG/SQL) pour le ITP-Verlag (daison d'édition allemande)

En 2015, Birgitta a reçu la bourse d'études John Earl Speaker. En 2018, elle a reçu la bourse d'études commémorative Al Barsa.

Depuis 2020 elle est un IBM Champion.

57

Un grand merci à

Holger Scherer – RZKH Rechenzentrum Kreuznach

- Pour avoir offert un système IBM i permettant la création des exemples et du code utilisés dans mes présentations.
- http://www.rzkh.de

Your data is save! … in the bunker

58

Merci!

Sécurisez vos données avec Row and Column Access Control (RCAC)

Yes, i can!

Si vous êtes intéressé par des classes individuelles plus détaillées, sur place ou à distance, veuillez me contacter directement

Birgitta Hauser – Modernization – Education – Consulting on IBM i
Diplom-Betriebswirt (BA)
Database and Software Architetct
IBM Champion seit 2020
eMail: Hauser@ModEdCon.com / Hauser@SSS-Software.de
Web: https://modedcon.com/

59