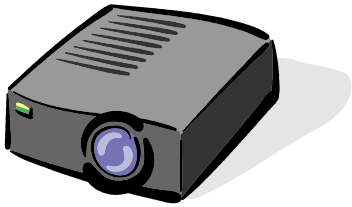
Modernisation et développement d'applications IBM i
Stratégies, technologies et outils

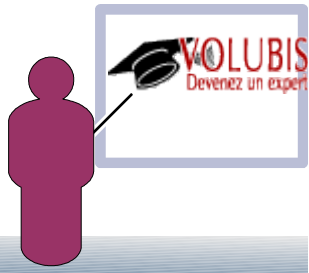
16 et 17 mai 2011 – IBM Forum de Bois-Colombes



Volubis.fr

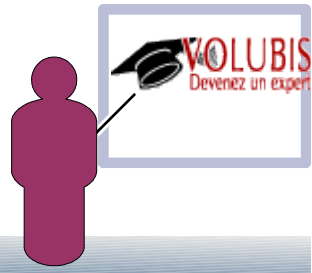
Conseil et formation sur OS/400, I5/OS puis IBM ***i***
depuis 1994 !

Christian Massé - cmasse@volubis.fr



Nouveaux types de données

- Attributs et Types de données récents
  - Gestion de la valeur nulle
  - Dates / Heures /Horodatage
  - Champs auto-incrémentés(Rowid / identity)
  - Blob vs DataLink
  - XML

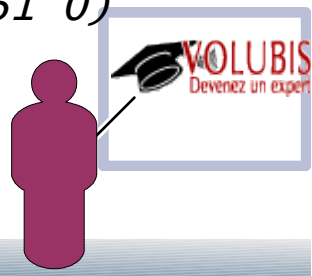


Nouveaux types de données

- Valeur nulle
  - Permet de faire la différence entre « rien » et 0
    - Important avec les dates
    - NULL sous SQL
    - IS NULL / IS NOT NULL lors des tests
    - IFNULL pour remplacer « NULL » par une valeur
    - Variable indicateur pour embeded SQL

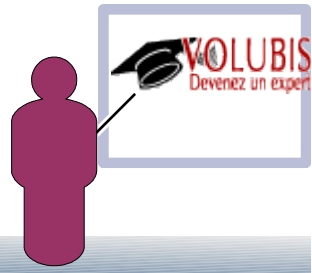
```
select datfin into :datfin:datfinI …
 datfinI étant une variable intéger 2 o.(5I 0)
```

    - %NULLIND en RPG



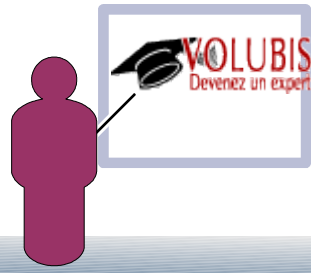
Nouveaux types de données

- Dates / Heures / Horodatage
  - Le seul moyen d'avoir un accès simple à la puissance de calcul des langages RPG et SQL
    - Vérification de la date lors des insertions
    - Supportées par les DSPF (contrôle inclus)
      - Reconnaît le format *JOB qui prend le format en cours et transmet de l'ISO au programme
    - Le format n'est qu'un problème d'affichage :
      - Mettez vos dates en ***ISO** partout sauf au moment de les afficher aux utilisateurs



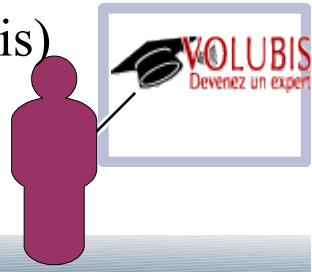
Nouveaux types de données

- Calculs sous SQL
  - Mots-réservés
    - Current date, current time, current timestamp ou now()
  - Opérateurs + et – avec une durée :
    `datcmd + 3 months + 10 days`
  - Opérateur – produisant une durée :
    current date – `datliv → 10215`
    - `se lit aaaammjj soit 1 an 2 mois 15 jours`



Nouveaux types de données

- Fonctions SQL 1/2
  - DAYOFWEEK(date) = jour dans la semaine (1=dimanche)
  - DAYOFWEEK_ISO(date) = jour dans la semaine (1=Lundi)
  - DAYOFYEAR(date) = jour (julien) dans l'année.
  - QUARTER(date) = N° du trimestre
  - WEEK(date) = N° de la semaine ! 1er Janvier = semaine 1
  - WEEK_ISO(date) = N° de la semaine, 1er janvier = 1 ou 53.
  - DAYNAME(Date) retourne le nom du jour (en Français)
  - MONTHNAME(Date) retourne le nom du mois (en Français)

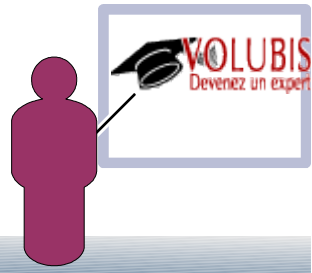


Nouveaux types de données

- Fonctions SQL 2/2
  - LAST_DAY(date) retourne la date correspondant au dernier jour du mois
  - ADD_MONTHS(date, nbr) ajoute un nombre de mois à la date

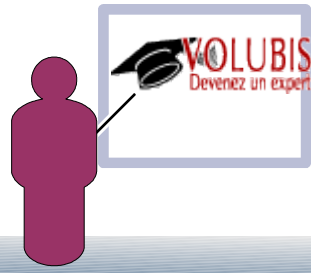
    (si la source est une fin de mois, le résultat aussi)
  - NEXT_DAY(date , 'JOUR')

    ```
    NEXT_DAY('2011-05-17' , 'Dim') → 2011-05-22
    ```
  - EXTRACT(DAY from Date) extrait la partie jour de la date
    possible aussi avec MONTH, YEAR, HOUR, MINUTE, SECOND



Nouveaux types de données

- Calculs en RPG
  - Historiquement avec les codes opération
    ADDDUR / SUBDUR
  - En format libre avec + et – et les fonctions de durée :
    - Datliv = datcmd + %months(2);
    - Hier = aujourdhui - %days(1);
    - Un = %diff(demain : aujourdhui : *Days);



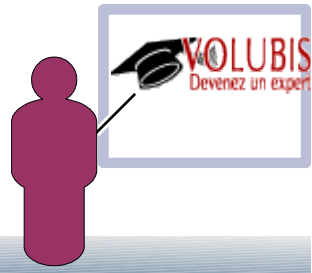
Nouveaux types de données

- Calculs en COBOL
  - Avec les fonctions du ILE COBOL
    - ADD-DURATION

```
MOVE FUNCTION ADD-DURATION (datcmd MONTHS 2) TO datliv.
```

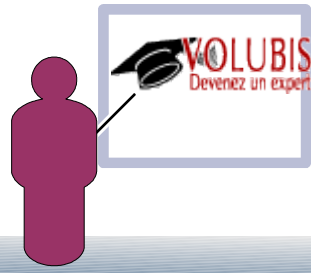
    - FIND-DURATION

```
COMPUTE ecart = FUNCTION FIND-DURATION
      (datliv datcmd MONTHS) .
```



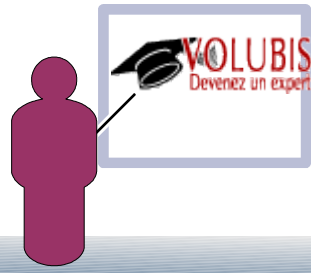
Nouveaux types de données

- Champs auto-incrémentés
  - ROWID
    - VARCHAR(40) basé sur le n° de série
    - Peu pratique à manipuler
    - SQLTYPE(ROWID) en SQLRPGLE



Nouveaux types de données

- Champs auto-incrémentés
  - Numérique AS IDENTITY
    - Type numérique « classique » (integer,décimal)
    - --START WITH (valeur initiale)--------------------
      --INCREMENT BY(incrément) –MINVALUE(valeur mini)-
      -MAXVALUE (valeur maxi) ------------------------
      ---CYCLE | NO CYCLE
    - Avec CYCLE quand MAXVALUE est atteint, on boucle...



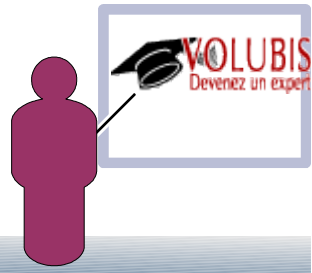
Nouveaux types de données

- Champs auto-incrémentés
  - Récupération de la dernière valeur
    - Identity_val_local()
    - Encore mieux en V6 : FINAL TABLE
    - Exemple avec T1 possédant Code AS IDENTITY et Quand de type TIMESTAMP

```
SELECT Code, Quand FROM FINAL TABLE
(INSERT INTO T1 (lib, quand) VALUES('test', now()) )
```

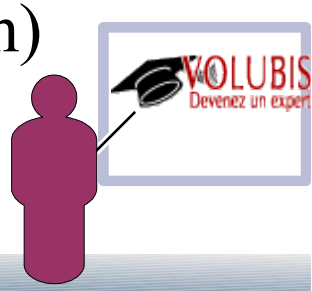
    - affiche:

```
CODE QUAND
   3   2011-05-17-11.34.17.455674
```



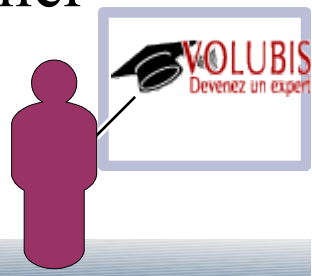
Nouveaux types de données

- LOB vs DataLink
  - LOB : Large Objects
    - Contiennent physiquement la donnée
    - De trois types :
      - BLOB : binaire, non soumis à CCSID
      - CLOB : caractère, texte CCSID un octet
      - DBCLOB : CCSID DBCS, Unicode
  - Ex : CALL CREATE_SQL_SAMPLE(un_nom)
    puis regardez EMP_PHOTO



Nouveaux types de données

- LOB vs DataLink
  - Datalinks
    - Contiennent simplement une URL (protocole Http:// ou File://)
    - Peuvent faire référence à un serveur DLFM
      - Chargé de vérifier et de garantir l'existence
      - La gestion des droits peut être à la charge de DLFM ou du système de fichier
    - Attention, un même fichier ne peut être référencé que par **une seule ligne**



Nouveaux types de données

- Une colonne de type LOB peut-être manipulée par
  - son contenu (Pensez au 16 Mo du RPG V6)

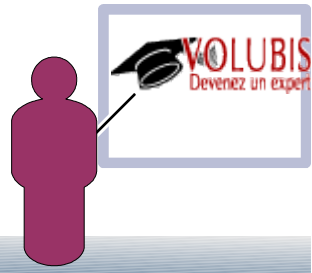
vous devez déclarer en RPG par:

```
DMYBLOB          S                   SQLTYPE(BLOB:500)
```

ce qui génère :

```
D MYBLOB          DS
D MYBLOB_LEN                        10U 0
D MYBLOB_DATA                       500A
```

- Pour recevoir le contenu dans une variable



Nouveaux types de données

- Une colonne de type LOB peut-être manipulée par
  - Un « locator » (sorte de pointeur)

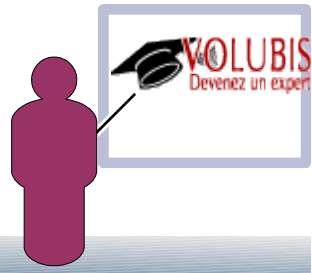
vous devez déclarer en RPG par:

```
DMYBLOB          S                   SQLTYPE(CLOB_LOCATOR)
```

ce qui génère :

```
D MYCLOB          S              10U 0
```

- Permet d'insérer facilement le résultat d'un Select



Nouveaux types de données

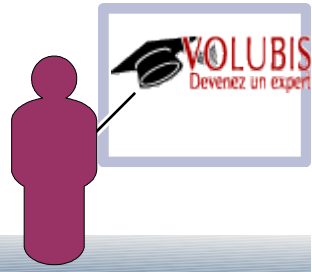
- Une colonne de type LOB peut-être manipulée par
  - Un « File locator » (pointeur sur un fichier)

vous devez déclarer en RPG par:

```
D MYFILE          S                   SQLTYPE(CLOB_FILE)
```

ce qui génère :

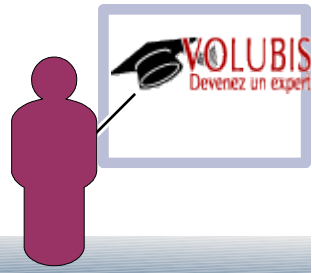
```
D MYFILE          DS
D  MYFILE_NL                      10U 0    [lg du nom]
D  MYFILE_DL                      10U 0    [lg des Data]
D  MYFILE_FO                      10U 0    [file permission]
 * SQL génère SQFRD (2), SQFCRT (8), SQFOVR(16) et SQFAPP(32)
D  MYFILE_NAME                   255A      [nom du fichier]
```



Nouveaux types de données

- Une colonne de type LOB peut-être manipulée par
  - Un « File locator » (pointeur sur un fichier)
  - Lors d'un Select cela Exporte les données dans le fichier dont le nom est fournit dans MYFILE_NAME
  - Lors d'un Insert cela copie les données actuellement stockées dans le fichier dont le nom est dans MYFILE_NAME
  - La V7R10 apporte aussi les fonctions :
    - GET_BLOB_FROM_FILE()
    - GET_CLOB_FROM_FILE()
    - GET_DBCOLB_FROM_FILE()

```
EXEC SQL values
 GET_CLOB_FROM_FILE('/annotate.log')
 into :variable;
```



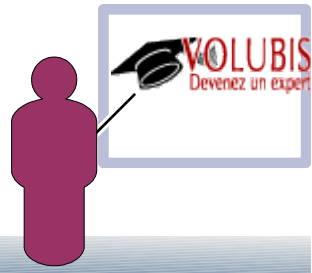
Nouveaux types de données

- V7 : XML
  - CREATE TABLE CUSTOMER (CID BIGINT NOT NULL PRIMARY KEY , INFO XML) ;
  - Sous une session 5250, la sérialisation (conversion en une chaîne simple) n'est pas faite par défaut :

```
Première ligne à afficher . . _____
....+....1....+....2....+....3....+....
                    CID   INFO
                  1.000   *POINTER
                  1.002   *POINTER
                  1.003   *POINTER
********  Fin de données  ********
```

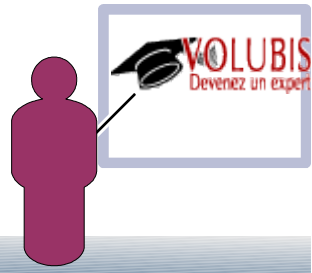
  - il faut utiliser XMLSERIALIZE(INFO as CHAR(2000)) pour voir le contenu, le CCSID du job doit être renseigné (pas de 65535).

```
Première ligne à afficher . . _____
....+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8....+....9.
                    CID   XMLSERIALIZE
                  1.000   <customerinfo xmlns="http://posample.org" Cid="1000"><name>Kath
                  1.002   <customerinfo xmlns="http://posample.org" Cid="1002"><name>Jim
                  1.003   <customerinfo xmlns="http://posample.org" Cid="1003"><name>Robe
********  Fin de données  ********
```



Nouveaux types de données

- V7 : XML
  - EN programmation se manipule comme un LOB
  - Avec les SQLTYPE : XML_BLOB , XML_CLOB , XML_DBCLOB
  - La déclaration suivante
  - ```
    D MON_XML                S                   SQLTYPE(XML_DBCLOB:2500)
    ```
    Génère :
    ```
    D MON_XML                DS
    D MON_XML_LEN                          10U
    D MON_XML_DATA                           C     LEN(2500)
    ```


    ```

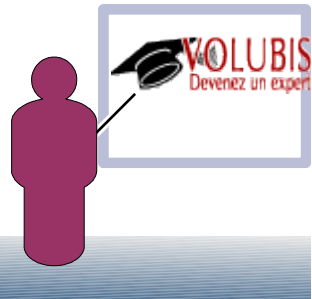
Nouveaux types de données

- V7 : XML
  - XML_LOCATOR
  - XML_BLOB_FILE , XML_CLOB_FILE , XML_DBCLOB_FILE
  - La déclaration suivante

```
D MON_FICHIERXML     S                 SQLTYPE(XML_CLOB_FILE)
```

Génère :

```
D MON_FICHIERXML     DS
D MON_FICHIERXML_NL              10U
D MON_FICHIERXML_DL              10U
D MON_FICHIERXML_FO              10U
D MON_FICHIERXML_NAME           255
```

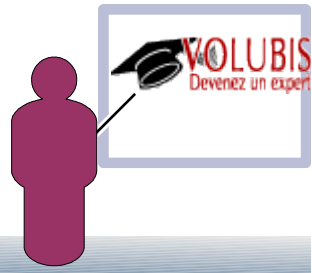


Nouveaux types de données

- V7 : XML en RPG, exemple

(rappel, par défaut les champs XML sont en UNICODE)

```
 * récupération du XML dans une variable du langage
 *=============================================
D MON_XML          S                SQLTYPE(XML_DBCLOB:2500)
D pos              S           5i 0
 /free
      exec sql
         select info into :mon_xml from posample/customer
           where cid = 1000;
      eval pos = %scan(%ucs2('phone') :mon_xml_data);
   *inlr = *on;
 /end-free
```

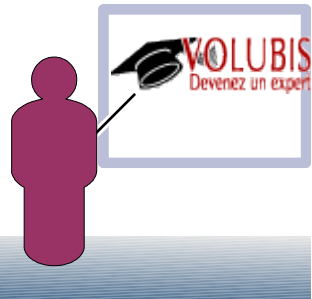


Nouveaux types de données

- V7 : XML en RPG, exemple

```
* récupération du XML dans un fichier de l'IFS
*============================================
 DMON_FICHIERXML   S                 SQLTYPE(XML_DBCLOB_FILE)
  /free
    mon_fichierxml_name = '/temp/xml01.xml';
    mon_fichierxml_nl = %len(%trim(mon_fichierxml_name));
    mon_fichierxml_fo = SQFOVR;

     exec sql
        select info into :mon_fichierxml fromposample/customer
          where cid = 1000 ;
     *inlr = *on;
  /end-free
```



Nouveaux types de données

- V7 : XML en RPG, exemple
  - le fichier xml01.xml est bien généré dans /temp (CCSID 1208) et contient :

```
************Beginning of data**************

<?xml version="1.0" encoding="UTF-8"?><customerinfo
  xmlns="http://posample.org" Cid="1000"><name>Kathy
  Smith</name><addr country="Canada"><street>5
  Rosewood</street><city>Toronto</city><prov-
  state>Ontario</prov-state><pcode-zip>M6W 1E6</pcode-
  zip></addr><phone type="work">416-555-
  1358</phone></customerinfo>

************End of Data********************
```

  - un ordre INSERT aurait lu le fichier XML de l'IFS et placé son contenu dans une colonne de la table SQL.
  - La fonction GET_XML_FROM_FILE() est aussi utilisable