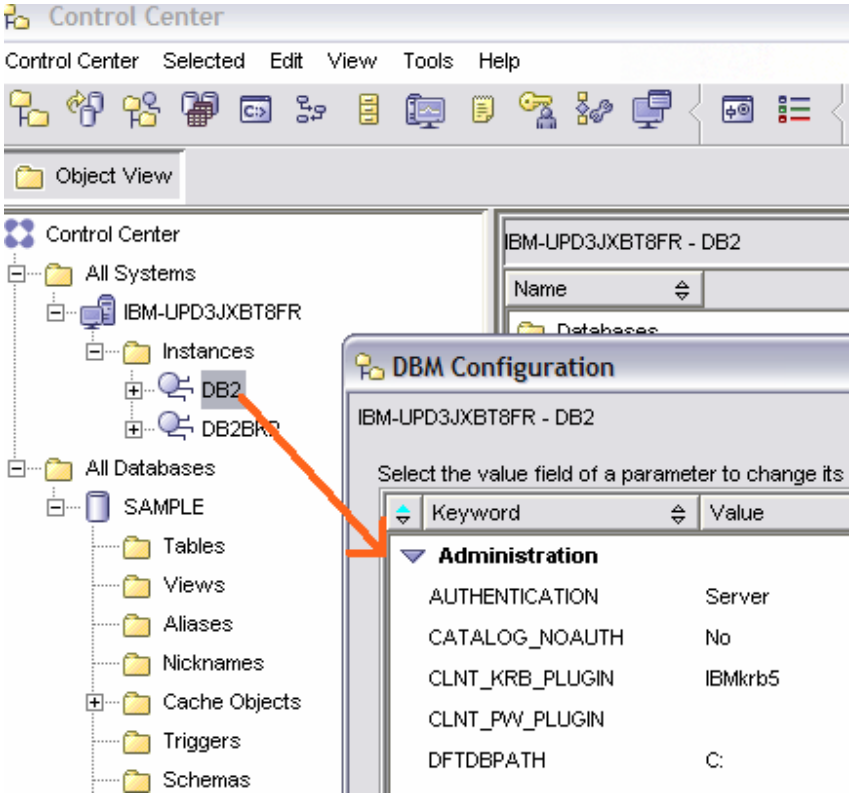
# LAB 03 - SECURITY

## A. INSTANCE LEVEL SECURITY

First let's check the instance configuration parameters from the Control Center

1. Control Center ⇨ right click instance "DB2" ⇨ "Configure Parameters"
2. Review **Administration** section of DBM cfg parameters from the GUI
   - This "section" of DBM cfg parameters control security at the instance level
   - A "section" of DBM cfg parameters is a Control Center concept only, this arrangement is to help you find "like parameters" within the GUI




3. Click on parameter **AUTHENTICATION** ⇨ value box
   - You can set the DBM cfg value through the GUI
   - Notice the "Hint" at the bottom of the GUI box that describes what this parameter does
4. Click on '**…**' in this to begin edit ⇨ notice various kinds of authentication available
   - Note: Review this only; don't change anything!

5. Review parameters **TRUST_CLNTAUTH** & **TRUST_ALLCLNTS**
   - Note: these work with client authentication
6. Review parameter **SYSADM_GROUP**
   - Note: This authority grant is very important on UNIX system setups as it is the first highest authority level you can have
   - Is NULL on Windows due to the way windows handles administrator capabilities
   - Note other instance level “authorities” SYSCTRL, SYSMAINT. Do you remember what privileges these give you?
7. Review parameter **SVRCON_AUTH & SVRCON_PW_PLUGIN**
   - What do these do?

**Section Answers:**

A6. See presentation for authorities overview of SYSCNTL and SYSMAINT.

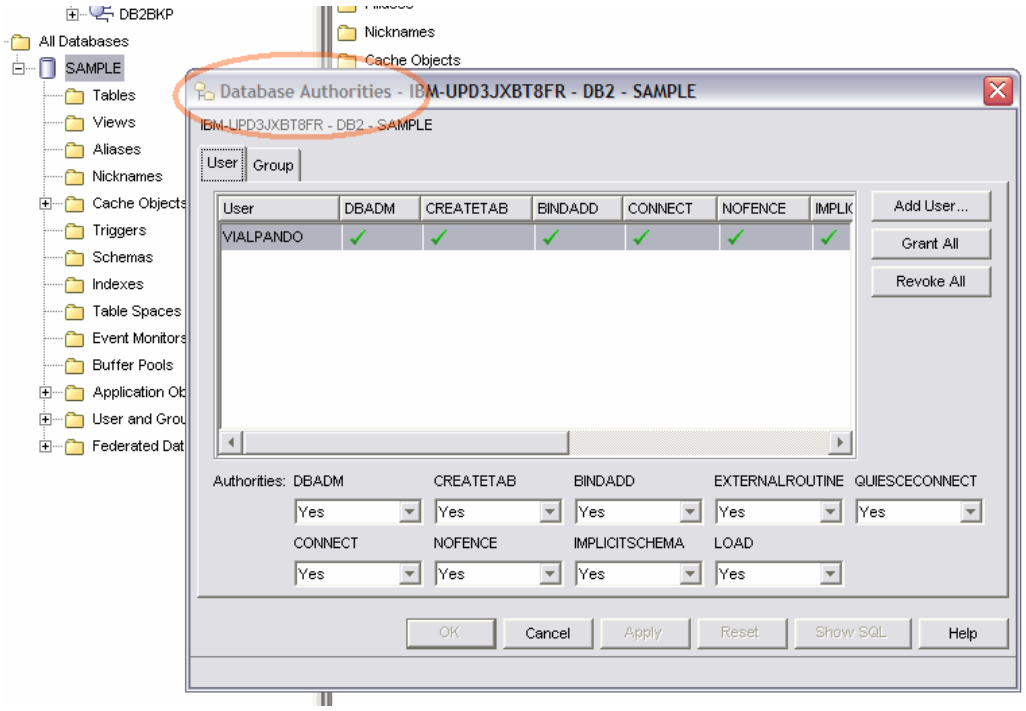
A7. As the presentation mentioned, plug ins are DB2s way of doing database authentication. The basic plug in that uses the OS to authenticate comes with DB2, all others are optional purchases from vendors or they can be custom built by your IT shop.

Note: These together can override the AUTHENTICATION parameter setting

## B. DATABASE LEVEL SECURITY

Now let's check database level security. Unlike with instances, this is not parameter controlled for databases, but rather is controlled by *authorities* for a user or group.

1. Control Center ⇨ right click database "SAMPLE" ⇨ "Authorities




2. Notice **DBADM**, the second highest authority level you can have
   - Do you remember what privileges this gives you?
   - Find 8 other database authorities
3. On "User" tab click [Add User]
   - Notice a list of OS users comes up for selection
2. Select **GUEST** (or any other user) and click [OK]
   - Note: You did NOT just now add a user into the database! You are simply ready to create DCL for authorizations in the GUI and have enabled the DB2 GUI to know about an OS user that does not yet have any grants in the catalog
3. For user **GUEST**, under **DBADM** change "no" to "yes"
4. Click [Show SQL] to see what you've accomplished so far
   - Don't click [apply] just yet

5. Add another user, then grant them something different ⇨ Click [Show SQL]
6. Click [Grant All] for one of the users ⇨ Click [Show SQL]
7. [Apply] executes the DCL you've just created through the GUI
   - Note: Groups work exactly the same way as users
   - Check for group PUBLIC and revoke these authorities if you don't want everyone to have these

**Section Answers:**

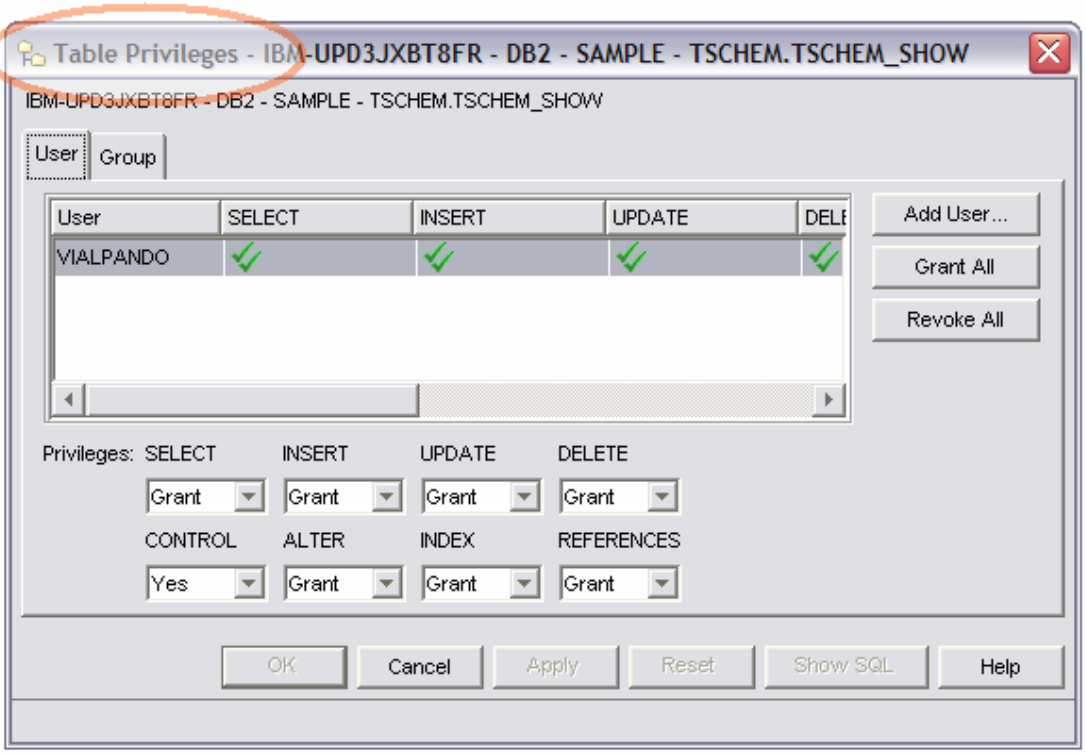
B2. DBADM privileges are covered in detail in the presentation.

DBADM, CREATETAB, BINDADD, EXTERNALROUTINE, QUIESCECONNECT, CONNECT, NOFENCE, IMPLICITSCHEMA, LOAD

## C. OBJECT LEVEL SECURITY

Control Center ⇨ click database “SAMPLE” ⇨ [+] (expand tree) ⇨ Tables

1. Right click on any table ⇨ Privileges
   - Notice individual object privileges work similarly to database authorities, only the difference is these have a “grant option” selection ability on most of them, while authorities do not.

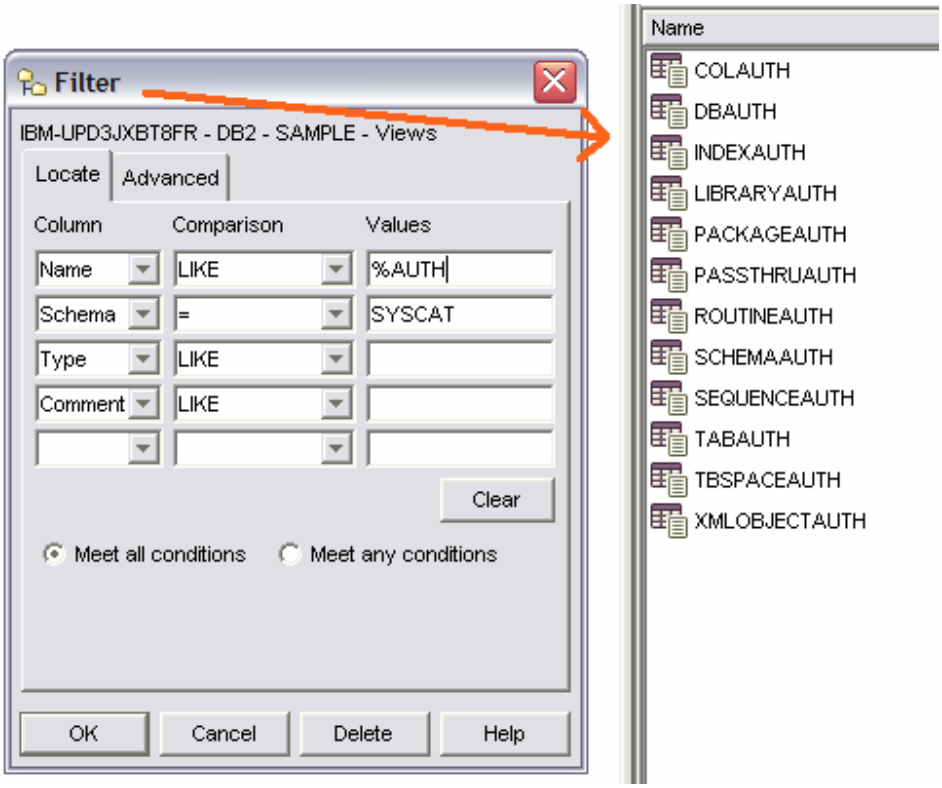



2. Grant various privileges to these and explore this functionality
   - Security for most other database objects works similarly (take a look at a couple of these)
   - The exact kind of privileges vary from object type to object type, for example
     - table spaces have only one privilege: use
     - indexes only have: control
   - Other objects, like triggers and buffer pools, don’t have any specific privileges associated with them directly at all; again, if you haven’t already, take a look at other object’s privileges
3. Exit from wherever you are back to the main CC menu

## D. USING THE CATALOG SECURITY VIEWS

Now let's look at how we can tell what privileges and authorities we have in our database by looking at the catalog instead of the CC GUI:

1. Control Center ⇨ click database "SAMPLE" ⇨ [+] (expand tree) ⇨ Views
2. Create a filter: Name like **%AUTH**, Schema = **SYSCAT**

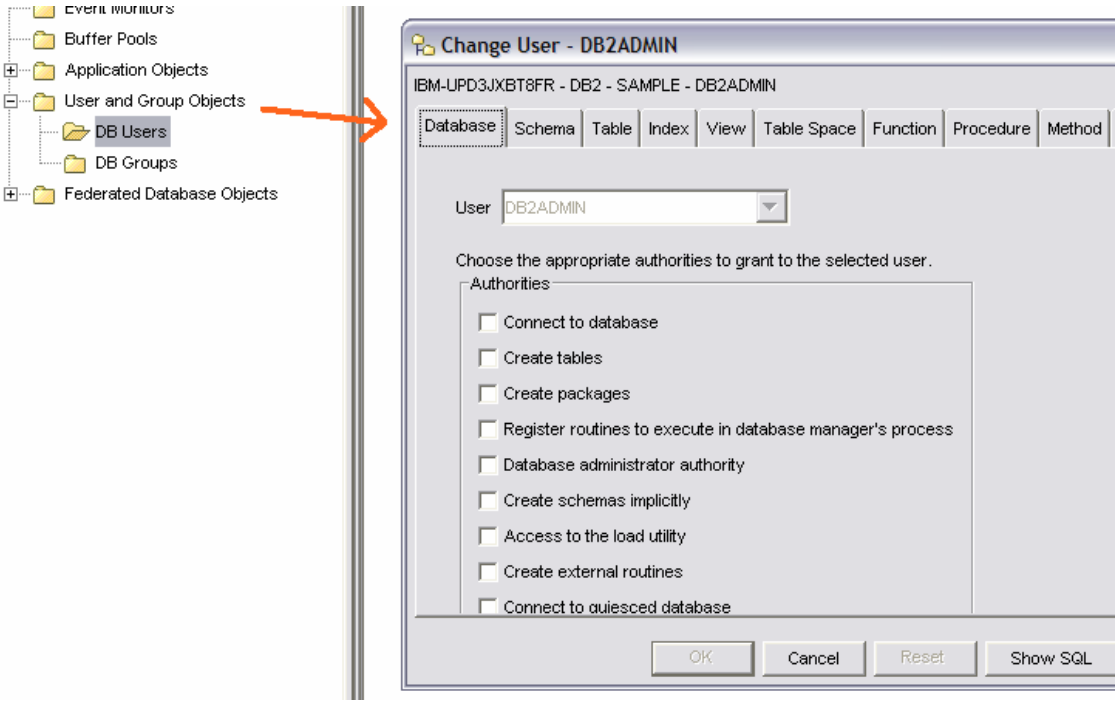



3. You should see a view called **DBAUTH**, double click on it. Are there any grants to PUBLIC? If so, in your own IT shop at work, would you want these for PUBLIC?
   - Revoke PUBLIC privileges if you don't want everyone to have them.
   - Do you remember where or how to grant or revoke DBAUTHS from the GUI?
4. Double click on view **TABAUTH** ⇨ [Fetch More Rows] to see more in the view
5. In this **TABAUTH** view, click on [Filter] ⇨ Tabschema = [your user id] ⇨ [OK]
   - This will show all your current table privileges
   - Note: you cannot add records directly to a catalog table or view, this must be done through grants just like in Oracle
6. [Close]

## E. USER AND GROUP OVERVIEW

Now let's explore the CC special view for users and groups

1. Make sure you position yourself in the C:/POT/03 SECURITY/ directory before continuing
2. Control Center ⇨ click database "SAMPLE" ⇨ User and Group Objects
3. Double click on any user or group to get an overview of all the things we have explored so far about privileges and authorizations
   - Remember: users in DB2 are only in the grantee/grantor context within the catalogs. If you have not made grants to a particular user or group in DB2, they don't "exist" in the database.




4. Add some new grants and apply them. See if you can find these in the other places we've already looked.

You can also get your current user id authorizations by issuing a "get authorizations" command:

5. Review and run these scripts:

   **Security03001.CMD** which executes ⇨ **Security03002.DB2**

6. Explore output file **Security003001_OUTPUT.TXT**

## F. SCHEMAS

1. Since you do not create users in DB2, you do not need them to create a schema either, so this is how you create schemas in DB2; review and run scripts:

   **Security03003.CMD** which executes ⇨ **Security03004.DDL**

   - No output file is made, but schema TSCHEM is created

2. Explore in the Control Center the schema created and the table created in that schema


   - Hint: a table filter for schema "TSCHEM" will help you find the new table
   - Who "owns" the schema? Who "owns" the table?

3. You can control default schemas in a line command with
   - **db2 set current schema=[your schema name]**
   - This sets a special register called **CURRENT SCHEMA**
     - From the Command Window try: **db2 select current schema from sysibm.sysdummy1**
   - IMPLICITSCHEMA privilege lets DB2 create a schema for you if it doesn't exist when you create an object with schema_name.object_name

**Section Answers:**

F2. The id that created the schemas own them by default, same situation for the tables.

## G. DATA ENCRYPTION

**Note: Data encryption in DB2 is done with a set of user defined functions (UDFs)**

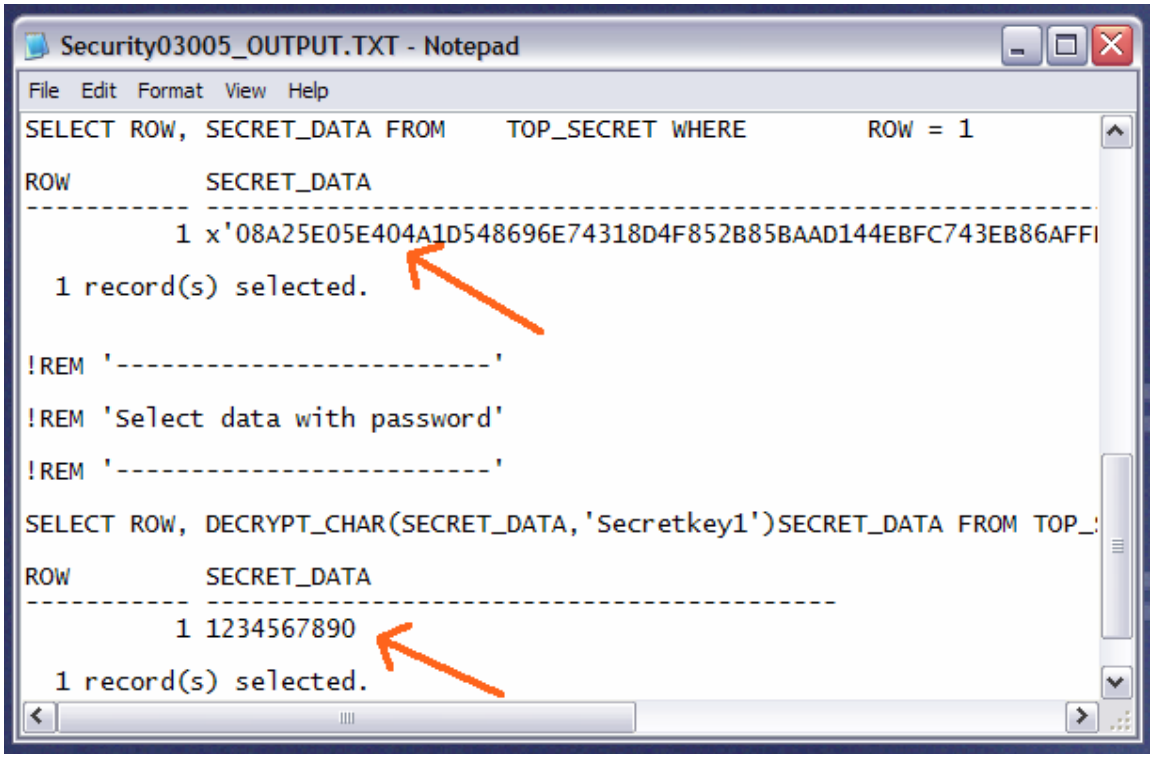
- encrypt (datastring, passwordstring, hintstring) — Encrypts the data
- decrypt_char (datastring, passwordstring) — Decrypts character data
- decrypt_bin (datastring, passwordstring) — Decrypts binary data
- get_hint (encrypted_data) — Retrieves the hint for the password in the data

You can also set a password string with setting a special register:

**set encryption password [passwordstring or host variable]**

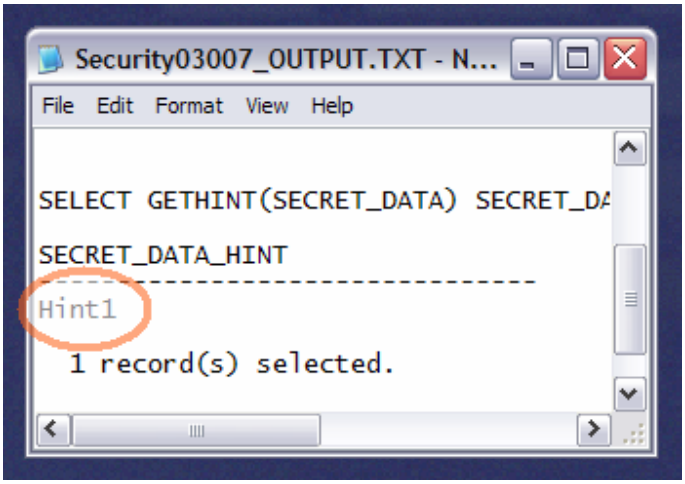
1. Explore data encryption:
   - Review and run scripts:

     **Security03005.CMD** which executes ⇨ **Security03006.SQL**
   - Explore output file **Security003005_OUTPUT.TXT**
   - Notice how data column "secret_data" looks with and without the password



2. Explore getting password hints out of data:
   - Review and run scripts:

     **Security03007.CMD** which executes ⇨ **Security03008.SQL**
   - Explore output file **Security003007_OUTPUT.TXT**
   - Notice value of “secret_data_hint” from select using the built-in function




3. Explore using encryption password register:
   - Review and run scripts:

     **Security03009.CMD** which executes ⇨ **Security03010.SQL**
   - Explore output file **Security003009_OUTPUT.TXT**
   - Notice how you can set a password with a register for inserting and retrieving encrypted data

## H. EXTRA EXERCISES - DB2AUDIT:

DB2s passive security (auditing) is controlled by the db2audit facility. In this lab we cannot possibly go into all the various scenarios that this utility can audit, but you can get an idea by reviewing the db2audit command: **Security DB2AUDIT_Command.txt**

1. Turn on ALL auditing (not normally recommended, but done here for demonstration purposes) and perform some DB2 commands to see how db2audit works:
   - Review and run scripts:

     **Security03011.CMD** that executes ⇨ **Security03012.BAT**
   - Explore output file **Security003011_OUTPUT.TXT;**
   - Note: script starts and stops db2audit for you!
   - Note: output file will have an **SQL30082N** code, this is supposed to happen!
   - Review ***.del** files in C:\Program Files\IBM\SQLLIB\DB2\security\
     - You just created them with this script from db2audit.log using db2audit.cfg
     - .del files can be loaded into a DB2 table

**Note: db2audit has these 7 auditing categories that encompass anything you would want to audit:**

| | |
|---|---|
| Audit events | Checking events |
| Object Maintenance events | Security Maintenance events |
| Sysadmin events | Validation events |
| Context events (any SQL for example) | |

2. One of the 7 output delimited files (see note above) is called the "context" data file, let's load it into a DB2 table by using the following scripts; Review and run scripts:

   **Security03013.CMD** that executes ⇨ **Security03014.DDL** & **Security03015.DDL**
   - Explore output file Security003013_OUTPUT.TXT
     - The last part of the output shows the content of **AUDIT.CONTEXT**
     - Or use Control Center to browse table **AUDIT.CONTEXT**

## I. OTHER EXTRA EXERCISES:

Operating System user workings

1. Create an OS user
2. Use Control Center to see if this user is listed anywhere in the catalog security views
3. Use Control Center to grant connect authority to the database only

**Section Answers:**

I1. Intro.doc already gives a step by step on doing this.

I2. Under Views, create a filter with name like AUTH%. Check these out.

I3. Your DML should look something like:
**GRANT CONNECT ON DATABASE TO USER DBAPOT;**