

## Lesson 3 Transcript: Part 2 of 2 – Tools & Scripting

### Slide 1: Cover

Welcome to lesson 3 of the DB2 on Campus Lecture Series. Today we are going to talk about tools and scripting. And this is part 2 of 2 parts. My name is Raul Chong, and I'm the DB2 on Campus Program Manager.

### Slide 2: Agenda - Command Window / CLP

In the first part we covered the first 3 sections of the agenda, and in this second part, we will start talking about the command window, and the command line processor, or CLP.

### Slide 3: Command Window and Slide 4 Command Line Processor (CLP)

So let's start with the command window. And rather than showing you these slides, let me actually do some demonstrations.

### Demo

So to start a command window, you can do it in different ways. you can go to Start > Programs > IBM DB2 > DB2COPY1 > Command Line Tools > Command Window. This is the long way to do it. The short way to do it would be to go to Start > Run, and just type **db2cmd** and press Enter. Note that the title of this window is "DB2 CLP - DB2COPY1". So there's a distinction, like, many people may get confused about this window, because they think this is an MS-DOS prompt window. An MS-DOS prompt window is not the same. So for example, if I type from here **db2 connect to sample** I have no problems in connecting. On the other hand, if I start a regular "cmd", or MS-DOS prompt, like here, and I type **db2 connect to sample** then I will get an error message, because this is not the command window or command line processor. I can from here type **db2cmd**, and that would open another window, which is another DB2 command window. I just close the MS-DOS prompt.

So anyway, just make sure that you are working with the command line processor or command window by looking at this title, and it's not the same as regular MS-DOS prompt. Now that I have two windows open, you can treat each of these windows as independent applications, in a sense. From here I could connect to another database, and say **db2 connect to research** and now I'm connected to the RESEARCH database in this window, and I'm connected to the SAMPLE database in this other window. So you can think of these windows as different applications that are connecting to different databases, or even to the same database. Let me just exit from this window, to keep things cleaner. Now, from here, we are talking about the db2 command window. And that means that every command you execute will have to be prefixed by the keyword **db2**. So, right now that I'm connected, if I do **db2 select \* from staff** then I will get the output. By the way, if you are working on Linux you will need to put quotes here.

On Windows it's optional, but on Linux you need to put the quotes because otherwise the Linux parser may interpret the wildcard characters in a different way. So, on Windows, as I said before, it's optional. Now, what happens if I type just the command **db2** and press Enter? If I do that, the prompt changes to **db2**. So when the prompt says **db2** as in this case, I don't need to prefix anything with **db2** anymore, I can just type **connect to sample** from here without prefixing. I can do **select \* from staff** without doing any prefixing because I'm within the db2 prompt. When I'm within the db2 prompt, this is called the Command Line Processor. If I'm outside, and if I type **quit** now the prompt is different. Then this is called the command window. So that's the only difference between the command window and the command line processor. The command window, the prompt will not have **db2**; the command line processor, the prompt will have **db2**.

Now, my preference is to work from this prompt, that means from the command window, because if I work from this prompt, which is the operating system prompt, I can issue statements like this, I can say **db2 select \* from dept**, and I can redirect this to another file. Let's call it... I can put it into a file, let's say, `raulerase.txt`, for example, and then I can redirect the file. I can open the file, `raulerase.txt`. That'll open notepad, and I will get my result here. So basically, because I'm working from this prompt, I can use some operating system characters like redirection, while in the db2 prompt, I would not be able to do this. Ok, so that's pretty much it with respect to the command window and the command line processor. We will talk more later about it, or we will use these tools later when we talk about scripting. So let's exit for now. And let's move on to the next presentation, or next slide.

### **Slide 5: Agenda – Task Center**

So now we are going to talk about the Task Center.

### **Slide 6: Task Center**

So what is the Task Center? So let me just provide a demo from here.

#### **Demo**

The Task Center can be started using different methods as well, you can go to Start > Programs > IBM DB2 > DB2COPY1 > General Administration Tools > Task Center. Another way to start it will be to use Start > Run > **db2tc**, "t" for task, "c" for center, and again in Linux you can start these tools this way as well. So I click OK and I'm starting the DB2 Task Center.

### **Slide 7: Task Center—No Tools Catalog Warning Dialog**

Now, as shown here on the presentation, the Task Center and Journal, or these type of tools where you need scheduling, requires another database, or another set of tables called the "Tools Catalog". This is basically a set of tables that will store all this scheduling information. Like, for example, if you want to perform a backup at 3:00 in

the morning, this information will be stored in this tools catalog database or this set of tables.

### **Slide 8: Scheduling With Task Center**

You can create these tables. I will show you in a few minutes how to create the m.

#### **Back to demo**

So this is the Task Center. To create that tools catalog database, you can go to Tools > Tools Settings, and from here you go to Scheduler Settings. And from here, this is the area that you need to complete to create the tools catalog. In my case it is not allowing me to create another tools catalog because it's already been created, so that's why this has been grayed out. Anyway, it's fairly easy to create the tools catalog database. So I'm sure you can complete that task. Ok, now that I'm in the Task Center... The Task Center, as the name says, is used to create some tasks, right? An example I gave you is a typical example of using the task. So the example I provided was to create a backup at 3:00 in the morning, let's say, every Sunday.

So obviously you don't want to wake up at 3:00 in the morning and issue the **backup** command. So what you could do is to create a task and you can schedule the task for this backup to be executed at 3:00 in the morning on Sundays, automatically. So let me create that task very quickly right now, and click on Task > New. I can say, OK, what's the name of my task. Let's call it "mytask". And then I'm going to put a description of the name, which is "This is a test". It could be a DB2 command script, an OS command script, even an MVS shell script. So you could also create tasks to execute against the mainframe. Then we have to specify which instance we want to execute this task on. You can click on those three buttons to choose a DB2 instance and the partition. And we say... we click on that arrow and we click OK.

Now in DB2 Express-C there's only one partition, so it will always be 0. But with other editions you can have more partitions. This is not going to be covered in this particular course. Now, I'll move on to the next tab. So I click Command Script, and from here I can do **connect to sample**. So here I'm going to create my script. And I chose it to be a DB2 script. So I can issue SQL statements here, and say **connect to sample, select \* from staff, select \* from employee**, etc. The termination character, again, is a semicolon. And for Run Properties, you could add more information. We'll just keep it simple. We can add some of the information in terms of Scheduling and Notification, you know, if the task is successful, or if there's a failure, who should I contact. Task Actions means that if your task is successful, whether I should run another task, should I enable the scheduling of another task, should I delete a task, etc. If the task fails, the same conditions apply. You could do other actions.

Security means in terms of who can run this task. So right now, let's just click OK, and that will create my task, which is shown here. I can run it now by right clicking on the task and choosing Run Now. So I'm going to put my password here, and then

execute. And you can see that the icon now changes to a man running because the task is running right now. If I refresh this, I go View > Refresh, it now shows that the task has finished, and there's a check, saying that the task was run successfully. I can choose the task, right click, and select Show Results, if I want to from here, and you can look at this script that ran, and here's my output. So I connected, I executed **select \* from staff**, and **select \* from employee**, and everything was fine. So that's how you can create a task. Now from here, since we are working with tasks, we are also going to talk about the Journal, which is another tool. So let's click on this button to start the journal. The journal is basically showing you several messages for Task History, Database History, Messages, or Notification Log.

So for Task History, I can choose a system which I want to take a look at the tasks. Here we can see the same tasks that are run from the Task Center. So I can double click on this task and it's the same dialog box or the same window that I showed you before when I right-clicked on the task and selected Show Results. So from the Journal you can also see the results of your tasks. You can see some information about databases, messages in general, and then the notification log for different messages at the system level. So this is a good place for a database administrator to find problems.

Let's say, a user comes to you or drops by your office and says, "You know, I noticed that yesterday at 6 p.m., the system was running slow", or something like that. And then you, as a DBA, can go to this Journal, and try to find 6 p.m., or 6 a.m., whenever the problem was being reported, to see what type of messages you find at that time. So, that's one of the main purposes of using the Journal. Basically it keeps track, or logs messages about things going on with the database. Ok, so that's the Task Center, and that's the Journal. Let's move on with this presentation.

### **Flipping through slides 9-11 Journal**

I've already talked about the Journal,

### **Slide 12: Health Center**

So let's talk about the Health Center.

### **Slide 13: Health Center**

So the Health Center is basically a tool that allows me to be proactive about my database. So rather than waiting for a problem to happen, let's setup some thresholds, some limits, where I'll get a warning, or an alert. So those will be sent to me before the problem happens, and that way I'm being a proactive DBA.

### **Demo**

So let me show you very quickly the Health Center. I can go to Start > Programs > IBM DB2 > DB2COPY1 > Monitoring Tools > Health Center. Now, from the Health Center, you can configure things at the instance level, or at the database level. So there are different levels where you can perform configuration. And this tells you how

often you will refresh the information. So right now I can expand this. And in this case I'm seeing that I'm getting some sort of alarm for the Monitor Heap Utilization, which has a value of 100%. So this is telling me that there's a problem right now for this DB2 instance.

I can also configure, and I choose Health Center > Configure. I can configure different settings, as I said before, at instance level, global settings, or object setting. If I choose at the instance level, and then I select the DB2 instance, then at the instance level I have, in this case, only 3 settings, so in this case, for example, Monitor Heap Utilization, I have that a warning will be sent at 85% an alarm will be sent at 95%, and the unit used is percentage. I can double click here, and I could change these to different values, at 88%, 96%, etc. But basically here I'm setting the thresholds that I want so that I get these messages and I can take a proactive approach. So basically that's our main purpose of using the Health Center.

### **Flipping through Slide 13-14**

Moving on

### **Slide 15: Agenda - Scripting**

We are now going to talk about scripting.

### **Slide 16: Scripting**

So, there are two types of scripting. One is called SQL scripts, which means all the content of your script will be SQL statements. And the other one are operating system scripts, or shell scripts, which are scripts from the operating system.

### **Slide 17: A Basic SQL Script**

So we are first going to talk about SQL scripts, and here I have one example, where the name of the script is script1.db2. Now, the extension doesn't have to be db2, it could be anything. And the contents are SQL statements. In the SQL script you need to use a terminator character, in this case, it's a semicolon, which is the default terminator character.

### **Demo**

So let's create quickly a script from here. Let me exit from here, and let's use Notepad, and let me call this raulscr.txt. So this is a new file. From here, let's say **connect to sample, select \* from dept**, and then say **create procedure p1 begin end**". So, I'm just going to connect to sample database, selecting everything from department, and creating a stored procedure. We will talk about stored procedures later. And this stored procedure is doing nothing, but it could be created like this. I think I already had a procedure with this name, so let me put "p199". Now I'm going to go File > Save. Ok, so now I've just created my script. And by the way we already covered scripts as well when we talked about the Command Editor. In the Command Editor we also saved a script and we opened it.

So you can work either from the Command Editor, which is a GUI version of the command window or the command line processor, or you can work here from the command line processor. Now, to execute this script, you can type... you invoke db2 first, and typically you invoke a script using these flags "tvf", and then the name of the script. Ok, what does the "t" means? It means there is a terminator character. And I'm not indicating which one it is because by default it's the semicolon. So, just by putting a "t" means there's a terminator character. "v", it's optional, but it means "verbose", so echo the command and produce the output. And "f" means that the commands are coming from a file, which in this case, is this script file, raulscr.txt.

So if I press "Enter" now, it's going to execute the script, as you saw. So, it has executed the script successfully. And that's very good. So now, what happens if I change the terminator character? Let's say I put an exclamation mark. Exclamation mark, and exclamation mark. And, let me add a **drop procedure p199** here because otherwise it will complain, saying that the procedure already exists when it tries to execute it again. So let me put that as part of the script. And then go File > Save.

Now, to run this new script, I cannot run it like this because the "t" means there's a terminator but it will assume that is a semicolon terminator. So now that we have a different terminator, I can put a "d" here to indicate the delimiter, and the delimiter that I'm using, which is the exclamation mark. And then I need a space and I continue with other options. So here I need a space, so that's why these need to be started again. I cannot put it all together. So I go "db2 -td!", which means there's a terminator where the delimiter is an exclamation mark, and then continue with "verbose", and it comes from a file, and this is the file name. So now I'll execute this and again it executes successfully, including the "drop". So that's how I can execute a script from this command window. And you can also execute this script from the command editor just by opening the same file.

### **Back to slide 17**

Now, ... we worked on a basic SQL script.

### **Slide 18: Executing SQL Scripts**

And this is more information about some of the flags that I used.

### **Demo**

If you want to know more about these flags, you can type the command **list command options**, and it will show you all the different flags that you can use, and also the current setting.

### **Slide 19: When a different statement termination character is needed**

Now if... on this other slide, it shows you typical problems that can happen with scripts.

## Demo

So let me go back to this script here. In this case, I was creating a procedure that was doing nothing. But let's put some code here. So now I can say, "declare x", it's an integer, then "set x = 1", and so on. Now, for a stored procedure... let me do some indentation... for a stored procedure, every sentence needs to end with a semicolon as part of the syntax of creating a procedure. But the entire statement is all of these, which ends with an exclamation mark. Now, what if I had not used the exclamation mark as my delimiter, if all of these had been a semicolon instead of an exclamation mark, which is the default. Then I would have had a problem because DB2 will say, ok, here my statement terminates because I found a semicolon. Here my statement terminates because I found a semicolon. And so on and so on. But now when it goes to here it will say, "Oh, OK, this one stops here because here is where I found my semicolon." But that's incorrect, because the whole statement should end here.

So basically you should make sure you change your terminator, that's in this case, it's an '@' symbol, whenever you are using a character that is used in other statements. In "create procedure" we used a semicolon. In a "create table" statement, we would say "create table T1 (col1 int, col2 int)". Here if we had used a comma as a terminator of the entire statement, then it would also cause a problem. So, it's just information for you so that you make sure to change the terminator so that there is no conflict with characters used within the statements. So let's just save this, and let's execute it again. We change the terminator to an '@' symbol in this case. We execute, and again it connects, and then successfully creates the procedure p199 again.

## Slide 20: A Simple Operating System (Shell) Script

Ok, moving on, we are now going to talk about operating system scripts. And in Windows, you have to create a script and add the batch "bat" extension. This is required only on Windows. In Linux you will basically have use "chmod +x" to give executable privilege to the given file. Within the file, as shown in this example, you can use operating system commands, like "set". And then if you want to invoke db2 SQL scripts within your operating system script, just invoke it as I showed you before. You can use **db2 -td@**, etc. So within an operating system script, you can invoke SQL scripts, but make sure you invoke them correctly.

Now, the advantage of using operating system scripts versus SQL scripts is that you can add parameters, like you can see here, for example, "%DBNAME%". That's a parameter or a variable you can use from the operating system command. And this would be a variable using the syntax on Windows. On Linux it would be a different syntax. So there are pros and cons. If you are using an SQL script, it doesn't matter if it's Linux or Windows, it's the same syntax. But if you use operating system scripts, you know, depending on the operating system, the syntax of the script may have to be changed a little bit. Ok, so for this particular script you could execute the command like this.

## Demo

So, if I were to create the previous script that I had here as an operating system script, first of all, I'll just have to get rid of these terminator characters. I can say, for example, **dir, set db2instance=db2**, and then I can **connect to sample**, but actually I cannot do this, I have to say **db2 connect to sample**. Then I can say **db2**, I can put quotes or not, depending if it's Windows or not, and **db2 drop procedure**, etc. And for simplicity let's just get rid of this. I could put that in a temporary file and invoke that file from here. Anyway, so this could be an operating system script where I'm using operating system commands, **dir**. I'm also using operating system command **set**. And I am invoking **db2**, and I can also invoke **db2**, for example, I can say, **db2 -td@ -vf** and the name of the script that I had before, which was raulscr.txt. Let's erase this because that was done as part of the SQL script. And as you can see, there's no need to put a semicolon here as terminating... for termination of this script.

Now there's a problem here, because I'm saving... I'm putting this in raulscr.txt, so actually I cannot execute now this because I just changed this. But that would be the syntax, and, because I would be invoking myself. So that was just a mistake. But assuming that file exists, you would have invoked that file the way I showed you.

Now let me save this, and close this, and to execute it. Before I execute it I have to change the name, so I use a **rename (ren)** command, so I will rename raulscr.txt to raulscr.bat, because that's what I need to do on Windows. If it was Linux as I said before, you can just change the mode to execute the script; so now if we want to execute the script, and I just press Enter. As you can see, it executed successfully. So the first operation it did was a **dir**, which was part of the script. It did a **dir**, after that, it went on and issued a **set db2instance**, then it did a **db2 connect**, and then it did a **db2 select \* from dept**, and it gave you the output. And you could have invoked another file to execute more other SQL scripts if you needed to.

## Slide 21: Technical Articles About DB2 Scripting

Ok, here are more articles if you want to get more information about scripting.

## Slide 22: QuickLab #5 – Creating An Installation Script

And now I would suggest you to pause this presentation and start working with QuickLab #5 where you can play with the creation of an installation script.

## Slide 23: What's Next?

So, we have reached the end of Lesson 3 (Part 2), and I want to congratulate you because you have completed lesson 3, Tools and Scripting. What is next? It would be Lesson 4, the DB2 Architecture. Thank you for listening.