Lesson 2 Transcript: Part 1 of 2 - The DB2 Environment

Slide 1: Cover
Welcome to Lesson 2 of the DB2 on Campus lecture series. Today we're going to talk about the DB2 environment, and this is part 1 of 2 parts. My name is Raul Chong and I'm the DB2 on Campus Program Manager.

Slide 2: Agenda
This is the agenda for today. Part 1 will cover an introduction and the topic of instances, and part 2 will cover the rest of the topics. We will start with the introduction.

Slide 3: DB2 the big picture
In the introduction I'm showing the big picture of DB2. On the left side I'm showing the commands or statements that you will be executing, and you will probably be executing them on tools, so the tools are shown in the middle of this slide. And with those statements that are input to the tools you will be going to the DB2 environment and to the DB2 engine, which will process those statements or commands, and when it is processed, it will return output to the tool. Now in this particular presentation we're focusing on the DB2 environment.

Slide 4: Agenda
Moving on to the next section, we're now going to talk about instances.

Slide 5: The DB2 Environment
So first of all, you should have already installed DB2 on your computer, so what you see right now is this grey box, which represents your computer with DB2 installed, either on Linux or on Windows. Now in the case of Windows, there will be a default instance created with the name of db2. It could have been called anything else, but it just happens that the default instance is called db2, on Windows. On Linux it's called db2inst1. Now what is an instance? An instance is an independent environment where you can start creating your databases inside the instance.

You can create an instance for production, for test, for development, etc., etc., where each of these instances is an independent environment. In the times where maybe there was not enough hardware, you would create instances to separate environments; one for test, one for development, for example, but nowadays that there is more availability of hardware you may want to put a database on its own instance on its own machine. So now what I'm showing you is that I can create two instances on this computer, on this machine. Of course you can create many more instances on the same machine if you need independent environments. Normally, you don't need to create any instances, though. Now the port that you see on each instance is to uniquely identify an instance and this will be discussed later when we talk about client connectivity. But this port refers to a TCP/IP port because when we talk about connectivity we will use basically TCP/IP.

Slide 6: Creating an instance
Now we will start executing some command to work with instances, but rather than just showing you slides, let me give you a demo.

DEMO

To start working with instances, let's open a DB2 command window, so if you type what you see there, **db2cmd**, and press OK, you will start a window which looks like the MS/DOS prompt, and actually is not an MS/DOS prompt. If you take a look here at the title of the window, it says "db2clp", so make sure it says that, because otherwise things may not work correctly for you. So, from this command window, and by the way, if you're working on Linux, just use the Linux shell after you are logged in as db2inst1 or the instance owner, and then just use a Linux shell.

Now, going back to the example, or to this demo, let's first take a look at the instances I have created or are available right now. So if I type the command **db2ilist**, where i stands for instance, list is to list instances, then I can see that I have only one instance called DB2. If I want to create a new instance, I type **db2icrt myinst**, assuming that myinst is the name of the instance that I want, so I type **db2icrt myinst**, and it should create a new instance called myinst in a few seconds. So now if I type **db2ilist**, I see now that I have two instances, one is myinst, the other is db2.

Now which of these instances is the current one that is active for my particular window? Well, if I type the command **db2 get instance**, it will show me that the instance db2 is the current active instance for this particular window, so that means whichever command I execute, will apply to this particular instance. Now I can switch to another instance by issuing an operating system **set** command and using the DB2 instance environment variable, and I can set it to myinst. So if I issue this command, **set db2inst=myinst**, I should be switching to the myinst instance. Note that there is no space between db2instance, the equal sign, and myinst; there should not be any space, otherwise you will get different results. Now if I type the command **db2 get instance** again, it should now say that myinst is the current active instance. Ok, great!

Ok, now let's create a database on this instance, type **db2 create db testdb**, and I press enter. First of all I'll get an error because by default on any instance, the instance may not be started. To start the instance, I have to type the command **db2start**, and I press enter, and that should bring up or start the instance. One thing is to say that the instance is active on this window, which is, what I mean by saying active is that whatever command I execute will apply to this instance, and the other thing is to say that the instance is started, and that means issuing the command **db2start**. Alright, so now the instance myinst has been started, so now I can create a database testdb on this instance. A database takes a few minutes to create, sometimes one or two minutes, and the reason for that will be explained later, but basically we creating some internal tables which are part of what we call the catalog or dictionary, and we're also doing some configuration as well. So, just wait for a few seconds until the database is created.

DEMO: 2nd DB2clp window

In the meantime, I can open another window, by typing **db2cmd** again, so now I have two windows, and what I want to show in this window, is that if I type the command **db2 get instance**, I will see that this is another window, and in this window, the active instance is db2, so that means any command that I execute on this other instance would apply to the db2 instance. And again, db2 is the name of the instance in this case, the one that is used by default, and it could have been called anything else. It just happens that the first default instance on Windows is called db2, and on Linux as I said before, it's called db2inst1.

Now as you can see from the top window, the database has been created successfully. If I type the command **db2 list db directory**, it shows me now that I have one database available in this instance, which is called testdb. If I type the same command, from the other window, **db2 list db directory**, I have more information, a bunch of databases, toolsdb, research, sample, and mydb because I'm looking at the other instance. The instance or the window at the top is looking at the myinst instance, the window at the bottom is looking at the db2 instance, and in the db2 instance I had several other databases. Let's just exit from these other windows so that we don't get confused.

DEMO: 1st DB2clp window

Now going back to the presentation, we already talked about **db2icrt** to create an instance. In Linux, to create an instance you need to follow a little bit more in terms of the syntax that you need to use and you need to run it as root. So on Linux you need another user, which is the *fence user*. And these are users from the operating system. And there is more information on this slide as to other ways to create an instance; you can also create it using the GUI tool from Linux, which **db2isetup**. And now that we're back in this other window where myinst is the default, is the instance that is active right now, what I'm going to do is do the opposite. I will start by dropping the database, I will stop the instance, and then I will drop the instance, and I will go back to setting the db2 instance as the active instance. So first we drop a database by issuing the command **db2 drop db testdb**. So we drop the database, now we're going to stop the instance and we do a **db2stop** command. Now we're going to drop the instance by doing a **db2idrop myinst**, so I drop the instance and then press enter. And now the instance is dropped. So now if I type **db2ilist**, it will show me that I only have one instance available, which is the db2 instance.

And now I'm going to set back the active instance to db2. Now if I type **db2 get instance** I'm back to having the DB2 instance as the active instance. So this is a very quick demonstration on working with instances from the command line. If you didn't understand this or if you don't like to use the command line we will show you in the next lesson how to do this using the graphical tools.

Slide 7: Creating an instance

Now going back to the presentation, we already talked about **db2icrt** to create an instance. In Linux, to create an instance, you need to follow a little bit more in terms of the syntax that you need to use, and you need to run it as root. So on Linux you need another user, which is the *fence user*. And these are users from the operating system. And there is more information on this slide as to other ways to create an instance; you can also create it using the GUI tool from Linux, which **db2isetup**.

Slide 8: Switching instances

I already talked about this, which is how to use the instance, how to set the db2 instance environment variable and set command. And on Linux, the difference between Windows and Linux, there's not much difference, but one of the main differences is that on Linux, an instance has to map an operating system user. So if I have an instance db2inst3, for example, then db2inst3 has to map a Linux operating system user where there will be a `/home/DB2inst3` directory. Now each of these instances, for these instances, there will be a file called **db2profile**

created which will be added to the .profile or .login files, and in this **db2profile** file, there will be a call to exporting the variable db2inst and setting it to the right instance name, which is in this case is db2inst3. Now depending on the shell, the name may change, it may be **db2profile**, it may be **db2csch**, depending if you're using a C shell or a Born shell.

Now if you are in Linux, you can also just **su** to switch user to the other instance, and that will imply that .profile or .login file will be executed automatically, which would execute the **db2profile** file, which would execute the **export** or **set db2instance=** to the other instance. So just by switching users on Linux, you would be also switching to another instance. In Windows on the other hand, you don't need to map a user to an instance. You create an instance from a given user, and that's it; it doesn't have to map a user (on Windows), and to change instances, you just use the **set** command as shown on the demo.

Slide 9: Reviewing instance information
On this slide we show the **db2ilist** command, which I mentioned before to list all instances on the server and **db2showinst** to show the current active instance on the particular window.

Slide 10: Starting and stopping instance
To stop and start an instance, you issue a command. To stop, you issue **db2stop**, to start, it's **db2start**. Now I show the option **force** on **db2stop** and what that will do is, if you have any connections on databases within an instance, those connections will be forced. That means they will be brought down, and then the instance would be stopped. Of course you probably don't want to do this in production, or at least you have to give a warning that you are doing this because many people will get upset if you do a force on their connections.

Slide 11: Dropping an instance
To drop an instance, you have to stop the instance using **db2stop** and then use **db2idrop** as shown in the demo before. And on Linux, it's the same procedure.

Slide 12: Agenda
The next section to talk about is databases, but that will be covered in the next part, Part 2.

Slide 13: To be continued in part 2...
So this concludes Part 1 of the DB2 Environment, please continue with Part 2. Thank-you.