openSAP
ABAP Development for SAP HANA

WEEK 4, UNIT 1

00:00:12 Hey and welcome to week 4, the final week of this openSAP course.

00:00:18 We hope you earned again a lot of points in the weekly assignments.

00:00:22 Hi there. Let me outline this final week for you. First we will start with how you can consume native SAP HANA artifacts in the Application Server ABAP.

00:00:32 Then we will take a look at the domain-specific consumption of the Core Data Services.

00:00:38 And in the unit 4, we will have a special guest from SAP Business Warehouse showing you how to leverage these new features in the Business Warehouse scenario.

00:00:48 We will conclude the course with an outlook and a preview of what we have in the upcoming releases of NetWeaver 7.4.

00:00:58 So, let's start with unit 1.

00:01:58 Yeah, as mentioned, unit 1 will be about native HANA consumption and that will be the ABAP Database Connectivity, or abbreviated with ADBC.

00:01:10 We will see how to use a call sequence, what a typical call sequence is,

00:01:15 and how you can consume an SAP HANA procedure with ADBC and how cumbersome that actually is.

00:01:23 And last but not least, there will be a sneak preview for the ABAP Managed Database Procedure consumption.

00:01:29 Okay, so what is ADBC? Before we come to that, let me say why you would need it.

00:01:39 In week 2 we had the Detect phase, so if I'm migrating to SAP HANA, first of all I have to detect functional correctness.

00:01:48 Well I hope functional correctness is there. I have to check for functional correctness, for example, with the ABAP Test Cockpit.

00:01:55 Then in week 3 we came to the Optimization phase with Open SQL and Core Data Services. And now we are really with ADBC

00:02:04 or in the upcoming units with ABAP Managed Database Procedures at the Explore phase in this diagram.

00:02:11 So if you really would like or really would have to consume native HANA artifacts or features that are not accessible with Open SQL

00:02:21 or with CDS, you have to dig deeper and you have to consume them natively.

00:02:27 For example, the difference between two dates can be accessed with HANA functionalities

00:02:36 or anything you might be able to think of, there are such a high amount of features in the SAP HANA Business Function Libraries,

00:02:47 which you can access, for example, with ABAP Database Connectivity.

00:02:51 And that really brings us to rethink and innovate your ABAP application programming.

00:02:58 Okay, so ABAP Database Connectivity.

00:03:02 That is the successor, let's say, of EXEC SQL. It is an object-oriented ABAP application programming interface for doing relational database accesses.

00:03:14 It's not limited to SAP HANA. You can do native database programming on any database. Of course, this is database-dependent,

00:03:23 as we have already learned in week 2 when talking about the ABAP Test Cockpit checks for native database programming.

00:03:30 What you are doing in the ADBC is you access the entire SQL functionality of the underlying database

00:03:39 and that's because you just have an SQL string that you pass to the database. We will see that in the system demo in a second.

00:03:47 You have dynamic creation and execution of Native SQL statements

00:03:54 and the good thing, or the good thing compared to EXEC SQL, is that you have clean concept from multiple database connections

00:04:02 and you also have the possibility for exception handling.

00:04:06 And those of you who know Java, you are certainly familiar with JDBC,

00:04:12 so ADBC is very similar to JDBC. So only a hint for those who know already JDBC.

00:04:21 So let me show you a demo what a typical call sequence of an ADBC report is

00:04:27 and how you can consume a HANA Database Procedure will be shown by Jens.

00:04:35 First of all go to the system. I have already created the report for you

00:04:40 and I have already done the data declarations so a string variable holding the SQL string, I already created that for you.

00:04:48 We will have a simple SELECT, with an aggregation then of course, also with a GROUP BY.

00:04:54 It's not that important what this statement does. Let me just say it's a valid statement. It can be executed on the SAP HANA database.

00:05:02 Then we have an lo statement (lo_stmt) that is a reference to cl_sql_statement, one of the four important classes when talking about ADBC.

00:05:14 Okay, first step. We have to create such a statement object. For that I will use here this statement object,

00:05:24 put it down here, it doesn't work. It worked the first time, sorry, I was a bit too slow,

00:05:31 and I just used a new create, the new operator, and the class object here.

00:05:39 In principle, I could now pass as an argument the connection. I will just use the standard connection to the primary database,

00:05:47 which is SAP HANA in that example.

00:05:51 Okay, second step is to execute the query. That is executed on the statement object,

00:05:57 so let me just copy it again and I'm doing the execute query.

00:06:03 You could have DDL statements, procedures, and updates, but we are interested in queries here.

00:06:10 And for that I'm passing the statement string,

00:06:14 just put the statement here. I have prepared that already. Okay.

00:06:21 This execute query does not really return the result yet. It returns an instance of the cl_sql_result_set class.

00:06:32 Therefore, I retrieve the result into the lo_res object, which is here mentioned above a reference to cl_sql_result_set.

00:06:46 This result set now allows us to fetch the data, for example, into an internal table, but before we can do so we have to bind the output parameters.

00:06:56 So, on the result_set object, I could now set the parameter. If it was a scalar parameter,

00:07:05 I could have BLOBS or CLOBs or structures or, in my case, it's an internal table.

00:07:11 I have to give it a reference to an internal table, so I'm just using the reference operator here.

00:07:20 And I have created already a type standard table of the corresponding type for the result, for fetching the result.

00:07:29 So here I put the reference to this internal table. Now we're really ready to fetch the result in the next step.

00:07:39 So on the result_set object, I will do not a fetch, but a next

00:07:47 and next would be for scalars or for a structure output parameters or next package for internal tables where you could even give a package size.

00:07:57 But I will just fetch everything; the result is not that large.

00:08:03 And last but not least, as a good programmer, or good programming style, I'm releasing the resources by closing the result_set object.

00:08:14 Okay, so no syntax errors. Even if that statement would be not valid, no syntax error.

00:08:22 Let me just execute it and you can see I'm retrieving the result as requested with an aggregated gross amount.

00:08:32 So that is the typical call sequence. Not that long but already a bit complicated, but not as complicated as it could get.

00:08:42 Let's see how complicated it can get. Thank you Jasmin.

00:08:48 Now that you know how you can create an ADBC statement and execute it, let's go to a more complex case.

00:08:56 In this case we will execute a HANA Database Procedure.

00:09:02 A HANA Database Procedure, as you remember, is something similar like a function call,

00:09:10 so it's just a method where you can execute procedural coding but on the database,

00:09:15 in the language, in the SAP HANA SQLScript. We will learn about this later in unit 2.

00:09:25 Okay now I got the sentence. So let me start with a demo. I think it is a better idea.

00:09:30 So this procedure, as you might remember, is the one we have executed in week 1, unit 3.

00:09:37 It will return us the customer invoices which have been paid on a specific date. We have already prepared the input parameter

00:09:48 of this procedure on the SAP HANA database inside the statement which is the given date

00:09:54 and it returns us three internal tables, so three tables result set and now let's see how we can consume this.

00:10:04 So this is the statement how you would also execute it on the database directly. We pass the statement to the class SQL statement object

00:10:13 and we get the result as Jasmin has shown you. And now we set the parameter table which is important in this case

00:10:21 because if you have more than one exporting parameter of the procedure, you need to do this exactly that way.

00:10:29 So now we get the package and now there comes a special because we have more than one exporting parameter.

00:10:39 We need to go through a name-value table, internal table, to loop this and find the corresponding exporting parameter names.

00:10:47 These are the names of the exporting parameters of this procedure. After we've done that and bound them to our result table

00:10:55 in which we want to consume them, we are ready to get it and get the content in that internal table that we've bound with a next_package again

00:11:05 and then we close the result_set to clear up the resources on the database.

00:11:10 And then the result is, the same as you've seen in week 1. Here are the customers which invoices have been closed on that specific date

00:11:21 but as you have seen, this can get quite complex if you are coding it natively against a HANA database using procedures.

00:11:29 And for this we can do, maybe, some better things where we will give you a sneak peak already, but let's first of all take a look at the pitfalls

00:11:39 of this ADBC approach. Some of them we have already mentioned during the demo, but let me just sum them up.

00:11:48 So first of all your SQL query you are passing to the database is just a string so you won't get errors during the compile time

00:11:59 or design time. You will, of course, get a runtime error if you do that and runtime errors are not always a good idea.

00:12:07 And, of course, you will have to take care for all of the classical things the Application Server ABAP takes care of in Open SQL for example,

00:12:14 like client handling, the type mapping.

00:12:18 Remember week 1 again where I've told you about the different type systems of databases and application servers.

00:12:25 So this you have to do on your own. If you're not doing it right, again, a runtime error will occur

00:12:31 and of course you have to release the resources and take care for proper locking of your database tables.

00:12:38 And table buffer is also not available. And last but not least, you are here, of course,

00:12:48 database-dependent in most cases. So remember if you have other databases you have to take care for that to

00:12:56 do an IF statement around that. And of course, remember the check which is available for functional correctness in the Test Cockpit.

00:13:04 Yeah, so a lot of advantages. You can really consume HANA native artifacts and do all of the things that SAP HANA provides for you.

00:13:14 But on the other hand side, you have this pitfalls that you have to remember and have to take care about a lot of things.

00:13:21 You are responsible for a lot more things than you would be with Open SQL.

00:13:28 Here is a little sneak preview now on how you could also consume such an SAP HANA Database Procedure

00:13:36 and that is the consumption of an ABAP Managed Database Procedure.

00:13:39 So you see here on the right-hand side, it's just looking like a standard ABAP method call

00:13:48 and that's exactly what syntax it is. We will hear about that in the upcoming unit, in unit 2,

00:13:56 when we are talking about ABAP Managed Database Procedures. That thing here gives you exactly the same result as we have just seen.

00:14:04 So let me conclude. We have learned about ABAP Database Connectivity, the ADBC, to leverage the full features of SAP HANA.

00:14:13 We have learned about the pitfalls and we have given you a small sneak preview for the upcoming unit.

00:14:19 So see you for the ABAP Managed Database Procedures.

00:14:22 Thank you and good-bye. Bye!

WEEK 4, UNIT 2

00:00:12 Hi there and welcome back to week 4, unit 2: ABAP Managed Database Procedures.

00:00:18 Hi there, I'm not really surprised that they are back. I think they liked a lot the sneak preview on ABAP Managed Database Procedures.

00:00:26 I guess.

00:00:27 Okay, what's inside this unit? We will talk, of course, about ABAP Managed Database Procedures.

00:00:33 We will talk about the concept, how you can define and consume them. We will show you ugly runtime errors, and we will show you how you can prevent them.

00:00:43 We will also have a bit of time to talk about additional information, but let's just jump into where we are at the moment.

00:00:52 So in the migration phase, as discussed in the previous unit, we have left the path of Detect and Optimize and we are right in the Explore phase.

00:01:01 In the previous unit we have learned native HANA consumption using Native SQL or, in our case in the ABAP, the ABAP Database Connectivity (ADBC).

00:01:11 In this unit, we will focus ABAP Managed Database Procedures. Question is, what are ABAP Managed Database Procedures?

00:01:19 So ABAP Managed Database Procedures are a fully integrated way in the ABAP development environment and ABAP language

00:01:27 to call and create HANA procedures or database procedures.

00:01:35 This is one our most powerful tools to follow the Code-to-Data paradigm

00:01:40 because in this database coding, you can really fully leverage the power of the database,

00:01:48 especially the HANA with the parallelization and in-memory. So you can really do some really cool things down there.

00:01:54 This is done by using the HANA SQLScript language.

00:01:59 There is more information in the additional materials to this language. You will also get in touch, later in the demo, a little bit with SQLScript coding

00:02:07 but first of all, how can you create such an ABAP Managed Database Procedure?

00:02:15 So we had a pretty cool idea here to include it in our ABAP development.

00:02:22 So you just have to create a class and define a method like anything else

00:02:28 and with that method and input/output parameters based on ABAP types,

00:02:34 you can then, in the implementation of this method, specify that

00:02:46 this method is a database procedure and you can then write the SQLScript coding in that. This has big advantages.

00:02:54 You can easily call it from any other class, like a method in ABAP,

00:02:59 and of course Application Server ABAP will completely take care of the lifecycle management

00:03:06 and so you just have to transport the class with the SQLScript coding.

00:03:11 It's nothing else but a container, this method. And the procedure itself will be deployed and executed on the HANA database.

00:03:20 You don't have to take care for that. And, of course, we have it fully integrated in our ABAP Development Tools, by the way,

00:03:28 the same with the CDS views. You have to use the Development Tools here to create and maintain those ABAP Managed Database Procedures.

00:03:37 But enough of that. Let's just take a look at how you create them.

00:03:41 Yeah, thank you Jens. Let me jump to the system.

00:03:44 I will show you cool things like syntax checks and so on and so forth in this demo.

00:03:50 Okay, the first thing if I would like to use an ABAP Managed Database Procedure,

00:03:55 I have to make the ABAP language or the ABAP Application Server aware of that and for that in the public section of this class,

00:04:0 standard class I already prepared for you. I have to put an interface here.

00:04:09 That is just, as stated here, a marker interface. Then I have some type definition for internal table types and so on and so forth

00:04:18 And here comes my method. So this method will get some customer information, as the name suggests,

00:04:26 and does not look different than any other ABAP method, except I'm using here in the interface VALUE,

00:04:35 so like, for example, for RFCs. I can only pass parameters by value.

00:04:42 Okay, that much about definition. Let's jump to the implementation.

00:04:49 So currently, it's an empty method and it doesn't do anything and it's of course not yet a container for SQLScript.

00:04:57 For that, I have to prepare the implementation part here in a specific way. So, I will do that with a little help of my auto-completion,

00:05:08 so I say that is by DATABASE PROCEDURE and it already helps me with that. I would like to do that method as by database procedure.

00:05:18 I can say that is for my HANA database—the three letter acronym is HDB is for HANA database—

00:05:28 and then I can even specify the language. Here I put, of course, SQLScript.

00:05:34 And with that concept you already see that the AMDP guys already thought about supporting other databases and other languages.

00:05:43 Now something happened here already and that is the background coloring and that is allowed by the ABAP Development Tools.

00:05:55 They show you that you have embedded coding here. Same holds also for EXEC SQL if I would have EXEC SQL.

00:06:03 Background coloring is, by default, white, but you can configure that in the windows

preferences for the ABAP classes here, for the background coloring.

00:06:13 Okay and the only thing that I would like to add because I'm a fan of read-only database procedures,

00:06:22 that allows you even for having the full support of parallelization in the HANA database, so I'm using the options read-only.

00:06:33 Okay, I can right now not activate it because it's not allowed to have an empty body here,

00:06:41 but for that let me put in some meaningful SQLScript coding and as you don't want to see me type I have created here,

00:06:51 again a template putting all of the SQLScript coding in there. I mentioned already that there are syntax errors.

00:07:00 So here I'm informed that I'm using Dictionary objects, which I have not marked in the USING clause,

00:07:09 so whenever I would like to use Data Dictionary objects, or if I would like to call other ABAP Managed Database Procedures,

00:07:19 I have to put that in a USING clause. So what I'm doing here is snwd, the business partner table of our EPM data model

00:07:28 and the sales order table. And with that, syntax check is happy. I can activate that class

00:07:36 and I could, of course, now execute it. However, I would like not to do that directly, but in a small report.

00:07:45 So, what do I have here? I have a standard ABAP method as a container for SQLScript coding.

00:07:54 And if I would do syntax errors here, I would be informed that I have a syntax error

00:08:01 and that is really the syntax error coming from the database.

00:08:05 So I have, on the one-hand side, the ABAP Managed Database Procedure runtime giving you a syntax error on the ABAP level

00:08:11 but as well as compile-time errors from the database.

00:08:19 Okay, so you might imagine already how I can call that because we have had that sneak preview.

00:08:28 The consumption of an ABAP Managed Database Procedure is just a standard method call.

00:08:34 Here I have an object of the AMDP class and I'm just calling the method.

00:08:40 Let me just execute that and you can see it works perfectly fine, giving me information about the customers.

00:09:49 So, coming now to runtime errors.

00:08:55 For that, I would like to divide by zero and I'm putting here this undefined because I think not even HANA can divide by zero.

00:09:06 So I will do this stupid mistake. I'm declaring a variable as zero because if I would just put one divided zero here,

00:09:16 HANA will already tell me that in a syntax error, so I have to go here. It is one divided by

lv_zero.

00:09:25 I activate the class. And let me just execute it now again, the report, and you see...runtime error.

00:09:34 So why is she showing me a runtime error, you might ask? Let me go to the full runtime error.

00:09:43 You can find additional information on the database procedure here in this ABAP Runtime section, information about the database procedure

00:09:52 and you can find, in that section, the information passed from the database that I have divided by zero and that is, of course, undefined.

00:10:02 Question is, why not rather prevent them?

00:10:05 Yes, that's a good question, and of course, we have something prepared because we all don't like runtime errors.

00:10:12 The idea is quite simple and very easy so it's the same like catching

00:10:21 or raising an error from an ABAP implementation.

00:10:25 And since we have used the method of the class, as a container for those database procedures,

00:10:34 the concept was quite close and it's just RAISING and TRY...CATCH block in the calling report.

00:10:42 So let's just do it to let you see how this works. So let's get rid of this runtime error

00:10:50 and switch to the class. I go again to the definition of the class...of the method, sorry...in the class,

00:10:58 and the only thing that I have to add here is RAISING and the correct ABAP Managed Database Procedure exception,

00:11:09 We have more exception objects ready. This is also in your additional material, but enough for that. We have the RAISING,

00:11:17 we still have the definition with zero. Let's activate it and now let's go to the report which is calling it

00:11:27 and here I just need to add a TRY...CATCH block. And since I'm doing many typos, I've also prepared here a short macro

00:11:38 which is helping me (catch AMDP) and as you can see now I have a TRY...CATCH block and some meaningful coding here

00:11:48 which tells me what's happening in this example again. Also activating it and no runtime error, just the error information.

00:11:59 So, quite powerful tool or concept to really catch the exceptions of the database execution directly in your application server.

00:12:11 I really like it.

00:12:14 Even if you catch it now, of course the division by zero, you cannot really do anything against you, but really have to.

00:12:22 Maybe I could change a parameter here or something but, yeah, classical error handling.

00:12:28 Okay and so let me say something about additional information.

00:12:37 So there would have been a lot of things that we could have included already in this unit.

00:12:47 However, it's just so much more information and we would like to just guide you to this additional information.

00:12:56 And one thing that we would like to mention is the extension concept.

00:13:01 We have talked about CDS extension, the extension of CDS views, in the CDS.

00:13:08 Oh look her.

00:03:10 Yeah, a runtime error coming.

00:13:12 In the CDS unit, we have talked about extending CDS views and, of course, you are also interested in extending ABAP Managed Database Procedures.

00:13:22 For that we are using so-called ABAP Managed Database Procedure BAdIs (Business Add-Ins).

00:13:28 Those of you familiar with the BAdI concept, this is not really that complicated.

00:13:34 Just have a look at the SCN community for more information about AMDP BAdIs.

00:13:40 What you can do there is you really call an enhancement spot inside an ABAP Managed Database Procedure inside the SQLScript coding.

00:13:48 The other point that we would like to put your attention on is debugging in ABAP Managed Database Procedures.

00:13:55 Me, as an ABAP developer, I would be completely happy if I could put a breakpoint inside the ABAP Managed Database Procedure SQLScript coding directly and debug that,

00:14:05 but it is currently not possible and so not in a fully integrated way, but of course we have a workaround for you here.

00:14:13 Please find the SCN documentation and also this nice video tutorial on debugging ABAP Managed Database Procedures using external session debugging.

00:14:22 Thank you.

00:14:23 Okay, so let me just conclude this session by doing a short recap of the session.

00:14:34 What you have learned is that ABAP Managed Database Procedures are a very powerful tool to do code pushdown to follow the Code-to-Data paradigm,

00:14:44 that they are completely integrated in the ABAP classes, and the methods are just a container for them,

00:14:53 and with that you can really execute procedural coding down in the database and fully leverage the power of SAP HANA.

00:15:01 There are more additional concepts, but we really here wanted that you just get in touch with the basic concept.

00:15:10 We really invite you to our community where we are here to help you and give you more information.

00:15:16 But now for the next unit, where we will see again Core Data Services and how you can consume them in the domain-specific environment.

00:15:26 And now thank you and good-bye. See you in the next unit.

00:15:31 Bye.

WEEK 4, UNIT 3

00:00:12 Hi and welcome to the third unit of week 4: The Domain-Specific Usage of Core Data Services.

00:00:19 Hi there, I hope you enjoyed the ABAP Managed Database Procedure session

00:00:24 and in this session now we will take a look at the domain-specific usage of Core Data Services.

00:00:29 This means, what are the annotations in the Core Data Services and how you can consume Core Data Services views in SAP NetWeaver Gateway. Jasmin.

00:00:39 Okay, so let me just directly jump to the demo

00:00:46 because annotations can be best explained directly in the system demo. So first of all, why annotations?

00:00:55 We said that CDS or Core Data Services are extensions or extending the SQL, and with what?

00:01:06 With expressions. We've learned already that in the CDS View Definition unit.

00:01:11 We have also heard about the associations in the last unit of the previous week. And now the last point that we have mentioned

00:01:20 when we introduced the Core Data Services. That is the domain-specific metadata and that's exactly what annotations are.

00:01:27 Okay, we have already met our first annotation. Here in this ZCDS view. You can see that I have the ABAP catalog

00:01:36 and with that annotation—and annotations can be recognized by starting with the @ sign—with that annotation,

00:01:45 I pass the SQL view name for the given CDS entity inside the DDL source.

00:01:52 You can have two types of annotations: those that are valid for the complete view, so, for example, this AbapCatalog annotation here,

00:02:03 everything that comes before the defined view, and you can have annotations inside the projection list.

00:02:10 But let's start with the annotations for the complete view, for the entire view.

00:02:16 For example, typically, as an ABAP developer, your well used to the client concept,

00:02:22 so typically Data Dictionary objects are client-dependent Data Dictionary tables.

00:02:28 So that can be managed...well actually that thing is client-dependent but that's because the default value for the client handling is set to true.

00:02:39 So, I'm putting the @ sign and I also have auto-completion available here.

00:02:44 You can see here a lot of annotations coming up. So for example, I can use the ClientDependent and the value, the default value, is true.

00:02:54 I could put that as well to false, so that's the default value already.

00:02:59 So let me add something else that is typically for ABAP developers. What would you say?

00:03:02 Table buffer?

00:03:04 Why not? Table buffering.

00:03:07 So let me see whether we have some annotations for that, and yes, you have already seen that.

00:03:15 I can put the metadata information about the table buffer. So the buffering, that's the default value is switched off for that view.

00:03:24 I can have the buffering type and I have the number of key fields,

00:03:27 but let me just put SWITCHED_OFF, which is actually the default value if you don't specify an annotation.

00:03:35 Those are annotations which are valid for the entire CDS view or the CDS entity.

00:03:42 We also have the annotations which are only valid or make only sense in the projection list.

00:03:48 Here, for example, we have our well-known example of sales order ID, currency code, and gross amount

00:03:54 and I think you are familiar with the currency and amount with currency from the Data Dictionary,

00:04:03 so here I can put an annotation that this is a currency field.

00:04:08 So I have semantics, that one is a currency. And that is true and then I will relate the gross amount as an amount with currency.

00:04:19 So putting the @ sign again, I can say, okay that one is a quantity, that's a quantity, an amount with currency code

00:04:30 and I have to put the currency code and the currency code in my example. Sorry, it is, of course, just named currency_code.

00:04:39 I could give it an alias and then I would put the alias in there.

00:04:43 Okay, so much about annotation. And a good question is, what is in there for me?

00:04:51 Yes, so basically okay, we've seen the annotations. And since CDS is a completely new concept,

00:04:59 this is just the first step of annotations. There is much more to come. We will come back to that in the outlook session at the end of this week

00:05:07 But now let's see how we can consume such a CDS view, for example, in the SAP NetWeaver Gateway,

00:05:19 For that let's switch to the slides back and take a look.

00:05:24 So you've seen this architectural diagram already but now we have specified a little bit more

00:05:31 so we have the abstraction layer down here on the database between the database and the Application Server ABAP,

00:05:39 which are the Core Data Services. And above as abstraction layer now we have SAP NetWeaver Gateway

00:05:46 exposing the OData Service to a SAPUI5.

00:05:50 And what we will do now is consume this existing CDS view in our AS ABAP with SAP NetWeaver Gateway.

00:06:00 So let's just jump to the system and one Alt+Tab would have been better, but let's just go here

00:06:09 and take a look at the CDS view that we want to consume. We have already prepared that.

00:06:15 Here we have the CDS view which does quite a bunch of things but

00:06:23 it's basically categorizing our business partners by a specific behavior.

00:06:31 And to make it easier, we just add the count of the open sales orders invoices

00:06:40 and define on the count we have different categorizations: A, B, and C.

00:06:47 And we have another view on top of this which is consuming this existing view

00:06:52 and adding the business partner information for the company name. The result in the data preview looks like this.

00:06:59 We have the customer ID, company name, and different categorizations.

00:07:04 Okay, now we want to consume this view in the SAP NetWeaver Gateway.

00:07:10 For this, we open the...

00:07:16 Not the development object.

00:07:19 I would have gone to the SE93 correct? I'm always doing the same.

00:07:23 Here we already have the SEGW and we have already prepared a project for you in SAP NetWeaver Gateway

00:07:32 If you want to know how you can do this, we have the complete end-to-end guide to create this whole complete scenario available in our slides.

00:07:42 I will show you also later where you can find it.

00:07:44 So what you have to know is that we have an entity defined in this OData Service,

00:07:52 which is the customer information, which has three fields: CustomerId, companyName, and Category.

00:07:57 And by chance, this is exactly derived from the structure of the CDS view, so this I have done before.

00:08:05 And what we now want to do is take care for the implementation.

00:08:09 And the good thing here is you don't have to write any line of coding. This is done automatically by just telling the implementation

00:08:17 that he has to map it to the CDS view. And for that, of course, I should go to the Edit mode.

00:08:26 Now I can choose Map to Data Source and what I might say is I want a Business Entity.

00:08:34 And we have F4 help available for it, where we can now choose our Core Data Service view. Let's just take a look.

00:08:45 I forgot how I named it so it is zcdsv_cust_classification and that's it.

00:08:56 I now say Continue and now I have many windows so let me maximize it.

00:09:03 I see here the entity which is based on the structure

00:09:07 and the only thing I have to do now is just drag and drop those fields of the CDS view to the structure,

00:09:18 one-to-one mapping, quite easy. Save it and activate it.

00:09:24 Now there is a bunch of classes and implementations being generated. The name should be fine for us, so we just generate it,

00:09:34 We add it to a package, which is our week 4.

00:09:40 We press Save, add it to a transport, and we are good to go.

00:09:46 So now he's generating those classes, you see, and we have green boxes everywhere so everything is fine.

00:09:53 Implementation done. OData Service ready to consume.

00:09:57 We just need to go to the Service Maintenance of it, click on Register so that it's active.

00:10:07 It's a local...again sorry.

00:10:16 The package assignment, “OPEN*4”. I should remember the name of those four weeks now finally,

00:10:30 but that's fine. Now we have the package, the service is registered, and basically it's active.

00:10:39 Now we can take a look at our OData Service and we can do this easily if we click the Maintain button here.

00:10:48 Now just go to Gateway Client and we have here the request and response window where we can now just execute it.

00:10:59 And here we have the result, and it's XML, it's an OData Service,

00:11:08 but we can also navigate to our entity set to get that information.

00:11:12 I think we need to add the one option which is sap-ds-debug=true.

00:11:21 Just execute it again. It's looking much more prettier now.

00:11:26 And here we have our entity set, the customer info, and if you navigate to it...

00:11:35 Developer...Did I type it correct? I don't think so.

00:11:42 You don't know your password?

00:11:44 Oh come on!

00:11:46 Here we have all the customer info as you have seen in the data preview. Again, CustomerId, Name, and Category,

00:11:56 But XML doesn't look so impressive sometimes, so it would be nice to have it in a UI5 now.

00:12:05 And this is exactly what is described in this end-to-end guide. Also, how you create the Gateway Service

00:12:12 and also how you can bring ABAP Managed Database Procedure in the game. So this is all described in this document.

00:12:19 Take a look. The link is in your additional material.

00:12:23 But let's take a look at the demo. So basically here this is the result of OData Service with a company name and a category

00:12:30 and if we click one now, we see more information with the open invoices and open invoices

here in the list.

00:12:38 This is done by ABAP Managed Database Procedure, including the currency conversion on this side.

00:12:46 We also have U.S. dollar here now instead of euro the whole time. But we encourage you to take a look at it and try it on your own.

00:12:54 This is a really a cool thing to get in touch with the technology.

00:12:58 Okay, back to the slides and to Jasmin to give you an outlook and a conclusion.

00:13:06 Yes, thank you Jens. So we have now seen the full functionality which is currently available for Core Data Services.

00:13:14 We introduced Core Data Services, you remember. We have talked that it is an extension or an enhancement of SQL

00:13:21 with expressions, associations, and domain-specific metadata.

00:13:27 And the domain-specific metadata, we've heard in this unit, the annotation can enhance our data models with domain-specific information,

00:13:37 in particular, we have shown you the ABAP annotations. And you've seen how you can, in a domain-specific way,

00:13:45 consume the Core Data Services, here using SAP Gateway as OData provider,

00:13:50 providing us with an OData Service and giving us a nice little Fiori-like application that you can build on your own, on your trial systems.

00:13:59 And with that, and the link you can also try out on your system,

00:14:05 I would like to say thank you. See you for the next unit when we will have a special guest

00:14:11 talking about SAP BW, the Business Warehouse, and ABAP for SAP HANA, how they perfectly match together.

00:14:19 See you in the next unit.

00:14:21 Oh, well not me—our special guest.

00:14:23 Yes. Thank you and good-bye. Bye.

WEEK 4, UNIT 4

00:00:12 Hi! Great to have you back.

00:00:14 Welcome to week 4, unit 4, SAP Business Warehouse and ABAP for SAP HANA: A Perfect Match.

00:00:22 Since Jasmin and me are not Business Warehouse experts, we have invited Marc from the product management of Business Warehouse

00:00:31 to guide you through the features how the Business Warehouse is leveraging those new features.

00:00:36 So, Marc, welcome.

00:00:37 Thank you for the invitation. My name is Marc Hartz. I'm a product manager for BW on HANA, all the data warehousing topics,

00:00:45 and especially we are working and focusing on all of the mixed worlds between BW and HANA, all the latest functionality we brought down to HANA.

00:00:54 I'm happy to be here and to show you this integration.

00:00:57 Yeah, and there is also another reason.We want to give you a teaser to the upcoming openSAP course, Business Warehouse powered by SAP HANA,

00:01:07 where Mike and his colleague will guide you through all of the new stuff available with BW on SAP HANA.

00:01:14 But for me it's now relax and enjoy

00:01:19 and Mike will show what they have provided here in a nice demo and I hope you also enjoy it.

00:01:27 So, I have to do the work here for this session. This openSAP course will be right upcoming after this openSAP course.

00:01:36 We will focus there on all of the latest features that we have in our version BW 7.4, which is the latest

00:01:44 and has the greatest features which are optimized for SAP HANA and we will demonstrate from the data modeling, from the data acquisition and so on,

00:01:53 everything which is new and which are great features coming with HANA underneath the BW.

00:02:00 But today I would like to show you one major use case where we also invested heavily in the BW area and that's basically to bring down as much as we can,

00:02:13 logic down to HANA. And what we have in this scenario, or what usually is a bottleneck in BW, is that when we do ETL,

00:02:23 that means when we extract the data into the Business Warehouse, which is the data warehouse application.

00:02:28 We transform the data there, that mean we are consolidating, we are harmonizing the data to one common dataset.

00:02:35 We load it into the reporting area where we can do corporate reporting, dashboarding, and all of the stuff which is necessary in the reporting area.

00:02:44 And in this transform area, we do have usually the bottlenecks and this is coming from that

there is application logic,

00:02:54 which means we are transforming your data to a certain logic. In the past, without HANA, this logic was totally being processed in the application server.

00:03:04 So this means that we took the data out of the database, processed it in the application server, and rolled it back. And this was time consuming.

00:03:10 Exactly. If you think about that, this is what we have talked about in week 1, week 2, week 3, all the time with the Code-to-Data paradigm

00:03:20 and here we have now a great example for it, hopefully.

00:03:25 Right, so BW is fully NetWeaver-based, so we can leverage all of the functionality we have from the latest ABAP versions.

00:03:34 We can fully leverage everything to bring the logic down to HANA

00:03:38 and this use case is really to show you how we integrated the ABAP Managed Database Procedure in the data flow area of BW.

00:03:46 So during a data loading process, means from one InfoProvider to another one, but this is now BW terminology already, so keep it very simple here.

00:03:56 We have different data containers and what you are doing between them is you transform the data to a common view to a business model.

00:04:04 And this process was, in the past, done in these transformations. We have standard transformations or the possibility for complex things to write custom code.

00:04:13 This custom code was depending on how good or how bad it was written, of course, slow or it could also be fast if it is very well and efficiently written.

00:04:25 It was delimited that you have to do it in the application server.

00:04:28 Fully right. It was fully processed there.

00:04:30 And what we have as a possibility here is really to integrate this ABAP Managed Database Procedure.

00:04:35 We do have it on another functionality as well, so we also have SAP HANA analysis processes, which is a different feature

00:04:43 but also able to leverage this technology. We will cover that of course in our openSAP course, but not now.

00:04:49 For now I would like to show you a simple example on how we can transform data between two data containers.

00:04:55 The advantages of this, so if you ask yourself why should I do that, it's very obvious with this technology. We are as close with logic as we can be to HANA.

00:05:07 This means that we avoid this round trip of the data. And one major use case and benefit is of course we are integrated in the whole BW process management.

00:05:17 This transformation process is scheduled by BW, the whole transportation management is included there, so that's really the integration advantage

00:05:26 And you're in the ABAP environment where you have been already with the classical BW.

00:05:32 You are fully right and this is exactly what we would like to show you

00:05:36 in a demonstration, in a short demonstration here, so let me jump into the system

00:05:42 to show you the functionality which is available with our BW 7.4 SP8, which is the support package that we released this functionality in.

00:05:53 You see also with the BW guys, we are not here in the traditional BW area, so also BW is moved to Eclipse.

00:06:00 So certain key functionalities are available in Eclipse, and for others we still have the integration possibility

00:06:08 so we can call an SAP GUI to work in the BW transactions.

00:06:13 So if I just call rsa1, you will see that we are fully here in the BW environment.

00:06:22 Okay, so this is your classical transaction.

00:06:26 Yeah, so what I just called is the BW Workbench, where you can model everything in BW, your data flows, your info providers and so on.

00:06:34 So this is where you usually BW guys are really working.

00:06:37 And what you are seeing here, what we had prepared is basically a data container.

00:06:43 It's a DataStore object, a very common data container in BW, and from there we are loading the data

00:06:50 from a source system, which is indicated here, so that's a data source. This is our interface into source systems.

00:06:57 It could be ERP, it could be a non-SAP application, could be whatever. We have many interfaces here in the BW world.

00:07:04 This is the interface for data loading. And in between, you see the transformation here

00:07:10 and the transformation is the place where you can put and where you can script your logic.

00:07:14 What you see here is we have on the left side the structure of the source, on the right side we have the target structure, and in between you can do the mapping.

00:07:23 Okay, and the source table is the Enterprise Procurement sales order. This is one you might probably know from our other exercises.

00:07:31 Right. The good thing is we will also use this model in the openSAP BW course, so we will show you how you can do it based on this model to report.

00:07:40 We will show you all the latest BI functionalities and so on.

00:07:43 But coming back here, the system was also doing a one-to-one mapping

00:07:47 and you see we have here fields which are indicated by a time stamp.

00:07:52 And the time stamp is, for BW, something we would not like to have there

00:07:57 because in the upper layers of BW, it would mean that you work in reporting with the time stamp

00:08:03 and this is not usually what our managers would like to do.

00:08:05 Not really.

00:08:06 So they would like to have, you know, they would like to drill down by year, by month, by quarter, by period.

00:08:12 Therefore, we have to transform this time stamp into date format. And this is exactly what we can do here.

00:08:19 In the past, I would call now an expert routine in ABAP, but in the ABAP application layer, and would write my exit there.

00:08:27 What is new in this functionality is that we can...I have to change to the edit mode first...that we can call the routine here.

00:08:39 So we have here the exit type which is indicating to us that this is something down in HANA.

00:08:47 So it means that it is a HANA exit type. It will call here the possibility to write an ABAP procedure.

00:08:54 This ABAP procedure is already generated. You see we have the source structure, we will have the target structure and the possibility here to implement our own code.

00:09:05 So this is just an example of the loading script here already commented out.

00:09:11 The BW is generating this, exactly.

00:09:14 And on the method implementation on top already, the syntax is created, correct?

00:09:19 I'm just seeing that here: BY DATABASE PROCEDURE FOR HDB LANGUAGE SQLSCRIPT.

00:09:24 So and I also learned something in your course.

00:09:28 I can now use this shortcut and template. So we place there already the logic we would like to apply here and now to transform the data.

00:09:38 So let me just call the logic here. We have here a procedure already and this procedure is doing exactly what I described.

00:09:47 It's taking from the source structure the time stamp and converting it to the date format we are using in BW for reporting.

00:09:55 If we now activate and run it, it will transform the data in a very fast manner also

00:10:00 but this is something I would not like to show here because we have many, many sales orders behind.

00:10:06 If you would like to see really the performance results and the impact of this in BW,

00:10:13 I would be very happy to see you again in the BW on HANA course where we will exactly compare the different methods.

00:10:21 I'm curious. I will join.

00:10:23 I want to, just to end here, the short demonstration. You see now the one-to-one mappings are gone. So we have now here the core.

00:10:31 This is now something down in HANA which is scripted. We have different possibilities, but in this case we are using the ABAP Managed Database Procedure

00:10:38 to bring this functionality down to HANA to avoid the round trips in BW.

00:10:43 Perfect. Great. Nice to see ABAP Managed Database Procedures in action

00:10:50 and thank you very much for joining here and showing us the demo and some introductions to the upcoming course.

00:11:00 Now let me switch back to the slides and just give you an outlook for the already last unit of this course

00:11:09 where we will do a recap of what has happened in the last weeks, what we've showed you,

00:11:13 and of course, give you a lab preview of what will be in the upcoming releases of SAP NetWeaver.

00:11:21

00:11:22 Stay tuned and bye-bye.

WEEK 4, UNIT 5

00:00:12 Hello and welcome back to the last unit of this course.

00:00:16 Last unit already, that is sad. Okay, so last unit typically means looking ahead, outlook, lab preview.

00:00:28 Before we do so we would like to recap what has been in this openSAP course: ABAP Development for SAP HANA.

00:00:37 Afterwards we will also guide you to where you can find more information.

00:00:41 In the first week we have introduced SAP HANA—well, a bit about SAP HANA—

00:00:48 and we have talked about how ABAP Application Server welcomed the SAP HANA.

00:00:54 We have talked about the in-memory capabilities of this column-oriented database

00:00:59 and what special things you have to take care of or have to keep in mind as ABAP developers when working with SAP HANA.

00:01:09 In the fourth unit we have talked about the Code-to-Data paradigm.

00:01:14 The classical approach of ABAP application development has been to get all of the data that you need from the database layer up to the application layer

00:01:24 and do all the calculation and the business logic in the application layer, and finally transfer the results to the user interface.

00:01:33 With the in-memory capabilities of SAP HANA, that picture changes dramatically.

00:01:38 We now encourage you to push down or delegate your data-intensive calculations to there, where the data resides, to the database directly.

00:01:48 With the in-memory capabilities of SAP HANA, that works. You can really do the business logic or the application logic in the database

00:01:57 and transfer only the results that you really need in the application server, for further calculations

00:02:03 or finally totransfer them to the user interface. And what have been the tools?

00:02:10 Okay, Code-to-Data paradigm. Check. Now let's switch to a slide that you probably might remember.

00:02:17 In the Detect phase, you have learned what is important for a HANA database migration. With every database migration there can be

00:02:27 functional correctness issues with your code if you have relied on database-specific code.

00:02:32 There we have introduced you to the ABAP Test Cockpit as a new tool to do those static code checks

00:02:43 and of course also to find optimization patterns that you could find.

00:02:51 But we also introduced to you that runtime data is very important to find the right spots in your custom coding to optimize

00:02:58 and this is done with the SQL Monitor, the new tool here.

00:03:03 You have learned how we would recommend you to do the process afterwards and that's exactly the process that we have done also internally,

00:03:14 so we have checked our code for functional correctness.

00:03:17 We have checked it with the static code checks for optimization potential and added the runtime data of the SQL Monitor

00:03:26 to really see where we can first optimize and then did the iterative approach

00:03:31 and then we used all of the new features to optimize our coding internally and for you.

00:03:39 Yeah, so you talked about the Detect phase, let me go a bit deeper into the Optimize and Explor

00:03:47 that has been the topic of week 3 and beginning of week 4.

00:03:53 We started with optimization of Open SQL and with the Core Data Services

00:03:59 And in the Explore phase, we talked if you would really like to squeeze the last percentage of performance for your ABAP applications,

00:04:08 you have to for example, use Advanced Database Connectivity, the ADBC, the successor of EXEC SQL and Native SQL usage

00:04:17 and last but not least, the ABAP Managed Database Procedures.

00:04:22 So here is the diagram again about performance improvements versus code adjustments or code complexity.

00:04:30 First step. You don't have to do anything. That comes practically for free our transparent optimizations like

00:04:38 exchange protocols between the database and the application layer, like fast data access or table buffer enhancement.

00:04:46 They come for free. You don't have to do any code adjustments, but also maybe performance improvements are only minor.

00:04:55 So if you would like to get more, I mentioned already Open SQL, we have introduced here the new features of Open SQL to follow the Code-to-Data paradigm

00:05:06 and we have shown you advanced view definition capabilities with the Core Data Services views

00:05:11 after we have introduced, of course, this very new topic, the Core Data Services in general.

00:05:18 So that gives you more performance hopefully, but also more code complexity and you have do adjustments.

00:05:27 So, last step and squeezing the last percentage in performance for the perfect performance boost,

00:05:32 you would like, for example, to use ABAP Managed Database Procedures, the containers for SQLScript coding.

00:05:40 You remember you have an ABAP class method and you directly embed SQLScript coding in that ABAP class method

00:05:46 and call it just as any other class method you would do.

00:05:51 And Native SQL with ADBC, we have showed you how cumbersome that might be.

00:05:56 But if you would like to use the full functionality, the complete feature set of SAP HANA, you might have to go that way.

00:06:03 Exactly so. But now let's take a look at those new entities we have brought into the game, and here especially the Core Data Service views

00:06:12 and ABAP Managed Database Procedure, which are completely and natively integrated in the ABAP development environment and inside the ABAP language

00:06:23 This is the basic idea in this whole concept from us, this so-called top-down approach, that you can manage, create, and maintain

00:06:32 all of these new features directly in your ABAP environment and you don't have to take care for the whole lifecycle management and deployment of those entities.

00:06:41 The ABAP Application Server will completely take care of this

00:06:45 and also you are able to transport all of those new entities directly with your well-known ABAP transport system, the CTS.

00:06:54 Yes, and HANA views and procedures get deployed for you just when you activate them.

00:07:00 Exactly. You don't have to take care of it.

00:07:03 So the question is, what's coming up next?

00:07:06 And there are many things.

00:07:08 So here is a nice little lab preview. So really, the looking ahead part of this presentation.

00:07:16 Of course we especially, as AS ABAP Integration Team for SAP HANA, are working hard to get the best integration between the ABAP Application Server and the SAP HANA database.

00:07:29 So we really encourage all of the development colleagues to work hard for you.

00:07:34 So here are a couple of pillars that we would like to emphasize.

00:07:39 So the first pillar here is we are still working on the transparent optimizations,

00:07:46 those that you don't see as developers but hopefully experience as application users.

00:07:50 Still working on the fast data access protocols or protocols in general,

00:07:55 the SELECT ... FOR ALL ENTRIES and also data exchange for example, for stored procedures

00:08:00 and stored procedures here are the ABAP Managed Database Procedures.

00:08:05 Okay, for the second pillar, our code pushdown topic, we are working hard to continue this top-down approach for you

00:08:13 and to add more features for the Core Data Services views. For example, we are introducing many more annotations,

00:08:25 that, for example, you will be able in the future to create BW Cubes directly by writing annotations inside your Core Data views,

00:08:34 so that you can have the OLTP and OLAP conversion much more integrated in one data model.

00:08:42 You will also be able to create annotations for SAP NetWeaver Gateway so that an OData Service is directly created

00:08:52 out of this Core Data Service artifact and you don't have to take care of all the configuration or anything in another transaction.

00:09:03 What's also important for us is that we enhance the database procedures, for example, the end-to-end debugging from a consumability point of view.

00:09:11 This is one important topic. But also we want to enable you to write SQLScript and also be able to consume it as a view.

00:09:18 This is a topic that's also on our list.

00:09:21 And type harmonizations that you will be able to use native HANA data types much more integrated in our ABAP server.

00:09:31 Yeah, but it's not only about Code-to-Data, but it's also about services.

00:09:35 So you as ABAPdevelopers, you are well familiar with all of the services provided by the ABAP Application Server

00:09:44 like, for example, enqueue services and so on and so forth.

00:09:46 So if we encourage you to do the code pushdown or theCode-to-Data paradigm, you say, I need the services on the database too.

00:09:56 Therefore we are also doing the so-called service pushdown. So we really would like to provide, on the SAP HANA database,

00:10:05 things like number ranges, GUIDs, the enqueue service I mentioned already, or calendar functions.

00:10:12 And last but not least, we have the lifecycle management and operations pillar

00:10:17 where we take care that the operations of SAP HANA and the Application Server will work as smooth as it can get.

00:10:26 Zero downtime management is here a very important topic,

00:10:29 because we know that the migration process can be hard for you so we want to make it as easy for you as possible.

00:10:36 We are also taking care that you can specify the workload management between ABAP Application Server requests and SAP HANA requests,

00:10:45 that you can have your prioritized business processes running smoothly while doing analysis in parallel.

00:10:54 So there is much to come. Stay tuned for that.

00:10:58 Yeah, and with that we would like to give you more information or where you can find more information.

00:11:04 So we hope you enjoyed this openSAP course and that you say, Hey, I like this course

00:11:10 and I would like to learn even more about ABAP development for SAP HANA.

00:11:14 And you can get more information, for example, in the SAP Training and Certification Shop. You can find the link here on the slides.

00:11:23 There is even in the Certification Shop an ABAP for SAP HANA certification available.

00:11:30 There is also the SAP HANA Academy. There is a special page for the ABAP development for SAP HANA,

00:11:39 and we have mentioned that a lot of times, the SAP Community Network (the SCN)

00:11:45 where you can find information about the reference scenario, video tutorials, and we even have an ABAP YouTube channel for the ABAP development

00:11:53 and for the ABAP development for SAP HANA.

00:11:56 Of course, not mentioned on this slide but very important for this openSAP course: If you have questions, for example, concerning this course but even beyond,

00:12:07 you can also have the openSAP Forum. You can meet, Jens, me, or our colleagues from the integration team on the openSAP Forum.

00:12:18 and last but not least, join us on the ABAP for SAP HANA Community on SCN.

00:12:25 See you there. hopefully. But not only seeing you. You can also meet us live on the SAP HANA CodeJam events.

00:12:34 They are all around the world. There is also the possibility for you to host them

00:12:38 and we will come with systems, demos, and experts for the SAP ABAP for HANA CodeJam.

00:12:46 It will be the experts from the integration team so Jasmin, me, or my colleagues.

00:12:54 How to host such an event or where new events take place are all available on those links here in the slides.

00:13:03 Yeah, to be honest, there are even other CodeJams also available, SAP CodeJams for example, on the ABAP Development Tools in Eclipse—

00:13:12 we have mentioned that already in the first week—or also about SAP HANA Gateway, Fiori, and so on and so forth.

00:13:20 So those CodeJam events are really cool because they are from developers for developers.

00:13:25 It's not the classical business stuff. We are talking about really cool development tools, development features.

00:13:35 Yeah and last but not least, it's time to say good-bye.

00:13:41 First of all, we enjoyed a lot this openSAP course and talking about ABAP development for SAP HANA.

00:13:49 I hope you enjoyed it the same way we did,

00:13:52 as much as we did, and it was really a pleasure delivering this openSAP course.

00:13:58 So again, meet us either in the Forum, first stopping point, or on SCN, or live in person on this SAP CodeJam

00:14:08 We wish you good luck with the final exam. So, good-bye.

www.sap.com