







Next, let's take a look at declaring complex data time vegetables are just as with normal vegetables.

We used the data statements again, but complex data types do require some additional information.

And in my mind, the really not complex study, but, oh, we have to do is make sure we declared the

length and if required, the decimals and the value.

So you can see in this example, we have to text with a type string and we're giving it a value of text.

Now, specifically, we string the system will automatically determine the length of the string.

So we no need to specify the length with this one.

But just keep in mind that it is a complex data type in that there's a bit of overhead with the system

in that it has to work out the length for us in the background.

It's not fixed like with the character data type.

And as we can see where specifying the value count time pi specifying value three and then our packed

decimal number, you can see with specifying the length of a with two decimal places and assigning an

initial value.

So this gives us 30 nos to the left of the decimal point and two numbers to the right.

Then we have to sign as well.

Now let's have a look at declaring our own data times, so to do this, we, first of all have to use

the statement times and this allows us to declare our local data times.

The data types we declare are only available in the local program.

So the syntax goes times and we give it a name.

Then we say what type our type is going to be and in the specification, straight down to the examples

we can see we're creating a type age here of type integer employee name of type and the length weight,

salary type, identifying with two decimals, all fairly simple stuff.

A couple of things to keep in mind, times cannot have a user defined default value.

Think about it when declaring a type here.

We're not declaring data objects, variables, just the types.

And as I mentioned, types are local to the program.

If you want to declare global data types, then that's where we must use the dictionary.

So you would go to see someone create a brand new type and specify the various parameters taken, and

then you can use it in any program.

So now we know how to declare types.

Let's have a look at declaring scripture types, a structure type is really a group of individual types.

It allows us to encapsulate multiple data elements into a single structured type.

So as we can see with the example here, we use the types with the Kawan and then we start our structure,

we use begin of and then give it a name.

Then we include the individual times that we've declared and then we finish off declaring this type

structure with the end of and the name we had, I should also have the full stop here.

Now, just like wouldn't normal single data types that we can create, we cannot store any data in this

structure as yet because we haven't used the time to declare any internal table or single line work

area.

It is only a type at this stage.

When we do want to store some data, that's when we have to move down and then declare a data object

based on our type structure.

So here we can see data employee type string employee.

So we're referring to this whole type structure here.

We can then store individual text within our data structure so the individual feels he has a name in

full name we can populate just by referencing the type of reality the data statements here.

So it's the employee Dosch field name, surname in our case.

Same thing here.

Employee Dush full name.

So that will store the surname of full name within a structure that we can then use with our programs.

Next, let's have a look at declaring Constance, we use Constance quite a lot in our shop and they

do exactly what it says on the tin.

A constant is a constant variable that never changes.

So straight onto the example, we have a constant of country code type C with a value of GBE and specific

for constants here.

If you think about it, it makes perfect sense.

We must specify a value.

There's no point trying to declare a constant that doesn't contain anything.

Another example, pie type float with a value three point one for remember, with a float, well, with

decimal places, we have to put our numeric value in single quotes.

Now, let's have a look at some existing system data variables that we can make use of in our program.

There's a structure within S&P that's already been defined and that our programs use automatically without

having to implement any code at all.

There's a structure called s Why City and if you have a look at this structure in the data dictionary,

you will see that it contains many system maintain values.

So let's do that.

I'll open Opus 11 and we'll have a look at the structure table.

Here we have the same guy, we go into C11.

What we need to do is go on to day to type and we're keen ways to.

He would go.

So this is the structure that all of our programs use automatically, and you'll see that there's lots

of what are called obsolete fields or components here that we don't need.

Well, we won't use, but there are many, many more that we do reference quite a lot in our programs.

Scroll down.

You can see there's many components he.

So let me move this across.

If we have to look at the slide here, I do mention that there are many components, many are obsolete.

Some examples are datum for the current date of the application server.

There's Sub Arcy.

That's the return code field that we use for referencing whether something has been performed successfully

or not and so on if we bring this back up.

A message for datum, here it is.

So that's the component that we reference in our program, we do a search for some RC news that an awful

lot so you can see all these fields are defined.

And if you think there might be something that you want to check out, come into this structure, have

a look and see if you can make use of them.

One last thing that you should know.

If you're familiar with Umbach, we don't actually reference these objects by saying SWG Dosch and then

the component.

What we actually do within our programs is reference them by using the S y dash, then the component

names.

So for that return, code example, it would be swang dash soap RC for the datum.

S Why does the U.N..