Now you've got your first program under your belt. Let's expand on the program a little bit and add in a few more ABAP statements. So, let's step back. Back to our coding screen. And what we're going to do now Is access some of the data records that we created when we built our first table in the ABAP dictionary.

To do this, the first thing we have to do is include the table statement into our program.

Now following tables, we key in the actual table name that we created.

**TABLES z\_employee\_list.**

When the program is executed, this table statement will create a table structure in memory based on the structure that we defined in the ABAP dictionary of our zemployee list table. The table structure will contain all the fields that we created. And this will allow us to read records from the table, and store them in this temporary structure for our program to use. So, to get these records into our table structure, we're going to use the select statement.

**SELECT \* FROM ZEMPLOYEES.**

**WRITE ZEMPLOYEES.**

**ENDSELECT.**

the select statement is a loop. So, our loop starts with a select and ends with end select.

we are going to say write a whole row at once to the output window. Now we have our select loop, we're going to write out every row of our table and display it to us in the output window.

Now if we try to print this report, it could well be that there are too many columns, or too many characters to actually fit on a standard piece of A. This will be determined by the width of the paper specified in the printer settings. So don't count on your output being in the right format for the reports you create.

What I suggest is you use an addition to the report statements where you specify the width of each line.

**REPORT Z\_EMPLOYEE\_LIST\_01 LINE-SIZE 132.**

Let's now enhance our program a little bit more by adding some specific formatting to the write statements. So, what we're going to do is insert a line break. At the start of every row that we output within our select statements.

**WRITE / ZEMPLOYEES.**

**ULINE.**

uline simply means underline. Writes a continuous underline in a new line.

**SKIP.**

To display an empty line.

**SKIP [number].**

To display n number of empty lines.

when you want to specify an individual field, you use a dash, and then the actual field name. So, we have surname, forename, title, date of birth

**WRITE / ZEMPLOYEES-SURNAME.**