

TÍTULO ABREVIADO DO DOCUMENTO

Online Course: Introduction to Software Development on SAP HANA

Mafalda Sousa Rosado

(Activity report)

Abstract—Programming logic has been changed over the years to accompany technological innovation. Within SAP, on a technical point of view, the main innovation in the last 5 years has been the developing and launch of SAP HANA, an in-memory database that increased the overall system performance and changed data storage paradigm. This report will provide an overview on SAP HANA how it works and its main features. The purpose will be acquiring the knowledge necessary to develop native SAP HANA applications.

Not really an ABSTRACT of the document

Index Terms—Systems, Applications and Products (SAP), High-Performance Analytic Appliance (HANA), SAP HANA Extended Application Services (SAP HANA XS).

What?

The whole document is a description of the "course", not a description of the activity of "Taking The Course" /

1 INTRODUCTION

THIS course was divided into six weeks each one addressing certain topics related with SAP HANA:

- Week 1: Developing Applications for SAP HANA
- Week 2: Database Tasks, Loading, and Modeling
- Week 3: SQLScript Basics
- Week 4: Exposing and Consuming Data with OData
- Week 5: Exposing and Consuming Data with Server-Side JavaScript
- Week 6: SAP HANA Advanced Development Options

In each week the tasks that should be perform were to see the available videos, make the self-tests to guarantee the correct understanding of the contents and finally make a graduated homework. At the end of these six weeks a final exam should be conducted which worth 50% of the grade. The goal behind this course was to understand what is SAP HANA,

- Mafalda Sousa Rosado, nr. 55835,
E-mail: mafalda.rosado@tecnico.ulisboa.pt,
is student in Information Systems and Computer Engineering,
Instituto Superior Técnico, Universidade de Lisboa.

Manuscript received July 08, 2014.

be familiarize with the available functionalities and discovered how can the features be implemented.

2 DEVELOPING APPLICATIONS FOR SAP HANA

The first week focus was to understand what is SAP HANA, prepare our system to start developing (download, install and configure all necessary tools) and ended with a simple exercise to understand some basic concepts.

2.1 SAP HANA

What is SAP HANA? The first thing that anyone would say is that it is an in-memory database, but is not only that. SAP HANA is an application server, a programming language and a dedicated web server, so it gives the possibility to build a complete application end to end, from the schema, to the database tables, to the data-intensive logic, to the service enablement and even up to the user interfaces.

Another question usually arises when talking about SAP HANA: what is an in-memory database? An in-memory database is a database in which all write and read operations happen in memory. That type of processing leads to software optimization of the database

(1.0) Excelent (0.8) Very Good (0.6) Good (0.4) Fair (0.2) Weak	ACTIVITY					DOCUMENT						
	Objectives x2	Options x1	Execution x4	S+C x1	SCORE	Structure x0.25	Ortoqr. x0.25	Gramm. x0.25	Format x0.25	Title x0.5	Filename x0.5	SCORE
	1.0	0.5	2.0	0.4	3.9	0.15	0.15	0.15	0.2	0.5	0.5	1.65

engine, since it is guarantee that the processing will always be faster than within a normal database.

When talking about SAP HANA as an in-memory database some other advantages must be taken into consideration: this database was designed to take the most advantage of the modern CPU architecture, in other words, it is prepared for massively parallel processing. Another benefit is that no changes in programming logic are necessary since the database platform manages the queries made, spreading them automatically into multiple CPU's or server instances. However the major gain when using this database is the ability of storing data in columns, but why is that so important? By putting all the same data types together the data will compress much better and every column is a secondary index, so there is no need to build indexes or to materialize indexes and store the data multiple times in different ways.

And because of this high-speed access, it is possible to do things that we would not be able to do with a traditional database, such as apply analytical analysis directly on top of the transactional data without having to copy the data, materialize aggregates and materialize views of the data.

2.2 SAP HANA XS

SAP HANA native applications are those applications that are built completely to run inside of HANA. The ability to do that is related to the existence of an application server built in HANA itself and named SAP HANA XS. XS is not only the abbreviation of Extended Application Server, but also stands for extra small meaning that the application server layer is lightweight. The reason behind the lightweight application server is the pushing down of the data-intensive processing into the lowest layers of the database to leverage the power of the massively parallel processing, the in-memory and the columnar store. The improvements on devices were also important to the reduction of weight of the application server, since the presentation logic has been moved to the client side, because the devices have power

enough to become responsible for the interface rendering and the handling of the user interaction events. So just a couple of things need to be handle by the application server, such as control flow logic, service enablement and some validation logic. This Extended Application Server has another great advantage, since it is totally integrated with HANA no data types transformation is needed when the application communicates with database and vice-versa.

It's also important to know that it is possible to use another application server in addition to the one that's embedded inside of HANA. Using ABAP, Java, .Net with an open interface like JDBC or ODBC to connect to SAP HANA we will develop non-native SAP HANA applications that still leverage the power of HANA without the need to switch the server application.

2.3 SAP HANA Studio

SAP HANA Studio is the primary application development tool and it is based on Eclipse, this way we have a development tool with Eclipse look and feel. This tool offers dedicated perspectives representing the different roles that might work on it (e.g. Modeler, Administrative, Development and Debug) and they consist on multiple views. The one that will be focused in this course is the Development perspective that consist on the following views:

- Project Explorer: shows the local content;
- Repositories: shows all the server content;
- Navigator or Systems: is the only view shared between this perspective and the modeller perspective. It shows the systems configured and the user ID that is used to connect to it, inside each systems it is possible to found three folders:
 - Security: has the ability to create or change users and to display or create roles;
 - Catalog: shows the database catalog;
 - Content: shows the objects in the HANA repository that can be edit in this tool.

The authorizations in the content repository are managed at a package level, it is not pos-

sible to define authorization at a object level. But what is a package? The package builds a folder structure and it is responsible for the namespace concept which allows that SAP and client can create a table with the same name, because these two tables will be created in two different packages.

2.4 Development Resources

SAP offers two major development resources to help their users. One of them is the SAP Community Network (SCN) which is a portal where it is possible to learn, deepen the knowledge already gain and exchange ideas, find or provide solutions to problems. The second is the SAP HANA Developer Center, in this one more technical documentation can be found like APIs, source code documents and how-to guides. In addition this resource also provides various pieces of free software downloads, for instance SAP HANA Studio and SAP HANA Client.

2.5 Amazon Web Services (AWS)

SAP partnered with Amazon to use AWS as a cloud provider for the system images. The costs for using these systems are charged by Amazon, SAP does not get any money from it, and they depend on how much the system is used. An option to start, stop and suspend the system is available and should be used in order to save some money, since AWS charges by the hour for system uptime.

2.6 Application Files

There are two core application files that must be created in order to start the development process. The first is the .xsapp file that basically is a placeholder. This file will be completely empty and will denote to the system that the package and all subpackages will be XS-based applications. The second file needed is the .xsaccess file which is responsible for the exposure of web content and the control of access and authentication mechanisms.

3 DATABASE TASKS LOADING MODELING

The objective of this week was to be able to create database objects and understand what they are. Also an overview of the different types of views that exist on SAP HANA will be made.

3.1 Database Schemas

The schema is a mandatory database object containing all the other catalog artifacts. But it is not only used as a group mechanism but also as a authorization control mechanism, since we grant access to a particular schema and then the objects within that schema inherit those authorizations. The schema file has to be created with a .hdbschema suffix and it is a plain text file with the schema name and its authorizations.

3.2 Database Tables

A table is a set of data elements that are organized using columns and rows. The table file must be created with .hdbtable suffix and contains the following information: schema name, table type (column, row), table columns, data types and primary key. This object name is composed by <package.path>::<TableName> and its meta-data contains the constraints on table or in particular columns.

3.3 Database Sequences

Database sequences generate an incremented list of numeric values to identify tables, columns or rows and it is used by applications to reference sequence objects, generate primary key values, identify table columns/rows and coordinate keys across rows/tables. This object name is created using suffix .hdbsequence and package prefix. The keywords that can be used in this file are the following: schema, start_with, nomaxvalue, nominvalue, cycles, depends_on_table and depends_on_view.

3.4 SQL Views

A view is a virtual table based on the dynamic results returned in response to an SQL statement. This file has the mandatory suffix .hdb-view and contains the following information:

- Schema name;
- SQL query:
 - JSON notation;
 - select from;
 - left outer join on;
 - order by;
 - depends_on_table/view;
- Joined table names:
 - <package.path>::header;
 - <package.path>::item.

3.5 Authorizations

SAP delivers some default roles that can be automatically attributed to users, such as CONTENT_ADMIN, MODELING and PUBLIC, however it is possible to define specific roles. These roles have to be created in a file with .hdbrole suffix and define the privileges that an user with that role will have. To grant a role to an user _SYS_REPO procedure must be used with rolename and username parameters.

3.6 Single File Data Load of CSVs

Three files are needed to load a single data CSV file to SAP Hana.

- Source file: contains the values to load into the database table separated by commas. The number of columns in CSV file/target table and data types in target table/CSV columns;
- Import model: Specifies source file type and defines the target table in the database ("item"). This file has the mandatory file suffix .hdbtim;
- Import data: Specifies the source CSV file and connects the CSV source file to the data load model file. This object has to be created with .hdbtid suffix.

3.7 Attribute Views

Attribute view models an entity based upon relationships of multiple source tables and can

contain columns, calculated columns and hierarchies. To build an attribute view is necessary to define the sources (tables) and the relationships (join fields in tables) to make possible to get a consistent definition of the overall entity. Another important thing to define is the properties of a view column (e.g. key attribute, drilldown-enabled, hidden), since they affected the overall usage. Last but not least, a new calculated output field must be defined specifying the data type, field width, scale and the rule definition or using formulas and conversions in an existing field in the view.

3.8 Analytic Views

This kind of views leverage the computing power of SAP HANA to calculate aggregate data. They can contain two types of columns which are attributes and measures. Some important concepts should be understand in order to define and use analytic views:

- Data foundation view: represents the tables used for defining the fact table of the view and shows all the fields that can be incorporated into the final model;
- Logical join view: represents the relationship between the selected table fields (fact table) and attribute views and displays only the fields chosen to be include in this model, as well as the restricted and calculated measures that have been defined;
- Semantics view: where it is possible to classify columns and calculated columns as attributes and measures and create variables/input parameters and hierarchies and assign variables to the columns.

3.9 Calculation Views

Calculation views are composite views and can be used to combine other views and can perform complex calculations, which is not possible with other views. Calculation views can be created using a Graphical editor or the SQL Script editor. Graphical calculation views are created using the graphical editor and have the following capabilities:

- Can consume other analytical, attribute, other calculation views and tables;

- Built-in union, join, projection and aggregation nodes;
- Provides additional features like distinct, count, calculation and dynamic joins;
- No SQL or SQL Script knowledge required.

Scripted calculation views are created using SQL Editor and usually are used in complex calculations which are not possible through graphical approach. This subtype has the following capabilities:

- Can be regarded as a function, defined in the HANA-specific language "SQLScript", which is built upon SQL commands or special HANA pre-defined functions;
- Must be read-only.

3.10 Analytic Privileges

Analytic privileges control access to SAP HANA data models. It is possible to implement row-level security with analytic privileges, however we can restrict access in a given data container to selected attribute values. Other interesting concept is the dynamic analytic privileges that provide a more flexible approach. The actual filter conditions are obtained at runtime from a stored procedure, which can contain complex logic. This enables the re-usage of the same analytic privilege for several users and the change of the filter condition.

4 SQL SCRIPT BASICS

In this section an overview of SQL Script was shown and it was also explained how should it be used in SAP HANA.

4.1 SQL Script

SQLScript is an interface for applications to access SAP HANA, an extension of ANSI Standard SQL and a language for creating stored procedures in HANA. The main goal of SQLScript is to allow the execution of data-intensive calculations inside SAP HANA database eliminating the need to transfer large amounts of data from the database to the application and making possible the execution of

calculations in the database layer to get the maximum benefit from SAP HANA features such as fast column operations, query optimization, and parallel execution. Compared to plain SQL queries, SQLScript has the following advantages:

- Functions can return multiple results, while an SQL query returns only one result set;
- Complex functions can be broken down into smaller functions enabling modular programming, reuse, and better understandability. For structuring complex queries, standard SQL only provides SQL views that do not allow parameters;
- SQLScript supports local variables for intermediate results with implicitly defined types. With standard SQL, you need to define globally visible views even for intermediate steps;
- SQLScript has control logic such as if/else that is not available in SQL.

One of the most important characteristics of SAP HANA with SQL Script is the parallel processing, which made possible the execution of SELECT statements unless:

- Local scalar parameters and variables are used in the procedure;
- Read/Write procedures or DML/DDI operations are executed;
- Imperative logic is used within the procedure;
- SQL statements are used that are not assigned to a variable.

To call a procedure in SQL Script CALL [schema.]name (:in_param1, out_param [...]) syntax must be used.

4.2 Calculation Engine

The calculation engine is the execution engine for SQLScript. SQLScript statements are parsed into the calculation model as much as possible. The calculation engine instantiates a calculation model at the time of query execution. To be able to use the calculation engine the following syntax must be applied, depending on the object to be created: CE_COLUMN_TABLE ("table_name", ["attrib_name",...]); CE_JOIN (:var1_table,

```
:var2_table, [join_attr,...], [attrib_name,...]),
CE_PROJECTION (:var_table, [param_name
[AS new_param_name],...],[Filter]).
```

4.3 Imperative Logic

This style of programming can be used to create procedures for basically any transformation of data into output data. The imperative logic can be identified by the usage of the following keywords: IF, ELSE, ENDIF, WHILE, FOR and CASE.

5 EXPOSING AND CONSUMING DATA WITH ODATA

In this section and explanation of SAPUI5 and OData was provided to understand the new features that SAP HANA brings and how can they be used.

5.1 SAPUI5

SAPUI5 is an enterprise-ready HTML5 rendering library with a completely client-side UI. This is an open and flexible library with 3rd party JavaScript integration. Its extensibility and theming make it a great development library for smartphones, tablets and desktops, however we must take into consideration that HTML, CSS3 and JavaScript skills are necessary to develop an SAPUI5 application. Some key features and benefits of this library are:

- User side: any screen on any device, cutting-edge controls, powerful theming and branding, efficiency and performance;
- Innovation: unmatched extensibility, great productivity, timeless SAP data consumption, fast release cycles;
- Standard based: Eclipse-based design-time, enterprise readiness, well-known and easy to learn, openness and flexibility.

5.2 OData Services

OData service expose an end point that allows access to data in the SAP HANA database. To allow this exposure an OData data model is applied to organize/describe data with Entity Data Model(EDM) and an OData protocol (REST-based create, read, update, and delete

+ OData-defined query language) and some client libraries (pre-built libraries to request OData and display results) are used to help on this procedure. The process to develop an OData service requires some key points:

- Data End Point: define which data to expose (for example, table);
- OData Service Definition: define mechanism to expose authorized data;
- UI Call to OData Service: link UI table view to the OData service
- OData Display/Test: URL filters (\$top, \$format, \$orderby, \$filter, \$select).

6 EXPOSING AND CONSUMING DATA WITH SERVER-SIDE JAVASCRIPT

This section provided the server-side logic to expose and consume data, complementing this way the client-side explained in the previous section. This chapter focused in JavaScript, how is the development model and libraries that can be used and created.

6.1 Server-Side JavaScript (XSJS)

Server-side JavaScript is used to control data flow between client and SAP. JavaScript as the programming language was chosen because it is open source, easy to reuse, fast development, most people know the basics for programming in it and provides a lightweight procedural logic. Its API is responsible for the interaction with the SAP HANA XS runtime environment, the direct access to SAP HANA database capabilities and the exposure, update, insertion and deletion of data.

6.2 Extending the XSJS Service

The extension of XSJS Service allows a more efficient and organized way to simplify the development process. The reuse of external libraries, creation of custom libraries and the share of libraries between projects and/or XSJS services allows the source code re-utilization, custom output formats eliminates the need to manually change data format and the data extraction with SQL statements simplifies the reading data process.

6.3 Calling XSJS from the UI

AJAX techniques are used to call a JavaScript method on the UI. Since JavaScript is used as programming language JSON must be used for data-interchange and the method for the actual call is GET. However we should not forget that the URL for the XSJS service must be provided before the call, if not the method will not know where is located the function to call.

7 SAP HANA ADVANCED DEVELOPMENT OPTIONS

This chapter introduced two important tools embedded in SAP HANA that simplify the development process and cycle.

7.1 Lifecycle Management

Lifecycle management has the responsibility to manage software release, product development and application packages, enforce version control and monitor software packages. The objects created in SAP HANA are transported through delivery units, which are a collection of packages containing the development objects that are exported/imported using .tgz archive. To define a delivery unit the following information must be provided: delivery unit name, vendor ID (namespace), responsible owner and a version number.

7.2 SAP HANA UI Integration Services

UI Integration Services allow XS application developers to concentrate on the core functionality of their application, by providing UI services that enable the creation of multiple experiences and provides some important features on the application developer side to simplify the development process:

- Page building: state-of-the-art design environment for application layout and content;
- Property persistence and customization: widget can define properties that are persisted, enabling customization;
- Property personalization: widget can define properties that are persisted on user level;

- Publish-Subscribe: event mechanism that allows data interchange between widgets in the same application site;
- Context: all properties are reflected in the URL, allowing bookmarking and sharing.

8 CONCLUSION

SAP HANA is an innovative in-memory relational database management system that makes full use of the capabilities of current hardware to increase application performance, reduce cost of ownership, and enable new scenarios and applications that were not possible before. With SAP HANA, it is now possible to build applications that integrate the business control logic and the database layer with unprecedented performance. As a developer, SAP HANA brought a change in programming thinking since now we should transform and move the data in memory next to the CPUs to optimize the application performance, opposing to the previous best practices.

ACKNOWLEDGMENTS

The author would like to thank SAP for the opportunity to attend this course and also the support of all family, friends and colleagues in this final effort to finish the master degree.

REFERENCES

- not cited anywhere in the text!*
- [1] T. Jung and G. Nolan, *Introduction to Software Development on SAP HANA*, <https://open.sap.com/course/hana1-1> [Online; accessed 2014-07-01], 2013.
 - [2] SAP, *SAP HANA Developer Guide*, Version 1.2, 2013.

In this type of document (technical), the conclusion should start with a summary of the subject addressed and then should highlight the results.

APPENDIX

STATEMENTS OF EXECUTION



Confirmation of Participation

Mafalda Sousa Rosado

Date of birth: September 27, 1986
e-mail: mafalda.rosado@sap.com

has participated in the following openSAP course:

Introduction to Software Development on SAP HANA

Instructor: Thomas Jung

This six-week online course covered the following topics:

- Developing Applications for SAP HANA
- Database Tasks, Loading, and Modeling
- SQLScript Basics
- Exposing and Consuming Data with OData
- Exposing and Consuming Data with Server-Side JavaScript
- SAP HANA Advanced Development Options

openSAP is SAP's platform for open online courses. It supports you in acquiring knowledge on key topics for success in the SAP ecosystem.

Walldorf, December 2013

openSAP



Record of Achievement

Mafalda Sousa Rosado

Date of birth: September 27, 1986
e-mail: mafalda.rosado@sap.com

has successfully completed the following openSAP course:

Introduction to Software Development on SAP HANA

Instructor: Thomas Jung


This six-week online course covered the following topics:

- Developing Applications for SAP HANA
- Database Tasks, Loading, and Modeling
- SQLScript Basics
- Exposing and Consuming Data with OData
- Exposing and Consuming Data with Server-Side JavaScript
- SAP HANA Advanced Development Options

Maximum score possible for this course: 300 points.

The candidate scored 294.3 points (98.1 %) by working on weekly assignments and taking a final exam.

The candidate belongs to the top 20 % of the course participants.


Dr. Bernd Welz
Executive Vice President
SAP Solution and Knowledge Packaging


Thomas Jung
Instructor

Walldorf, December 2013

openSAP is SAP's platform for open online courses. It supports you in acquiring knowledge on key topics for success in the SAP ecosystem.

openSAP