This sample chapter describes the use of SAP HANA as an application platform. It covers developing applications using SAP HANA extended application services (SAP XS Classic), SAP XSA, and the SAP Web IDE.
In this chapter, we’ll explain how you can use SAP HANA as an application platform, and we’ll learn about the exciting new features available in the SAP HANA extended application services, advanced model (SAP HANA XSA) as part of SAP HANA SPS 11 and SPS 12.

3 SAP HANA as an Application Platform

We refer to SAP HANA as a platform, but sometimes we forget that the platform is capable of being more than a high-speed database for an SAP ERP or SAP Business Warehouse (SAP BW) implementation. SAP HANA is fundamentally a database platform that can support any type of application, not just SAP-delivered ones, and SAP has been hard at work developing an application platform that is flexible and powerful enough to meet the changing needs of a constantly evolving business landscape.

SAP took the first steps towards unleashing the power of SAP HANA as an application platform when the SAP HANA extended application services were included in the release of SAP HANA SPS 5. At the time of its release, this technology was referred to as the XS engine, or simply as SAP HANA XS; however, we’ll refer to this version of the application services as SAP HANA extended application services, classic model (SAP HANA XS Classic), because SAP significantly upgraded and expanded the application platform with SAP HANA SPS 11. The latest version of the application platform is known as SAP HANA extended application services, advanced model, or simply as SAP HANA XSA.

Before the release of SAP HANA XS Classic, in order to connect a webpage or application to the data in your SAP HANA database, you needed to connect through another application server, like the SAP ABAP or SAP Java application stack, or you could also connect through ODBC (Open Database Connectivity) or JDBC (Java Database Connectivity). SAP HANA XS Classic simplified this process by adding a lightweight application server as part of the SAP HANA system itself. With SAP HANA XS Classic, you could develop applications that rendered in a
browser or on a mobile device and connected through the XS engine in SAP HANA directly to the database layer.

With SAP HANA XSA, SAP added support for Node.js, Java, HTML, and other application programming interfaces, like C++. In an attempt to simplify and unify development between the SAP HANA on-premise and SAP HANA cloud deployments, SAP HANA XSA is based on the Cloud Foundry, which means you can now support multiple languages and multiple runtime instances within the same server. As a result, each deployment of an application or service has its own copy of the Java or Node.js runtime and runs in its own separate instance within the SAP HANA XSA architecture.

In this chapter, you'll learn about SAP HANA XS Classic and SAP HANA XSA. If your SAP HANA platform is on SPS 10 or an earlier release, you'll need to develop your applications with SAP HANA XS Classic. On the other hand, if your platform is on SPS 11 or higher, you can take advantage of SAP HANA XSA, so we'll look at how to use both platforms. We'll also learn how to develop applications using SAP HANA Studio and the SAP Web IDE.

**Note**
If you start with SAP HANA XS Classic because your system is on SPS 10 or an earlier release and later upgrade your system to SPS 11 or higher, you'll still be able to use the content you developed for SAP HANA XS Classic.

### 3.1 SAP HANA XS Classic

As part of making SAP HANA more self-reliant, SAP developed SAP HANA XS Classic to enable the development of native SAP HANA applications. If your SAP HANA system is on SPS 5 through SPS 10, you'll need to develop your native SAP HANA applications with SAP HANA XS Classic.

**Note**
The SAP HANA XS Classic was not available until SAP HANA SPS 5, so you'll need at least SPS 5 to be able to enable the development of native SAP HANA applications with SAP HANA XS Classic. If you're already on SPS 11 or higher, you should be using SAP HANA XSA, so see Section 3.2 and Section 3.3 for information on SAP HANA XSA and how to develop SAP HANA XSA content.
SAP HANA XS Classic also allowed developers to create applications that were easy to deploy via a web browser or mobile device. By combining HTML5 with XSJS (XS JavaScript), you could develop applications that were wholly contained within SAP HANA. The expense and overhead involved in deploying separate application servers could be avoided, and the application logic often could be pushed to the database layer for processing in SAP HANA’s in-memory computing engine (IMCE). Furthermore, SAP HANA XS Classic allows for authentication and security through SAP HANA, so there’s no need to configure complicated authentication protocols between the application server and the database.

### 3.1.2 Developing SAP HANA XS Classic Applications

The release of SAP HANA XS Classic prompted the development of a new perspective in SAP HANA Studio called the SAP HANA DEVELOPMENT perspective, which at the time was one of two preferred methods for developing browser- and mobile-based applications and content. (The other is the SAP HANA Web-Based Development Workbench, which we’ll discuss in Chapter 8.) The SAP HANA DEVELOPMENT perspective is intended for development of server-side scripting and object development (unlike SAPUI5, which is executed mostly at the client side during runtime). However, it is important to note that SAPUI5 is technically a component of SAP HANA XS.

With SAP HANA XS Classic, you get access to some serious server-side programming as well, including XMLA, OData, and more complex Java scripting with XSJS. You can also access the SAP HANA web server features directly. The functionality in SAP HANA XS Classic provides SAP with an open platform for companies that want to use SAP HANA as a development platform for targeted applications that go far beyond traditional SAP ERP and data warehousing.

A major benefit of SAP HANA XS Classic is that it does not require another application server. The application simply becomes native to SAP HANA and doesn’t need another piece of hardware or application server software; in other words, you build and run your application on the SAP HANA platform. The database is the SAP HANA database, with its column and row stores, and at application runtime, your application runs on SAP HANA XS Classic. In this type of deployment, SAP HANA XS Classic serves a dual role as an internal SAP HANA small-footprint application server, including a web server and a core application development platform inside SAP HANA. When you are designing an SAP HANA XS Classic application, you can use the SAP HANA Web-Based Development Workbench, as shown in Figure 3.3, or you can also use the SAP Web IDE (Section 3.3).
Unlike development in Eclipse-based SAPUI5 and SAP HANA Studio, you can use either the SAP HANA Web-Based Development Workbench or the SAP Web IDE to create a complete application without installing anything on your desktop. Instead, you’ll use your web browser to do all the development for your SAP HANA XS Classic application.

When you develop applications, you can also develop the backend in SAP HANA Studio. To do so, unlike the web interface option, you need to have SAP HANA Studio installed on your local PC, and you need to have a connection, user name, and password set up on an SAP HANA database system. You also need to be assigned an SAP HANA developer role by your system administrator to be permitted to do this kind of native SAP HANA XS Classic development.

Generally, SAP HANA XS Classic mini-application servers inside SAP HANA can assist frontend applications developed in SAPUI5 and SAP HANA XS Classic by merging data from many tables into simple access views.

You can also push application logic, such as a calculation, into the analytical and calculation views inside SAP HANA. This functionality simplifies code execution and lowers processing power requirements on smaller mobile devices because most of the logic is executed in memory at the database layer, instead of on the smaller devices, or even on traditional application servers—which, even if they are somewhat faster, are still extremely slow compared to SAP HANA. You can push much of the mundane application logic to the SAP HANA database server level instead of having to install dedicated external application servers. As a result, you can keep your SAP HANA environments simple and clean, which your SAP Basis support staff will appreciate.

To get started, go to SAP HANA Studio and click the Window menu option. Then select Perspective / Open Perspective / Other and select SAP HANA Development, as shown in Figure 3.4.

From the SAP HANA DEVELOPMENT perspective, you can create a project for your development effort. From the File menu, select New • XS Project (Figure 3.5) to create a new SAP HANA XS Classic development project. Creating a new project allows you to separate your work from other developers who may be working on other areas of the SAP HANA system.
other developers to collaborate with you on the project, or you can leave this blank. Next, choose the Use default location option to save the files to your working directory for your SAP HANA Studio installation. You can also browse to the location where you’d like to save your project files.

If you browse to a location, you’ll also need to choose the file system for the files from the following options:

- **Default**
  
  This is the default file system on your development machine.

- **Semantic File System**
  
  Allows you to specify a location within the Project Explorer landscape.

If you’re saving the project locally, just click Finish to create the project in the Project Explorer.

On the other hand, if you share the project to the repository, you’ll need to click Next to select the Repository Workspace for the project (Figure 3.6). You can either select from an existing workspace, or you can click Add Workspace to create a new one in the repository.

Click Next to advance to the next window for the SAP HANA XS project wizard (Figure 3.7), where you’ll configure the common object names for the project repository. From here, you’ll need to provide a Schema name for the new project, a DDL name, and an XS JavaScript name for the objects you’ll be adding to the repository. Click Finish to create the new SAP HANA XS Classic project.

Once the project has been created in SAP HANA Studio, you’re ready to start building your application. Please be aware that developing applications in SAP HANA XS Classic and SAP HANA XSA requires a solid background in coding, so you may need to develop or brush up on those skills as you’re learning to create applications on SAP HANA. The coding for SAP HANA XS Classic is quite similar to JavaScript, and SAP HANA Studio provides some development wizards to get you started with common tasks. The wizards also allow you to build your own JavaScript library files where you can store your most frequently used controls or custom code extensions. The purpose of these wizards is to make your job as easy as possible.

When your project has been created, the Project Explorer will show the project content in the navigation pane, and you can code your application in the editor, as shown in our simple HelloWorld_XS example in Figure 3.8.
Note

The steps below assume you’ve used the package and object names that we’ve presented in the examples above. If you’ve selected different names, make sure you update the object names in the code listings below to match your selections.

To duplicate the SAP HANA XS Classic version of the HelloWorld program, you’ll need to do the following:

1. Create the SAP HANA XS Classic project as explained earlier.
2. Open the .xsaccess file and enter the following line of code to the file:

   "default_file": "HelloWorld.xsjs",

3. Save the .xsaccess file.
4. Add a file called .xsprivileges to the project.
5. Add the following code to the file:

   "privileges":
   [ { "name": "Basic", "description": "Basic usage privilege" } ]

6. Save the .xsprivileges file.
7. Add a file named model_access.hdbrole to the project and add the following code to the file:

   role HelloWorld_XS::model_access {
     application privilege: HelloWorld_XS::Basic;
   }

8. Open the HelloWorld.xsjs file and add the code from Listing 3.1 to the file.

   $.response.contentType = "text/html";
   var output = "Hello, " + $.session.getUsername() + " <br><br>";
   var conn = $.db.getConnection();
   var pstmt = conn.prepareStatement( "SELECT CURRENT_USER FROM DUMMY" );
   var rs = pstmt.executeQuery();
   if (!rs.next()) {
     $.response.setBody( "Failed to retrieve data" );
     $.response.status = $.net.http.INTERNAL_SERVER_ERROR;
   } else {
     output = output + "This is the response from my SQL.
     The current user is: " + rs.getString(1);
   } 
   rs.close();
   pstmt.close();
   conn.close();
   $.response.setBody(output);

   Listing 3.1 Code for the SAP HANA XS Classic HelloWorld Program

10. Activate all the objects from your project with the Activate All SAP HANA Development Objects button (Figure 3.9) or by pressing (Ctrl)+(Shift)+(F3) on your keyboard.

Note

When adding files, you may be able to select the XS file type that matches the object you’re adding from the context menu or by navigating to File • OTHER; however, if the file type you’re looking for is not available, simply right-click in PROJECT EXPLORER to open the context menu and select NEW FILE.
11. Add the Hello World XS::model_access role to your user.

When you’ve activated all the objects in the project, you should be able to view your simple Hello World greeting by navigating to http://<hana_server>:80 <instance>/helloworld_xs/. Login to the server with the user with the Hello World XS::model_access role. You should be greeted by your user name as shown in Figure 3.10.

![Figure 3.10 Hello World Program Developed in SAP HANA Studio with SAP HANA XS Classic](image)

This example of creating an application in SAP HANA XS Classic should be enough to get you started. For more information about coding applications in SAP HANA XS Classic and SAP HANA XSA, you can download the following reference guides from SAP:

- SAP HANA XS JavaScript Reference
- SAP HANA JavaScript API Reference
- SAP HANA XSUnit JavaScript API Reference
- SAP HANA XS DB Utilities JavaScript API Reference
- SAP HANA REST API Reference

These guides, and many more, are available for download from [http://help.sap.com/hana_platform](http://help.sap.com/hana_platform).

### SAP HANA XS and SAPUI5

Although SAP HANA XS and SAPUI5 are separate components, it’s important to understand how they work together. You could build an SAPUI5 application without SAP HANA XS being directly involved, but if you build an SAP HANA XS-based application, you may be using SAPUI5 as your custom-developed frontend tool, or you can use the SAP Web IDE or the SAP HANA Web-Based Development Workbench.

You can clearly see this relationship if you look carefully under the content folder in your SAP HANA XS project. The subsections correspond to the different components as illustrated in Figure 3.11.

![Figure 3.11 Development Architecture for SAP HANA XS and SAPUI5 on SAP HANA](image)

It is important to note that this type of development is not for those with limited programming skills; some basic training with programming languages such as C++, Java, JavaScript, and VBScript is essential to learn how to do this properly.

### 3.2 SAP HANA XSA

With SAP HANA SPS 11, SAP released SAP extended application services, advanced model (SAP HANA XSA). SAP HANA XSA builds on the foundation provided by SAP HANA XS Classic, with support for JavaScript, Node.js, and Java, as well as support for GitHub and Maven. As a result, you can create applications with separate components using different programming languages, all on the same SAP HANA server.

SAP HANA XSA uses the micro-services approach where applications and their related, programming language-specific runtime environments are modular, decoupled services that exchange information through RESTful APIs (representational state transfer for application programming interfaces). This micro-service...
architecture means you can maintain one service without affecting other services. For example, you can apply a patch to one component of the application without causing the rest of the application to crash. As a result, you may potentially be able to avoid downtime, as new versions of an existing micro-service may be deployed and tested on the same SAP HANA instance without requiring a full system outage.

**Note**

REST is an architecture style used in designing networked applications. REST relies on client-server, cacheable, stateless communications protocols, most often HTTP. Applications based on the REST architecture typically allow the user to advance through the program flow by selecting links that load and render the next page of the application.

SAP HANA XSA also introduced a new runtime controller based on the Cloud Foundry, which enabled the introduction of build packs into SAP HANA development. Build packs include a set of compilers and utilities used in application design, and the new SAP HANA XSA runtime supports build packs with the following languages:

- XSJS (XS JavaScript)
- Node.js
- Java on Apache TomEE 1.7.3
- Java on Apache Tomcat 8.0.32

The runtimes in SAP HANA XSA are decoupled from the database in SAP HANA, which allows for flexible scaling during peak demand. New servers or nodes can be dynamically added or removed as needed to meet your business needs.

**Note**

You may notice that we’ve included C++ as one of the available language runtimes with SAP HANA XSA in our landscape diagram (Figure 3.12). While C++ and FastCGI runtimes are available internally, as of August 2016, support for C++ and FastCGI is not in the general release of SAP HANA XSA for SAP HANA SPS 12. In fact, since the SAP HANA XSA architecture is based on the Cloud Foundry, in the future, you may potentially be able to use other Cloud Foundry build packs, such as Go, Ruby, or PHP, as your chosen language or runtime for SAP HANA XSA.

In short, the release of SAP HANA XSA has completed the transformation of SAP HANA from a specialized application server and high-speed database engine into a true next-generation platform for meeting your business needs. It’s also important to note that, as of SPS 11, SAP recommends using SAP HANA XSA to develop any new applications for the SAP HANA platform. In this section, we’ll take a look at how SAP HANA XSA fits into the rest of the SAP HANA architecture. We’ll also provide step-by-step instructions for installing SAP HANA XSA and the SAP Web IDE so you can quickly start building your own applications.

### 3.2.1 SAP HANA XSA Architecture

Let’s take a look at the new SAP HANA XSA landscape (Figure 3.12) and compare it to the SAP HANA XS Classic landscape (Figure 3.2). The new landscape retains support for all of your SAP HANA XS Classic applications, so upgrading your SAP HANA installation to SPS 11 or greater won’t cause all of your SAP HANA XS Classic applications to fail, but only new applications can take full advantage of the improvements in SAP HANA XSA.

The application router accepts service requests from browser-based applications and mobile applications and routes these requests to the correct micro-service. A separate micro-service container is created for each application component, and these components exchange data through the RESTful APIs. Multiple containers may be created for any given runtime, allowing you to develop multiple application components using different languages.

![Figure 3.12 SAP HANA Application Landscape with SAP HANA XSA](image-url)
Each micro-service container has a complete copy of the entire Node.js, Java, or XSJS runtime, which locks each component into using the version with which it was deployed. As a result, the upgrade path for future revisions is much more stable, since a deployed component will continue to use its original runtime library until it’s individually deployed with an updated version in a future revision.

The micro-service approach is also applied at the level of the operating system, since each deployed instance of a given runtime will have its own dedicated operating system process. This decoupled runtime architecture means that the failure of one service will not cascade outward to other application components or services.

Next, we’ll take a closer look at some of the key components of the SAP HANA XSA architecture, including the role of multi-target applications and how they’re handled by the runtime platform. We’ll also explain how user accounts and authentication are handled in the landscape, as well as the role of OData services, and then we’ll finish this section with a look at the administration tools available for use in SAP HANA XSA.

**Multi-Target Applications**

SAP HANA SPS 12 introduced support for multi-target applications (MTAs). MTAs use what is commonly known as a *blue-green deployment technique*, which designates two identical target environments, the blue and green environments. In this deployment scenario, only one of the two environments will be live at any given time. Development and testing can therefore take place in the other environment, and switching between the environments can be as easy as updating the routing within your network to identify the development environment as the live version.

By making switching between environments easy, you can quickly and easily switch back to the known good environment in the event of an unexpected failure in the other. Once the new environment has been certified as stable through your company’s risk management processes, you can save the old environment to an archive and copy the new environment for use as the next development environment.

An MTA will consist of one or more application modules, each of which is considered a micro-application. These micro-applications may be bound to required services and pushed to a deployment platform. Each module may expose attributes for use in other modules and may also be dependent on other modules.

An MTA deployment descriptor file specifies the modules and dependencies for an application, including the technical types, dependencies, and any parameters required for the modules. The descriptor file is used to verify that the required dependencies exist and that the modules are deployed in the correct order and also to set up any necessary connections between modules. All the application files for the MTA, including manifests, descriptors, and service configuration information, are gathered into an archive package, and this package may be used to deploy the MTA. An MTA archive typically contains a folder for each module in the application. The exact folder structure (as specified in the MTA manifest) will depend on the structure of your application but may contain folders such as:

- **java/**
  Java application files
- **web/**
  State web content and application routing configuration
- **db/**
  Views, tables, procedures, calculation views, etc. common to an SAP HANA database
- **js/**
  Node.js application and XSJS files

When you develop your applications using the micro-service architecture and MTAs, you should adhere to the following guidelines:

- **Isolated and independent**
  Services must be isolated and not share any libraries and components with each other. Common libraries and frameworks are embedded in each service container, which facilitates the goal of isolating each service from another. Redundancies will exist, and different services may be running on different versions of the same libraries.

- **Languages and runtimes**
  Services may be implemented in any language and run in an available runtime environment, regardless of the languages and runtime environments implemented in other services.
Lifespan
Each service has a lifecycle that is independent of all other services, allowing for stable integration with other services and facilitating isolated upgrading or replacement of the service.

Extensibility
Services may wrap, replace, or extend existing services and may be aggregated with new and existing services to provide a “mashup” service.

For more information about developing with MTAs, see the SAP HANA XS Advanced Multi-Target Applications subsection of the Introduction to Application Development and Deployment (XS Advanced Model) site on http://help.sap.com.

Runtime platform
The runtime platform is the framework that makes the entire micro-services architecture of SAP HANA XSA work properly. Each micro-service container for each of your multi-target applications relies upon the runtime platform to manage a vast array of functions within the landscape, including:

- Routing
- Load balancing
- Identity and access management
- Lifecycle management
- Elastic, on-demand scaling
- API management for each container

The XS platform starts and stops each micro-service container as required and enables monitoring of each container instance and application, as well as CPU, memory, network, and file system resources used by each container.

Without the coordination and management activities carried out by the SAP HANA XSA runtime platform, the entire landscape would require multiple application servers, each dedicated to one application, and would lack the benefits of the micro-services approach, especially those pertaining to lifecycle management. Without the runtime platform, updating one component of an application would require redeploying the entire application. Instead, with SAP HANA XSA, you can update and deploy changes that are isolated to their respective containers, and you don’t need separate application servers for each application.

User Account and Authentication Service
The User Account and Authentication (UAA) service handles any unauthenticated request to an SAP HANA XSA application or the XS Application Router. UAA relies on an external user store, such as SAP Cloud Identity, Lightweight Directory Access Protocol (LDAP), or the SAP HANA database, and can process different authentication methods, such as Security Assertion Markup Language (SAML) assertions, X.509 certificates, and basic user names and passwords.

When the UAA successfully processes a login, an OAuth token is issued. This token can be used in all other calls to application services for the user and can also be forwarded to another service for user-based authorization. For example, the UAA could provide an OAuth token to the SAP HANA database for authorizations based on SAP HANA database users. The OAuth token provides a unified authentication approach while also specifying the scope of the user’s permissions, thus allowing functional authorization checks, based on the scope, for every service layer in an application.

OData Services
OData is the SAP-recommended protocol for RESTful data access. To enable support for OData services, SAP included OData server libraries for use with SAP HANA XSA, including the libraries for Java and Node.js. Your SAP HANA XSA applications may define their own data providers using the OData libraries or may define xsodata metadata artifacts or core data services (CDS) views as generic data providers.

Note
CDS views are not covered in detail in this book, but they provide developers with the ability to define entities and the relationships between those entities using an SQL-based data definition language (DDL). This SQL-based DDL has been extended to allow for the definition of relationships between CDS views, annotations regarding the specific use of CDS objects, and aggregation expressions, to name a few. For more information about CDS views, consider reading “Enhanced ABAP Development with Core Data Services (CDS) – How CDS Brings a Modern Data Modeling Approach to ABAP” by Karl Kessler, published on October 8, 2015 in Volume 16, Issue 4 of SAPinsider.
Administration Tools
To monitor your SAP HANA XSA landscape, as well as any applications you develop, SAP has provided administration tools for SAP HANA XSA. These tools are available for easy integration with the SAP HANA Cockpit (see Chapter 10 for detailed instructions about adding tools to the cockpit). The following administration tools are available for integration with the SAP HANA Cockpit:

- Application Monitor
  Monitors SAP HANA XSA applications.
- Organization and Space Management
  Create, list, and delete user spaces and organizations in SAP HANA XSA.
- Application Role Builder
  Manage and maintain user roles and collections in SAP HANA.
- SAML Identity Providers Configuration
  Allows configuration of SAML identity providers for SAP HANA XSA model applications that use SAML.
- User Management
  Manage and create business users for SAP HANA XSA.
- SAP HANA Logical Database Configuration
  Manage database instances for SAP HANA XSA applications.
- SAP HANA Service Brokers
  Monitor and manage SAP HANA XSA service brokers.
- Job Scheduler Service Dashboard
  Schedule, create, and manage long-running SAP HANA XSA jobs.

### 3.2.2 Installing SAP HANA XSA Runtime
Before you begin installing the SAP HANA XSA runtime server, you need to determine if your SAP HANA server is a single-host or multihost system. The single-host installation is the easiest to install and configure, and this installation can be scaled up as needed. The host system will need both the database worker and xs_worker host roles in a single-host installation. See Figure 3.13 for an example of the system service landscape in a single-host system.
A multihost installation of SAP HANA XSA will allow for load balancing between different hosts. For this installation, the server software is installed in a shared file system, and the file system must be mounted by all the hosts in the distributed system. In this configuration, the system hosts may be active, or they may be idle and only activated when the load on the system increases. See Figure 3.14 for an example of the SAP HANA XSA system landscape in a simple multihost system.

You can also perform an advanced installation of your SAP HANA XSA runtime server, in which your worker, xs_worker, system, and xs_standby roles are distributed across multiple systems. In this type of installation, the role for each system must be determined and assigned during the installation of SAP HANA XSA. For an example of an advanced multihost deployment, including a worker system, xs_worker system, and a standby system, see Figure 3.15.
Installing the SAP HANA XSA runtime is a fairly simple process, regardless of the version of SAP HANA lifecycle manager (SAP HANA LCM) you choose to use, so we’ll describe the steps to install the runtime using both the graphical and command line versions of the SAP HANA LCM in the following sections. Then, we’ll finish this section with a detailed look at installing the SAP Web IDE.

**Before You Begin...**

As you should be using the SAP Web IDE to develop any SAP HANA XSA applications, you should download the SAP HANA XSA components you want to install prior to beginning the installation. The components can be downloaded from the SAP Marketplace at http://launchpad.support.sap.com/.

Click the Software Downloads tile, then select By Alphabetical Index (A-Z) • H • SAP IN-MEMORY (SAP HANA) • HANA PLATFORM EDITION • SAP HANA PLATFORM EDITION • SAP HANA PLATFORM EDIT. 1.0. Click the SUPPORT PACKAGES AND PATCHES button in the upper right, then select ENTRY BY COMPONENT to find the components listed below:

- XS ADVANCED RUNTIME
- HANA SHINE CONTENT FOR XSA (optional)
- XS ADVANCED SERVICES
- XS ADVANCED MONITORING
- SAP WEB IDE 4 SAP HANA
  - DI CORE 1
  - SAP WEB IDE 1
- SAP HANA RUNTIME TOOLS

When you’ve downloaded all the components, you need to create a folder on the Linux host of the SAP HANA system to contain the installation media. For convenience, you may also consider creating a separate directory within this directory to hold the extracted SAP HANA XSA runtime components.

You can then extract all the installers to this installation directory and simply point the lifecycle manager to this location with the option to Include Subdirectories when adding components. Then you simply need to choose the components to install in the order recommended later in this section, and the SAP HANA XSA runtime will be detected properly as a component to install each time. See Figure 3.16 for an example of the installation directory we used to install our components.

**Warning!**

When you install the SAP HANA XSA runtime, you should specify the fully qualified domain name for your SAP HANA system. If you fail to do so, and instead you select the local domain, you may have authentication problems when you use the SAP HANA XS command line interface (XS CLI) to obtain the URLs for accessing the SAP Web IDE, as described in Section 3.3.

**Installation with the SAP HANA Lifecycle Manager Web Version**

To install the SAP HANA XSA runtime, you’ll need to use the SAP HANA LCM. You can use the graphical user interface (GUI), web-based interface, or the command line version of the lifecycle manager; however, the GUI lifecycle manager or web-based lifecycle manager is generally easier to use.

**Note**

You can find a complete description of the installation process in the SAP HANA Server Installation and Update Guide, available for download in PDF format from http://help.sap.com/. Also, the installation steps are nearly identical between the web-based
and GUI versions of the lifecycle manager, so we’ll be giving examples with the web-based version in this section.

You’ll need to use the following process to install the SAP HANA XSA runtime:

1. Login to the SAP HANA system as the root user.
2. Navigate to the directory containing the installation medium.
3. Enter `/<install_path>/<sid>/hdblcmweb` from the command line to open the web-based version of the SAP HANA lifecycle manager (LCM).
4. Select **Install New System** and then click **Next**.
5. Click **Add Software Locations** (Figure 3.17).
6. Navigate to the location of the installation media and click **OK**.
7. Confirm the Detected Software Components by clicking **Next**.
8. Check the Install SAP HANA XS Advanced Runtime checkbox and click **Next**.
9. Select a system from the Existing Hosts in the SAP HANA System list and click **Next**.
10. (Optional) Select Add Host to specify parameters for additional hosts in a multi-host installation. You’ll also need to specify:

<table>
<thead>
<tr>
<th>Role</th>
<th>Role name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Worker</td>
<td>worker</td>
<td>Worker host used for database processing.</td>
</tr>
<tr>
<td>Database Standby</td>
<td>standby</td>
<td>Idle and available for failover in high-availability.</td>
</tr>
<tr>
<td>Dynamic Tiering Worker</td>
<td>extended_storage_worker</td>
<td>Host for dynamic tiering.</td>
</tr>
<tr>
<td>Dynamic Tiering Standby</td>
<td>extended_storage_standby</td>
<td>Standby host for dynamic tiering.</td>
</tr>
<tr>
<td>Accelerator for SAP ASE Worker</td>
<td>ets_worker</td>
<td>Worker host for SAP ASE accelerator.</td>
</tr>
<tr>
<td>Remote Data Sync</td>
<td>rdsync</td>
<td>Remote data sync host.</td>
</tr>
<tr>
<td>Smart Data Streaming</td>
<td>streaming</td>
<td>Host for smart data streaming.</td>
</tr>
<tr>
<td>XS advanced runtime worker</td>
<td>xs_worker</td>
<td>Host for SAP HANA XSA.</td>
</tr>
<tr>
<td>XS advanced runtime standby</td>
<td>xs_standby</td>
<td>Available for failover/dynamic scaling.</td>
</tr>
</tbody>
</table>

11. For the System Type, select either a **Single-Host System** or **Multiple-Host System**. (This prompt won’t appear in single-host system landscapes.)
12. Select Yes or No in the **Automatic Assignment of XS Runtime roles** window. This option will determine whether or not you want to assign the SAP HANA XS runtime role to the host of the database you selected. If not, you’ll need to specify a different host for the runtime role.
13. Enter the **System Administrator (<sid>adm) Password**, **Database User Name**, and **Database User Password** and click Next.
14. The **Configure XS Advanced Runtime** screen (Figure 3.18) will require you to provide the:
   - **XS Organization Manager user** *(XSA_ADMIN by default)*
   - **XS Organization Manager User Password**
   - **Customer Organization Name**
   - **Customer Space Name**
   - **Routing Mode**
     - **Ports**: SAP HANA XSA system services will be assigned to the host domain with different ports for each service (e.g., `www.<domain>.<extension>:3<instance>` is usually the link for the uaa-security service).
     - **Hostnames**: SAP HANA XSA system services will be assigned to the host domain with different domain prefixes (e.g., `http://uaa-security.<domain>.<extension>` and `http://webide.<domain>.<extension>`).
   - **Domain Name** *(a fully qualified domain name is recommended)*

![Figure 3.18 Configuring SAP HANA XSA Runtime Settings](image)

15. Click **Next** to confirm the settings.
16. Designate user IDs for the **XS Advanced SAP Space OS User ID** and the **XS Advanced Customer Space OS User ID**.
17. Click **Next** to confirm the user IDs.
18. (Recommended) Select the **XS Monitoring 1** and **XS Services 1** updates as part of the installation, if available, and then click **Next**.
19. Confirm the system settings you entered on the **Review and Confirm Entered Data** screen (Figure 3.19).

![Figure 3.19 Confirming System Settings for the SAP HANA XSA Runtime](image)

20. Click **Update** to install the runtime.

The installation will proceed, and you'll be notified of a successful installation, as shown in Figure 3.20.
Installation with the SAP HANA Lifecycle Manager Command Line Interface

In order to install SAP HANA XSA from the command line interface, you’ll need to do the following:

1. Login to the SAP HANA system as the root user.
2. Browse to the location where the installation media is located.
3. Execute `/<install_path>/SID/hdbclnm` from the command line in the installation media directory.
4. Select the update option from the command line interface. In our example (Figure 3.21), our SID is HB1, so we selected 1 and pressed [Enter].
5. Select the Update SAP HANA XS Advanced Runtime option and press [Enter].
6. Enter the credentials for the system administrator, the system user, and the SAP HANA XSA administrator (Figure 3.22).
7. Select the components you want to install. Use comma-separated indices for multiple selections, or enter "1" for all.
8. Enter "y" on the keyboard and press [Enter] to confirm the selected components.
SAP HANA XSA should begin to install. When the installation has completed, you can navigate to the Administrator Console in SAP HANA Studio and verify that the SAP HANA XSA components are **active**, as shown in Figure 3.23.

![Figure 3.23: SAP HANA XSA Model Services Installed and Active](image)

### Installing SAP Web IDE

The SAP Web IDE is a component of the SAP XSA platform, as is reflected in some of the documentation for SAP HANA XSA. However, you should consider it a separate installation that requires many of the same components as the SAP HANA XSA runtime, because the SAP Web IDE will not install without including the SAP HANA XSA runtime as a **detected software component** at the beginning of the installation process.

#### Warning!

If you don’t navigate to the location where the runtime package is extracted and add it as a component in the lifecycle manager process, you will receive an unhelpful error message stating that the runtime is required to install the other components regardless of whether or not the platform has already been installed. Again, the installation process currently requires the installation files for the runtime platform as part of any component installation, so make sure you select the SAP HANA XSA platform as part of any component installation.

<table>
<thead>
<tr>
<th>SAP HANA XSA Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAP Web IDE web client</td>
<td>Comprehensive browser-based integrated development environment for complex applications using web-based or mobile user interfaces, business logic, and SAP HANA data models.</td>
</tr>
<tr>
<td>SAP Web IDE development infrastructure</td>
<td>Core component of the SAP Web IDE, required to enable the SAP HANA Web-Based Development Workbench in your environment.</td>
</tr>
<tr>
<td>XS Monitoring</td>
<td>Allows the maintenance of the application environment, including security, authentication, and resource monitoring.</td>
</tr>
<tr>
<td>XS Services</td>
<td>Enables using a catalog of services managed by the SAP HANA Service Broker, including OAuth clients, user accounts, and job scheduling.</td>
</tr>
<tr>
<td>SAP HANA runtime tools</td>
<td>Tools that help in design-time development and runtime administration of MTAs with SAP HANA XSA.</td>
</tr>
<tr>
<td>SAP HANA Demo Model for XS Advanced</td>
<td>SAP HANA Interactive Education (SHINE) content for SAP HANA XSA.</td>
</tr>
</tbody>
</table>

See Table 3.3 for a description of the SAP HANA XSA components.
3.3 Developing SAP HANA XSA Applications with SAP Web IDE

SAP HANA XSA applications may be developed using either the XS CLI or with the SAP Web IDE. In this section, we’ll provide an overview for creating SAP HANA XSA applications using both approaches. We’ll begin with a brief overview of using XS CLI, and then we’ll move on to develop a sample SAP HANA XSA application using the SAP Web IDE.

Before we jump right into developing with SAP Web IDE, let’s take a moment to understand the overall process for developing applications using SAP HANA XSA. To be successful in any SAP HANA XSA application project, you’ll need to apply the steps in the process below:

1. Create the folder infrastructure for your application files.
2. Create the deployment descriptor files.
3. Add database artifacts and content as needed.
4. Add business logic to work with the database artifacts.
5. Create OData service definitions as needed.
6. Create the user interface (UI).
7. Add security to the project.
8. Define application routes.
9. Create service instances for the application.
10. Add scheduled jobs as needed.
11. Deploy the application.
12. Test the application.

In this section, we’ll offer some simple examples to get you started on the path to developing your own SAP HANA XSA applications. Much of the content in this section, including the development of the Tiny World application, is contained in the SAP HANA Developer Guide for SAP HANA XS Advanced Model, available for download from SAP at http://help.sap.com.

As the SAP Web IDE is the preferred application for unlocking the power of SAP HANA as an application platform, we’ll cover the following steps in this section:

1. Getting the SAP Web IDE address.
2. Setting up a project.
3. Developing and running application modules.

### Note

SAP Web IDE is the preferred tool for developing SAP HANA XSA applications, so we’ll focus on this development tool, but the XS CLI can also be used to develop SAP HANA XSA applications. XS CLI is not very user-friendly and thus should only be used by experienced developers, so we won’t cover it in much detail in this section; however, you can find a walkthrough for XS CLI development at http://help.sap.com/. Navigate to TECHNOLOGY PLATFORM • SAP HANA PLATFORM • SAP HANA PLATFORM CORE SPS 12 • INTRODUCTION TO APPLICATION DEVELOPMENT AND DEPLOYMENT (XS ADVANCED MODEL) • GETTING STARTED WITH APPLICATION DEVELOPMENT IN XS ADVANCED • WORKING WITH THE XS ADVANCED COMMAND-LINE CLIENT.

We also used the Tiny-World application section from the WORKING WITH THE SAP WEB IDE FOR SAP HANA section of this site as the basis for the content in this section.

These topics should be enough to get you started with using SAP HANA XSA as your preferred application development platform; however, this section is not intended to be a comprehensive guide. For many more related subjects, including debugging, packaging modules, and deploying your application, you should consult the SAP HANA Developer Guide for SAP HANA XS Advanced Model for a complete guide to developing applications with SAP HANA XSA.

#### 3.3.1 Getting the SAP Web IDE Address

The first step in your SAP HANA XSA development efforts will be opening the SAP Web IDE. The XS CLI that we referred to earlier is essential because you’ll need to use the XS CLI to determine the URL for your SAP Web IDE installation.

For a guide to extracting the SAP HANA XS client, see SAP Note 2242468.

First, you need to obtain the SAP HANA XS client for your system. To get the client, you’ll need to extract the SAP HANA Platform installation media to your computer and then navigate to the DATA_UNITS • XSA_CLIENT_10 folder (Figure 3.24).

Select the ZIP file that matches your system (for example, the Windows client is in the xs.onpremise.runtime.client_ntamd64.zip file) and extract it to a folder on your system. If you’re on a Windows system, you can navigate to the folder where you extracted the files and hold down the right-clicking on the bin folder to open...
the alternate context menu for the folder. Select **Open command window here** from the context menu to open a command prompt in the bin folder.

![Figure 3.24 SAP HANA XS Client in the SAP HANA Platform Installation Media](image)

**Note**
The command line options in the following section will work regardless of whether you’re using a Windows or Linux system. For Linux, you need to open the **Terminal** window in the **bin** location and add the "./" prefix to all of the commands.

From the command prompt, you can use the XS CLI client to obtain the URLs for the SAP HANA XSA platform components, but first you’ll need to set the client environment and login. To set the client environment and configure a secure SSL connection between your system and the SAP HANA XSA platform, you’ll need to configure SSL for your system. See SAP Note 2242468 for information about configuring your system for SSL, or see your system administrator to obtain the SSL certificate for your server. With the certificate available, you can now connect your client to the server with the following command:

```
xs api https://<FQDN>:30<instance>30 -cacert <path/certificate>
```

Next you’ll need to login to the platform. To do so, enter `xs login` at the command prompt. You’ll be prompted to enter your user name and password, and you should be notified that your authentication was successful by the return of the system connection information, as shown in Figure 3.25.

![Figure 3.25 Connection Established to the SAP HANA XSA Runtime Platform](image)

**Note**
You should use command `xs -v` in the XS CLI to obtain the addresses for other important components in your SAP HANA XSA system. In particular, you’re going to want the address for the XS Advanced Administration and Monitoring Tools portal, where you’ll need to create spaces for your applications, add developer users, and assign users to development spaces. The `xsa-admin` service has the address for this portal, as shown in Figure 3.26.

![Figure 3.26 Addresses for SAP HANA XSA Service Portals](image)

To get the URL for your SAP Web IDE connection, enter `xs app webide -urls` at the command prompt. The web address for your SAP Web IDE connection will be displayed in the **COMMAND PROMPT** window.
3.3.2 Project Setup

With the address for the SAP Web IDE in hand, you're ready to set up your project. Navigate to the web address you found using the directions in the previous section and login to the platform with your SAP HANA XSA developer user. The SAP Web IDE will open (Figure 3.27), and you'll be ready to set up your project.

![Figure 3.27 SAP Web IDE for SAP HANA XSA Application Development](image)

From the SAP Web IDE, select **File -> New -> Project from Template**. The **Template Selection** window will open, and you should be able to select the template for a **Multi-Target Application Project** (Figure 3.28). Select the template and click **Next**.

![Figure 3.28 Creating an SAP HANA XSA Application from a Template](image)

Enter a name for the project and click **Next**. Since we're using the tutorial from the developer guide, we entered “TinyWorld” for our **Project Name** (Figure 3.29). Click **Next** to advance to the **Template Customization** tab.

![Figure 3.29 Creating the TinyWorld Application](image)

On the **Template Customization** tab, you can specify the **Application ID**, **Application Version**, **Description**, and development **Space** for the new application (Figure 3.30). Enter the required information and click **Next** to advance to the **Confirmation** tab. Click **Finish** to create the new project.

![Figure 3.30 Assigning Descriptive Information to Your SAP HANA XSA Project](image)
The new application will appear in the Workspace folder in the SAP Web IDE, and a file named `mta.yaml` will be the only file in the new project. This file is the MTA descriptor file, which contains a record of any dependencies or prerequisites required when the application is ready for deployment. This file will grow as you add code and modules to your project. With the project created, you’re ready to begin developing the new project.

### 3.3.3 Developing and Running Application Modules

Next, we’re going to add a database module to the project. To add this module, right-click the `TinyWorld` project folder and select **New HDB Module** from the context menu. The **New HDB Module** window will open, and you’ll need to provide a **Module Name** for the project. In our example (Figure 3.31), we’ve named our module the “tinyWorldDB.” Click **Next** to advance to the **Template Customization** tab.

![Figure 3.31 Creating a New SAP HANA Database Module](image)

Confirm the **Initial View Details** for the **Namespace**. If you’ve been following along with our example, you should have `TinyWorld.tinyWorldDB` already entered here. Click **Next** to advance to the **Confirmation** tab and click **Finish** to create the new database module. The `TinyWorldDB` object folder will be added to the `TinyWorld` project folder, and the new object folder will contain a folder named `src`, which will appear to be empty upon first inspection; however, if you’ll select **View ▶ Show Hidden Files** from the SAP Web IDE menu, you’ll see that the `src` folder actually contains files named `.hdiconfig` and `.hdinamespace`. The `.hdiconfig` file is required for SAP HANA XSA projects and specifies the plug-ins used by SPA HANA to create a catalog object. The `.hdinamespace` file specifies rules for runtime namespaces and is optional.

Next, we’ll need to create a simple database table in our project using core data services (CDS) programming, an enhanced and extended version of SQL used to define and consume data models. Use the following steps to create the new data model:

1. Right-click the `src` folder in the `TinyWorldDB` folder and select **New CDS Artifact** from the context menu.
2. Enter “tinyTable” as the name in the **Create New CDS file** window (Figure 3.32).

![Figure 3.32 Creating a New CDS Data Model](image)

3. Select **Graphical** as the **Editor** and make sure the **With Context** checkbox is selected.
4. Click **Create** to create the new data model and verify that the file named `tinyTable.hdbcds` is created in the `src` folder.
5. Double-click the new file to open the editor.
6. Enter the code from Listing 3.2 into the editor pane.

```hdbcds
namespace TinyWorld.tinyWorldDB;

class world {
  key continent: String(100);
};
```

**Listing 3.2** Code for the `tinyTable.hdbcds` File
Note

The code in Listing 3.2 defines a table named "world" with a field named "continent" and a field type of up to 100 characters in a string.

7. Save the HDB module with the Save icon or with Ctrl+S on the keyboard.
8. Select Build • Build from the file menu in the SAP Web IDE.

You should receive a message stating that the build was completed successfully in the console window, which indicates that your simple data model has been activated in the SAP HANA database and can be used for storing and retrieving data.

Next, we'll create a Node.js module based on the XSJS library. Use the following directions to build a simple Node.js module:

1. Right-click the TinyWorld folder to open the context menu.
2. Select New Node.js Module from the context menu.
3. Name the module "tinyWorldJS."
4. Select the checkbox to Enable xsjs support.
5. Click Finish.
6. Double-click the lib/index.xsjs file to open the file in the JavaScript editor.
7. In the JavaScript editor, replace the default code with the following:
   ```javascript
   $.response.contentType = "text/html";
   $.response.setBody("Welcome to the tiny JS World!");
   ```
8. Save the file with Ctrl+S.

9. Right-click the tinyWorldJS folder to open the context menu.
10. Select Run • Run as • Node.js Application.

A new browser tab will open with the address for the new XSJS module (or in a new window, depending on your browser settings). By choosing to run the application, we implicitly tell the editor to build the application and then run it, so you can return to the SAP Web IDE console to monitor the progress. You should soon see the application status change to Status: Running, and you should have the message, "Welcome to the tiny JS World!" in the browser window for the application.

Finally, the process to add an HTML5 module to your new application is nearly identical to the one used for Node.js, with the obvious exception being the difference in programming languages between HTML5 and Node.js. Follow the process below to add an HTML5 module:

1. Right-click the TinyWorld folder to open the context menu.
2. Select New HTML5 Module from the context menu.
3. Name the module "tinyWorldUI."
4. Click Finish.
5. Double-click the resources/index.html file to open the file in the editor.
6. Replace the default code with the following:
   ```html
   <!DOCTYPE html>
   <html>
   <body>
   Welcome to your Tiny HTML World!
   </body>
   </html>
   ```
7. Save the file with Ctrl+S.

8. Right-click the tinyWorldJS folder to open the context menu.
9. Select Run • Run as • Web Application.

As with Node.js modules, a new browser tab or window should open, and the text "Welcome to your Tiny HTML World!" should appear in the new window.

We'll wrap up our discussion of SAP Web IDE here for now, as this should be enough to get you started. For a more in-depth exploration of creating SAP HANA XSA applications, you can download the SAP HANA Developer Guide for SAP HANA XS Advanced Model, or you can access the excellent series of blogs by Chaim Bendelac on the subject called Developing with XS Advanced: A TinyWorld Tutorial, which covers both basic and advanced development with SAP HANA XSA. Part 1 of the blog can be accessed at http://scn.sap.com/community/developer-center/hana/blog/2016/03/28/developing-with-xs-advanced-a-tinyworld-tutorial.

### 3.4 Summary

In this chapter, we explored the capabilities you can unlock with SAP HANA as your application platform. You can develop basic applications through the SAP HANA Web-Based Development Workbench, but the SAP Web IDE is the latest development interface for creating customized, high-performance web applications. The development interface you choose will depend on the support package stack of your SAP HANA system, and the SAP Web IDE is frequently updated to enhance its effectiveness and ease of use. The combination of SAP HANA as your
application platform and the new, powerful web-based development tools provides you with the flexibility to build anything you want with virtually no limitations on one of the fastest database systems in the world: SAP HANA.
Contents

Acknowledgments ............................................................................................ 15
Introduction ..................................................................................................... 17

1 In-Memory Computing, Big Data, and SAP HANA ................... 19
  1.1 Introduction to In-Memory Computing and Big Data ..................... 20
    1.1.1 In-Memory Computing and Analytics ............................... 20
    1.1.2 Big Data ........................................................................... 26
  1.2 Column-Based versus Row-Based Storage ...................................... 31
  1.3 Summary ....................................................................................... 38

2 SAP HANA as a Database ......................................................... 39
  2.1 The Fundamentals of SAP HANA ............................................ 39
    2.1.1 What SAP HANA Can Do ................................................. 41
    2.1.2 What SAP HANA Can't Do ............................................... 43
  2.2 Big Data Solutions ............................................................... 44
    2.2.1 SAP HANA ....................................................................... 44
    2.2.2 Dynamic Tiering ............................................................... 47
    2.2.3 SAP HANA Smart Data Access ....................................... 50
    2.2.4 SAP HANA Vora and Hadoop ........................................... 54
  2.3 SAP HANA for Non-SAP Use Cases ............................................. 56
    2.3.1 SAP HANA for Non-SAP Analytics .................................... 57
    2.3.2 SAP HANA for Non-SAP Applications ............................... 69
  2.4 SAP BW on SAP HANA ......................................................... 71
    2.4.1 Technical Requirements ................................................... 77
    2.4.2 Skill Requirements ........................................................... 80
    2.4.3 Project Plan Steps ............................................................ 83
    2.4.4 Choosing SAP BW on SAP HANA ..................................... 85
  2.5 SAP Business Suite Powered by SAP HANA ................................ 86
  2.6 SAP S/4HANA ......................................................................... 89
  2.7 Other SAP HANA Capabilities .................................................. 90
    2.7.1 SAP HANA Smart Data Streaming .................................... 91
    2.7.2 SAP HANA Predictive Analysis Library ............................. 93
    2.7.3 SAP HANA Graph Engine ................................................ 94
    2.7.4 SAP HANA Sentiment Analysis ....................................... 95
  2.8 Summary ....................................................................................... 96
<table>
<thead>
<tr>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>3</strong> SAP HANA as an Application Platform ......................................................... 97</td>
</tr>
<tr>
<td><strong>3.1</strong> SAP HANA XS Classic ................................................................................. 98</td>
</tr>
<tr>
<td>3.1.1 Architecture ....................................................................................... 99</td>
</tr>
<tr>
<td>3.1.2 Developing SAP HANA XS Classic Applications ..................................... 100</td>
</tr>
<tr>
<td><strong>3.2</strong> SAP HANA XSA ..................................................................................... 109</td>
</tr>
<tr>
<td>3.2.1 SAP HANA XSA Architecture .................................................................. 111</td>
</tr>
<tr>
<td>3.2.2 Installing SAP HANA XSA Runtime ...................................................... 116</td>
</tr>
<tr>
<td><strong>3.3</strong> Developing SAP HANA XSA Applications with SAP Web IDE ............... 130</td>
</tr>
<tr>
<td>3.3.1 Getting the SAP Web IDE Address ....................................................... 131</td>
</tr>
<tr>
<td>3.3.2 Project Setup ......................................................................................... 134</td>
</tr>
<tr>
<td>3.3.3 Developing and Running Application Modules ....................................... 136</td>
</tr>
<tr>
<td><strong>3.4</strong> Summary ................................................................................................. 139</td>
</tr>
<tr>
<td><strong>4</strong> SAP HANA Cloud Platform ........................................................................... 141</td>
</tr>
<tr>
<td><strong>4.1</strong> Overview ................................................................................................. 142</td>
</tr>
<tr>
<td>4.2 Features ...................................................................................................... 144</td>
</tr>
<tr>
<td>4.2.1 Access and Authorizations ..................................................................... 145</td>
</tr>
<tr>
<td>4.2.2 Enablement Services ............................................................................... 146</td>
</tr>
<tr>
<td>4.2.3 Schemas ................................................................................................. 146</td>
</tr>
<tr>
<td>4.2.4 Content Management ............................................................................. 147</td>
</tr>
<tr>
<td>4.2.5 Mobile Services ...................................................................................... 147</td>
</tr>
<tr>
<td>4.3 SAP HANA Cloud Platform Cockpit .............................................................. 148</td>
</tr>
<tr>
<td>4.4 SAP HANA Cloud Portal .............................................................................. 151</td>
</tr>
<tr>
<td><strong>4.5</strong> Getting Started ......................................................................................... 153</td>
</tr>
<tr>
<td><strong>4.6</strong> Summary ................................................................................................. 155</td>
</tr>
<tr>
<td><strong>5</strong> SAP S/4HANA ............................................................................................. 157</td>
</tr>
<tr>
<td><strong>5.1</strong> Overview ................................................................................................. 157</td>
</tr>
<tr>
<td><strong>5.2</strong> SAP S/4HANA Finance ............................................................................. 162</td>
</tr>
<tr>
<td>5.2.1 Universal Journal .................................................................................... 163</td>
</tr>
<tr>
<td>5.2.2 Period Close ............................................................................................ 164</td>
</tr>
<tr>
<td>5.2.3 Asset Accounting .................................................................................... 165</td>
</tr>
<tr>
<td>5.2.4 SAP Cash Management .......................................................................... 165</td>
</tr>
<tr>
<td>5.2.5 SAP Material Ledger ............................................................................... 165</td>
</tr>
<tr>
<td>5.2.6 SAP BPC for S/4HANA Finance ............................................................... 166</td>
</tr>
<tr>
<td><strong>5.3</strong> SAP S/4HANA Materials Management and Operations ......................... 167</td>
</tr>
<tr>
<td>5.3.1 Manufacturing ......................................................................................... 168</td>
</tr>
<tr>
<td>5.3.2 Supply Chain ............................................................................................ 169</td>
</tr>
<tr>
<td><strong>5.4</strong> Business Case for SAP S/4HANA ............................................................... 170</td>
</tr>
<tr>
<td><strong>5.5</strong> Migration from SAP ERP to SAP S/4HANA ........................................... 172</td>
</tr>
<tr>
<td><strong>5.6</strong> SAP Activate ........................................................................................... 174</td>
</tr>
<tr>
<td><strong>5.7</strong> Summary ................................................................................................. 175</td>
</tr>
<tr>
<td><strong>6</strong> Planning an SAP HANA Implementation ...................................................... 177</td>
</tr>
<tr>
<td><strong>6.1</strong> Technical Basics ....................................................................................... 177</td>
</tr>
<tr>
<td>6.1.1 In-Memory Computing Engine ................................................................ 177</td>
</tr>
<tr>
<td>6.1.2 Software Specifications ........................................................................... 179</td>
</tr>
<tr>
<td>6.1.3 Software Editions for Specific Purposes .................................................. 181</td>
</tr>
<tr>
<td>6.1.4 Network Speed between Servers ............................................................. 187</td>
</tr>
<tr>
<td>6.1.5 Software Install ....................................................................................... 187</td>
</tr>
<tr>
<td><strong>6.2</strong> Hardware Specifications and Options ....................................................... 189</td>
</tr>
<tr>
<td>6.2.1 Scaled-Up SAP HANA Systems ................................................................ 190</td>
</tr>
<tr>
<td>6.2.2 Scaled-Out SAP HANA Systems ............................................................... 192</td>
</tr>
<tr>
<td>6.2.3 Hardware Platforms ................................................................................ 193</td>
</tr>
<tr>
<td><strong>6.3</strong> SAP HANA as a Standalone Data Warehouse ........................................... 194</td>
</tr>
<tr>
<td>6.3.1 Data Modeling ......................................................................................... 195</td>
</tr>
<tr>
<td>6.3.2 SAP Business Suite on SAP HANA ........................................................ 196</td>
</tr>
<tr>
<td><strong>6.4</strong> Sizing ........................................................................................................ 200</td>
</tr>
<tr>
<td>6.4.1 New Features ........................................................................................... 201</td>
</tr>
<tr>
<td>6.4.2 Implementation Options .......................................................................... 203</td>
</tr>
<tr>
<td>6.4.3 Sizing ....................................................................................................... 208</td>
</tr>
<tr>
<td><strong>6.5</strong> SAP S/4HANA ........................................................................................ 213</td>
</tr>
<tr>
<td>6.5.1 Moving to SAP S/4HANA Cloud ............................................................. 214</td>
</tr>
<tr>
<td>6.5.2 Moving to On-Premise SAP S/4HANA ..................................................... 215</td>
</tr>
<tr>
<td><strong>6.6</strong> SAP Business Warehouse on SAP HANA ............................................... 218</td>
</tr>
<tr>
<td>6.6.1 Sizing ....................................................................................................... 220</td>
</tr>
<tr>
<td>6.6.2 Preparing for a Migration ........................................................................ 228</td>
</tr>
<tr>
<td>6.6.3 Performing a Migration .......................................................................... 233</td>
</tr>
<tr>
<td>6.6.4 Optimizing a Migration .......................................................................... 237</td>
</tr>
<tr>
<td><strong>6.6.5</strong> New Features in SAP BW 7.4 .............................................................. 244</td>
</tr>
<tr>
<td><strong>6.6.6</strong> New Features in SAP BW 7.5 .............................................................. 247</td>
</tr>
<tr>
<td><strong>6.7</strong> Summary ................................................................................................. 249</td>
</tr>
<tr>
<td><strong>7</strong> Reporting with SAP HANA ............................................................................ 251</td>
</tr>
<tr>
<td><strong>7.1</strong> Overview of Tools ..................................................................................... 251</td>
</tr>
<tr>
<td>7.1.1 SAP BusinessObjects Dashboards ......................................................... 251</td>
</tr>
<tr>
<td>7.1.2 SAP BusinessObjects Web Intelligence ................................................. 253</td>
</tr>
</tbody>
</table>
8 Data Modeling with SAP HANA ........................................ 297

8.1 Data Modeling with SAP HANA Studio ......................... 298
  8.1.1 SAP HANA Development Perspective ......................... 299
  8.1.2 Creating and Importing Tables and Data .................... 304
  8.1.3 Generating Time Data .............................................. 309
  8.1.4 Migrating Attribute and Analytic Views ....................... 310
  8.1.5 Delivery Units and Packages ...................................... 316

8.2 Calculation Views: Dimensional Views .......................... 318
  8.2.1 Dimensional Views .................................................. 319
  8.2.2 Adding Projection Nodes ........................................... 323
  8.2.3 Applying Filters ..................................................... 326
  8.2.4 Union Nodes .......................................................... 328
  8.2.5 Scenarios Pane Controls ............................................ 330
  8.2.6 String Functions .................................................... 332
  8.2.7 Input Parameters ..................................................... 334
  8.2.8 Semantics Node ...................................................... 339

8.3 Calculation Views: Cube Views .................................... 345
  8.3.1 Join Nodes ............................................................ 347
  8.3.2 Temporal and Spatial Joins ......................................... 353
# Index

### A

- ABAP Routine Analyzer program, 238
- Abstraction layer, 452
- Access Control Lists (ACLs), 147
- Active concurrent users (ACU), 190
- Active server, 508
- Administration, 63, 80
  - adding systems, 462
  - Administration Editor, 62, 467
  - alerts, 480–481
  - changing configurations, 468
  - changing file locations, 467
  - Configuration tab, 468, 483
  - diagnosis mode, 467
  - disk full event, 489
  - exploring and importing systems, 465
  - full disk event, 489
  - installation, 465
  - load balancing, 489
  - managing large tables, 486
  - monitoring, 477–478
  - monitoring disk usage, 478
  - monitoring memory, 485
  - monitoring performance, 479
  - monitoring services, 481
  - monitoring with alerts, 480
  - partitioning, 486–488
  - Services tab, 481–482
  - System Information tab, 459
  - table data, 483
  - tools, 116
  - unresponsive system, 490
  - updates, 490, 492
- Administration Console, 460
- accessing, 461
- modifying, 468
- Administrator Workbench, 241
- Advanced DataStore objects (aDSOs), 220
- Aggregates, 76, 79, 85, 270
- aggregation engine, 74
- Aggregation nodes, 354
- Amazon Web Services, 509
- Amount with currency, 358
- Analytics views, 270, 277
- Apache Spark, 528
- Application router, 111
- Application sandboxes, 142
- Asset Accounting (FI-AA), 163, 165
- Atomicity, consistency, isolation, and durability (ACID), 24–25, 94
- Attribute views, 270, 277

### B

- Backup, 505
- Bank Account Management, 165
- Batch job, 427, 429–430, 440, 445
- BEx Query Designer, 248
- BEx Web Application Designer, 252, 261
- BI Consumer Services (BICS) connection, 282
- BI Launchpad, 394
- BI data, 20, 26, 28–30, 44, 46–47, 169
- Bitmap index, 32
- Blue-green deployment technique, 112
- Business analysts, 81
- Business Content repository, 68
- Business Intelligence Consumer Services (BICS), 252

### C

- Calculation scenario, 241
- Calculation views, 270, 277, 310, 318
  - applying filters, 326
  - columns, 325
  - create new, 323
  - cube, 339, 345
  - dimensional, 319
  - extract semantics, 343
  - filter operators, 328
  - graphical, 326
  - input parameter types, 337
  - input parameters, 314
Calculation views (Cont.)
joins, 347
semantics node, 339
union nodes, 328
Cash Operations, 165
Central Management Console (CMC), 267
Change and Transport System, 533
Change data capture (CDC), 428, 430
mechanism, 429
Change management, 170
Cisco, 189, 192
Cisco/EMC, 213
Cloud Foundry, 110
CMISQL, 146
Columnar tables, 270
Column-based storage, 31–32, 35–37
advantages and disadvantages, 33
versus row-based storage, 37
CompositeProviders, 220, 245
Compressed request, 244
Compression, 35
Configuration and Monitoring Dashboard, 404
Connectivity Service, 145
Content Management Interoperability Service (CMIS), 146, 151
Controller module, 400
Controlling (CO), 163
Converged System 500, 192
Conversion checkbox enable, 358
Core Data Services (CDS), 218
CPU, 78
Cross-platform data consumption, 517
Currency conversion, 357
for a measure, 358
table, 357
Customer engagement, 31
Customer Vendor Integration (CVI), 216

D

Data aging, 201
Data compression, 23
Data federation, 452
Data flow, 424
Data governance, 30, 46
Data integrity, 30, 46
Data Lifecycle Manager (DLM), 524
administration, 527
installation, 524
monitoring KPIs, 527
Data Migration Server (DMIS), 215, 403
Data modeling, 67, 83, 195, 247, 297
advanced DataStore Objects (aDSOs), 367
calculations views, 318
SAP BW modeling tools, 362
SAP HANA Studio, 298
SAP HANA Web-Based Development Workbench, 370
Data provisioning, 385
choosing a method, 385
corporate data, 392
real-time data, 390
supplemental data, 393
technical considerations, 395
Data redundancy, 35
Data replication, 187
Data scope, 30, 45
Data storage, 79
Data stores, 432
Data transfer process (DTP), 229, 242, 400
Data virtualization, 452
Data volume, 30, 45
Database development objects, 302
Database migration, 80
Database Migration Option (DMO), 206–207, 216, 231
Database module, 136
Database schema, 405
Database structured query language (DBSQL), 256
DataStore Objects (DSOs), 48, 229, 235, 238
DB2, 80
DBA Cockpit, 533–534, 536
DB Performance monitor, 535
transactions, 534
Decimal shifts checkbox, 359
Dedicated host deployment, 521
Default session client, 361
Dell, 189, 213
Delta buffer, 34
Denormalized tables, 195
Dependent objects, 312
Development standards, 103
Dictionary compression, 36
Dimensional data modeling, 244
Dimensional keys (DIM IDs), 240
Dimensional modelers, 195
Disaster recovery, 25, 81
Disk space monitoring, 478
Disk storage, 177
Document Service, 145
DSN connection, 269
Dynamic join, 350
Dynamic runtime memory, 225
Dynamic tiering, 47, 368, 519, 524
administering and monitoring, 522
hosts, 520
query optimizations, 521
E

E and F tables, 228
Eclipse, 154, 220, 298
Mars, 154
modeling for SAP BW, 246
Neon, 154
Eclipse IDE, 292
EDGE license, 181
Enhanced CompositeProviders (ECPs), 247
Enterprise Connect Data Access (ECDA), 449
Enterprise data warehouse (EDW), 190
Enterprise information management (EIM), 65, 81
ETL-based replication, 387, 423
Excel 2010 standard MDX, 277
Exploration views, 255
Export and import table data, 483
Export wizard, 484
Expression Editor, 322–333
Extended store tables, 522
Extended tables, 50, 520
Extract, transform, and load (ETL), 44, 67–68, 180, 385, 423
tools, 196
F

Facet, 255
Failover system, 508
Feedback Service, 146
File replication lists, 287
Filter expression, 327, 335
Flow logic, 408
Frank, Christopher, 29
Fujitsu, 189, 213
Full refresh, 428
G

Geometries, 353
Graph database, 94
H

Hadoop, 54, 388, 454, 528
Hadoop Distributed File System (HDFS), 54, 454
Hadoop scripts, 528
High availability, 508
History table, 322
Hitachi, 189, 213
Hot data, 519
HP, 213
HP Sizing Tool for SAP Business Suite, 213
HTML, 98
HTML5 module, 138
HTTPS protocol, 143
IBM, 189, 213
IBM Labs, 193
Identity Service, 145
Index server, 508
Industry Solutions, 200
InfoCubes, 229, 235, 241
after optimization, 241
InfoCubes (Cont.)
before optimization, 240
fact tables, 242–243
future of, 244
SAP HANA optimized, 240–241
InfoObjects, 365
Information Design Tool (IDT), 267, 270, 273,
275–277
Information model, 270
Information spaces, 254
In-memory computing, 20, 22–24
In-memory computing engine (IMCE), 40, 100,
177
In-memory database (IMDB), 21
Intel, 189
Internet of Things (IoT), 55, 92, 161
Inventory cube, 243
Investment protection, 193
ISO 27001, 142
Java, 98, 154
Java Database Connectivity (JDBC), 53, 97,
263, 388
Java on Apache Tomcat, 110
Java on Apache TomEE, 110
Java Persistence API (JPA), 144
Java Virtual Machine (JVM), 155
JavaScript, 109
JavaScript editor, 138
Join nodes, 347
Join type, 351
full, 352
inner, 351
referential, 351
right outer, 352
text, 352
Kerberos, 496
Key figures, noncumulative, 242

L
Lenovo, 192, 194
License keys, 502
audit, 505
permanent, 502, 504
temporary, 502–503
Lightweight Directory Access Protocol (LDAP),
115
Linux, 181, 387
Load balancing across hosts, 489
Log storage, 78
Log-based replication, 385, 387, 448
LSA++, 229, 238
architecture, 219

M
M_Service_Memory, 485
Main memory, 23–24
MapReduce programming paradigm, 454
Massive parallel processing (MPP), 71
Master job, 406
Master server, 508
Material requirements planning (MRP), 168
Measures
linking with currencies, 362
type, 358
Memory consumption, 485
Micro-service container, 112, 114
Microsoft Excel, 57, 62, 280–281
Microsoft query, 281–282
Mobile Services, 147
Moneyball, 29
Multidimensional expressions (MDX), 41, 62,
179, 267
Multi-level partitioning, 487
Multinode, 79
Multinode system, 518
Multiple Components One Database (MCOD),
512
Multiple Components One System (MCOS),
512–513
MultiProviders, 76
Multi-target application (MTA), 112
deployment descriptor file, 113

P
Parallel reading, 424
Parameter-based rules, 408
Partition, round-robin, 488
Password policy, 501
Period close, 164
Persistence and logging, 178
Persistence Service, 145
Persistent Staging Area (PSA), 224
Persistent storage, 25
Physical memory, 478, 485
Plan-to-produce, 160
Platform as a service (PaaS), 141
Power user, 81
Predictive Analysis Library (PAL), 93
Private views, 291
Procure-to-pay, 160
Product Availability Matrix (PAM), 395
Profitability Analysis (CO-PA), 163
Project phases, 67
Project plan steps, 67
SAP BW on SAP HANA, 83–84
SAP HANA as a data warehouse, 66
R4load, 449
RAM, 78
Range partition, 487
Rank nodes, 356
Rapid Deployment Solutions (RDS), 180, 220,
424
Rauhe, Hannes, 37
Read module, 400
Real time, 58, 61, 65, 69, 404
<table>
<thead>
<tr>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAP BusinessObjects Analysis, 221, 259–260, 268</td>
</tr>
<tr>
<td>Microsoft edition, 259</td>
</tr>
<tr>
<td>OLAP, 259</td>
</tr>
<tr>
<td>what it can't do, 259</td>
</tr>
<tr>
<td>what it can do, 259</td>
</tr>
<tr>
<td>SAP BusinessObjects Analysis for Microsoft Office, 268</td>
</tr>
<tr>
<td>big data, 46–47</td>
</tr>
<tr>
<td>connecting to SAP HANA, 267, 270</td>
</tr>
<tr>
<td>SAP BusinessObjects BI Launchpad, 253</td>
</tr>
<tr>
<td>SAP BusinessObjects Business Intelligence, 19</td>
</tr>
<tr>
<td>SAP BusinessObjects Dashboards, 251, 253</td>
</tr>
<tr>
<td>SAP BusinessObjects Design Studio, 252, 250</td>
</tr>
<tr>
<td>SAP BusinessObjects Explorer, 59, 254–256</td>
</tr>
<tr>
<td>information spaces, 256</td>
</tr>
<tr>
<td>SAP BusinessObjects Live Office, 260</td>
</tr>
<tr>
<td>SAP BusinessObjects Lumira, 251, 263, 530</td>
</tr>
<tr>
<td>picking up SAP HANA views, 266</td>
</tr>
<tr>
<td>visualization, 265</td>
</tr>
<tr>
<td>SAP BusinessObjects Web Intelligence, 253</td>
</tr>
<tr>
<td>SAP BW, 53, 68, 228</td>
</tr>
<tr>
<td>7.3 Unicode, 77</td>
</tr>
<tr>
<td>7.4, 73–74, 218, 245</td>
</tr>
<tr>
<td>7.4 new features, 244</td>
</tr>
<tr>
<td>7.5, 218</td>
</tr>
<tr>
<td>7.5 new features, 247</td>
</tr>
<tr>
<td>ABAP Analyzer, 239</td>
</tr>
<tr>
<td>Administrator Workbench, 222</td>
</tr>
<tr>
<td>cubes, 68, 71, 79</td>
</tr>
<tr>
<td>system housekeeping, 228</td>
</tr>
<tr>
<td>SAP BW Accelerator, 48, 73, 241, 255, 391</td>
</tr>
<tr>
<td>SAP BW Analytical Engine, 536</td>
</tr>
<tr>
<td>SAP BW Metadata Cockpit for SAP HANA, 220, 225, 230–231, 238</td>
</tr>
<tr>
<td>SAP BW modeling tools, 247, 297, 362</td>
</tr>
<tr>
<td>CompositeProviders, 365</td>
</tr>
<tr>
<td>installation details, 364</td>
</tr>
<tr>
<td>Open ODS views, 369</td>
</tr>
<tr>
<td>SAP BW on SAP HANA, 26, 44, 71, 80, 82, 85, 179, 218, 391</td>
</tr>
<tr>
<td>advantages, 71</td>
</tr>
<tr>
<td>architecture, 78</td>
</tr>
<tr>
<td>implementation options, 71</td>
</tr>
<tr>
<td>migrate a copy, 236</td>
</tr>
<tr>
<td>migration, 233</td>
</tr>
<tr>
<td>migration approaches, 237</td>
</tr>
</tbody>
</table>

| Sales and operations planning, 169 |
| Same host deployment, 521 |
| Sandbox environment, 510 |
| SAP (Sybase) IQ, 536 |
| SAP (Sybase) Replication Agent, 448 |
| SAP (Sybase) Replication Server, 40, 57, 179, 385, 448 |
| installation, 450 |
| overview, 387 |
| SAP Activate, 174 |
| SAP Active Embedded, 175 |
| SAP BPC for S/4HANA Finance, 166 |
| SAP Business Client, 406 |
| SAP Business Explorer (BEx) Web Analyzer, 221 |
| SAP Business Suite, 255 |
| SAP Business Suite on SAP HANA, 86, 89, 163, 179, 198, 200 |
| architecture, 88 |
| greengrass implementation, 203 |
| implementation options, 87, 203 |
| in-place migration, 205 |
| requirements, 87 |
| SAP Notes, 202 |
| sizing, 208 |

| SAP BW on SAP HANA (Cont.) |
| migration optimization, 227 |
| migration prechecks, 231 |
| migration preparation, 228 |
| partial migration, 235 |
| requirements, 77, 80 |
| SAP Notes, 234 |
| sizing, 220 |
| standard migration, 234 |
| SAP BW Workspace Query Designer, 247 |
| SAP Cash Management, 165 |
| SAP Cloud Identity, 115 |
| SAP CO-PA Accelerator, 512 |
| SAP Crystal Reports, 262 |
| SAP Crystal Reports Enterprise, 263 |
| SAP Customer Relationship Management (SAP CRM), 87 |
| SAP Data Services, 40, 54, 58, 61, 65, 67, 83, 180, 218, 385, 391–392, 423 |
| 4.2, 424 |
| configuration, 424 |
| connecting to SAP HANA, 426 |
| creating batch jobs, 440 |
| data flows, 440 |
| data update mode, 428 |
| defining metadata, 438 |
| executing batch jobs, 445 |
| File Format Editor, 438 |
| importing metadata, 437 |
| loading data, 428 |
| Local Object Library, 434, 437 |
| metadata replication, 425 |
| overview, 387 |
| receiving data, 425 |
| setting up data stores, 433 |
| SAP Data Services Designer, 432–433 |
| project area, 434 |
| SAP EarlyWatch Alert report, 221, 533 |
| SAP ERP, 200 |
| SAP ERP 6.0, 202 |
| SAP ERP Financials, 209 |
| SAP ERP Material Management, 211 |
| SAP ERP Sales and Distribution, 284 |
| SAP Finance and Controlling Accelerator, 512 |
| SAP Fiori, 158, 201, 216 |
| SAP Fiori launchpad, 471 |
| SAP General Ledger (G/L), 163 |

| SAP GUI, 39 |
| server, 204 |
| SAP HANA, 19, 40 |
| application platform, 97 |
| application services, 183 |
| as a database, 61 |
| big data, 46–47 |
| business analysts, 64 |
| calculation and analytic views, 261 |
| capabilities, 58 |
| data services, 185 |
| database, 39, 182 |
| distributed system, 486 |
| editions, 179–180 |
| end user clients, 185 |
| for non-SAP applications, 69 |
| hardware options, 190 |
| hardware platforms, 193 |
| hardware specifications, 189 |
| host system roles, 123 |
| implementation, 177 |
| industries, 43 |
| installation roles, 188 |
| job roles, 62 |
| lifecycle management, 182 |
| Linux host, 120 |
| native applications, 184 |
| network speed, 187 |
| optimizations, 239 |
| pilot program, 25, 59 |
| reporting, 251 |
| SAP Notes, 186 |
| scaled-out systems, 192 |
| scaled-up systems, 190 |
| sidecar implementation, 57, 290, 369 |
| sizing, 196 |
| software components, 182 |
| software specifications, 179 |
| standalone data warehouse, 194 |
| standalone data warehouse sizing, 198 |
| standalone implementation for analytics, 179 |
| updating versus inserting, 34 |
| use cases, 43 |
| views, 266 |
| what it can do, 41 |
| what it can't do, 43 |
| SAP HANA Application Lifecycle Management cockpit, 526 |
First-hand knowledge.

Penny Silvia, Rob Frye, Bjarne Berg

SAP HANA: An Introduction
549 Pages, 4th edition 2016, $69.95

wehopeyouhaveenjoyedthisreadingsample.youmayrecommend
orpassitontoothers,butonlyinitsentirety,includingallpages.This
readingsampleandallitspartsareprotectedbycopyrightlaw.Allusage
andexploitationrightsarereservedbytheauthorandthepublisher.