Administration Guide | PUBLIC
Document Version: 1.23 – 2024-07-12

Administrator's Guide for SAP Financial Services Data Management

THE BEST RUN SAP

Content

1 Getting Started

SAP Financial Services Data Management aims to support you as a financial services provider in the further digitalization of your core business in the financial industry by establishing a business technology platform. The aim is to benefit from your own historical data as well as from market driven aspects in order to be able to make targeted decisions for your company more quickly.

SAP Financial Services Data Management enables you to store your company data in one single source and use it for any use case through efficient processes. This central data storage offers the following advantages for your company:

- **One single source of truth**: to avoid redundancies you store data for the same object only once.
- **One format**: you store data for different objects of the same class in the same format even though it might come from different source systems. This makes internal processes and communication much easier as your entire team has a common understanding about one unified data model.
- **Various use cases**: you use the same data for various use cases, which makes the reconciliation of your results much easier.

The unified data model provides a basic framework for the format in which the data is stored. As a standardized and business-driven data model for financial services, it comprises financial services topics such as business partner, bank accounts, or reinsurance, each describing a subset of semantically related entities.

In conjunction with the SAP HANA database technology, the unified data model uses a flexible data source-agnostic, in-memory data platform that allows you to analyze large volumes of data in real time. The database services of the SAP HANA technology enable you to use standardized interfaces to store and access your data in-memory and column-based.

The association between the unified data model and the SAP HANA database technology helps you to create a highly modern data platform that you can use as a solid basis for your analytical scenarios or procedures. This supports you in fulfilling regulatory requirements and reporting purposes.

Please note that SAP Financial Services Data Management can only collect the data as you submit it. Data consistency checks are not the focus of the product. You must, therefore, guarantee the quality of the data in your source systems and during the data loading process.

> ⚠ Caution
>
> Make sure that your license agreement from your source system database allows indirect access to the database to read and extract data. If you have additional questions about what your license provides or wish to discuss licensing features, please contact your SAP account team representative.

> ⚠ Caution
>
> The SQL-based data storage approach allows you to individually access your data using SQL. Please keep in mind that you can write or retrieve data from or to tables using only the generated procedures and views provided by the interface generator because direct writing or reading from or to the tables ignores the versioning of the data and your customer-specific extensions of the data model.

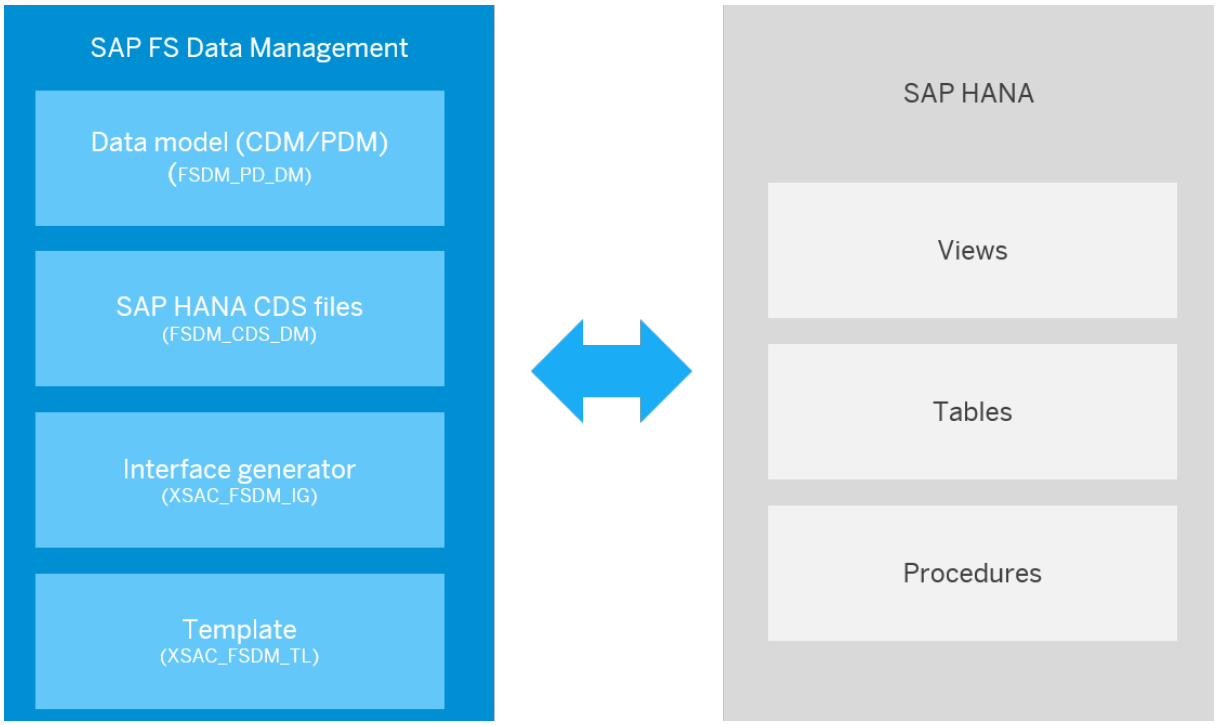
More Information

- Release information note for SAP Financial Services Data Management: 2559644
- Product page for SAP HANA Platform

1.1 What You Need to Know About the Software Architecture

Software Components

The following figure shows an overview of the software components provided by SAP Financial Services Data Management:



Data model (CDM/PDM): software component FSDM PD DATA MODEL (FSDM_PD_DM)

A unified, standardized and business-driven data model for financial services data with full integration in the SAP HANA database technology. The data model is available in two different formats:

- **Conceptual data model (CDM)**
  The conceptual data model is the basis. It explains the business perspective independent of implementation and technology.

- **Physical data model (PDM)**
  The physical data model is derived from the conceptual model and it is dependent on the technology because it represents an SAP HANA-optimized physical implementation of the conceptual model. It is needed to create the SAP HANA definitions of the entities and to generate the tables in SAP HANA.

We recommend that you use **SAP PowerDesigner** as a graphical enterprise architecture and design solution to display, understand, or extend the delivered data model. For more information, see Deploying SAP Financial Services Data Management [page 75].

For more information about conceptual and physical data models in SAP PowerDesigner, see Getting Started with Data Modeling.

For more information about the data model of SAP Financial Services Data Management and its construction, see the Data Model section in the user guide for SAP Financial Services Data Management.

SAP HANA CDS files: software component FSDM CDS DATA MODEL (`FSDM_CDS_DM`)

The PDM enables you to produce SAP HANA Core Data Services (CDS) file entities that correspond to the PDM entities and that you can deploy. By building views on top of the generated views, you can create your individual reports.

Interface generator: software component FSDM INTERFACE GENERATOR (`XSAC_FSDM_IG`)

You can use the interface generator as a tool to generate interfaces for read and write access with bi-temporal versioning. The generated interfaces manage historical data by leveraging system-versioned tables and application-time period tables in SAP HANA. For more information, see the Versioning section in the user guide.

Template: software component FSDM DATA MODEL TEMPLATE (`XSAC_FSDM_TL`)

SAP Financial Services Data Management provides different SAP WebIDE templates in the form of different projects.

- The SAP WebIDE template project *Financial Services Data Management Project* contains the data model for SAP Financial Services Data Management. This means it contains the data model as delivered with the software components FSDM_PD_DM (CDM/PDM), and FSDM_CDS_DM (CDS files). We recommend that you use this template to install the data model because this facilitates the installation process and extension of the data model. For more information, see Installing or Upgrading the Template to Use or Extend the Data Model [page 59].
- The remaining projects provide **content for dedicated use cases**. You can use the delivered content views as an example to see how you can create your own views. For more information, see the Sample Content Guide for SAP Financial Services Data Management.

Extension

You can extend the data model with additional fields. See the following steps for a brief overview of the extension process:

1. **Extend CDM**
   Extend the conceptual data model (CDM) in SAP PowerDesigner to ensure that you keep lineage between the conceptual data model (CDM), the physical data model (PDM), and the SAP HANA CDS model that is deployed.

2. **Regenerate PDM**
   Regenerate the physical data model (PDM) in SAP PowerDesigner so that the new fields and entities appear there.
3. **Export CDS files**
   Based on the extended PDM, export the new CDS definition files from SAP PowerDesigner and deploy them on SAP HANA. Then regenerate the read and write interfaces to include the extensions in the interfaces.
   For more information, see Extensibility Information [page 100].

SAP HANA Database Technology

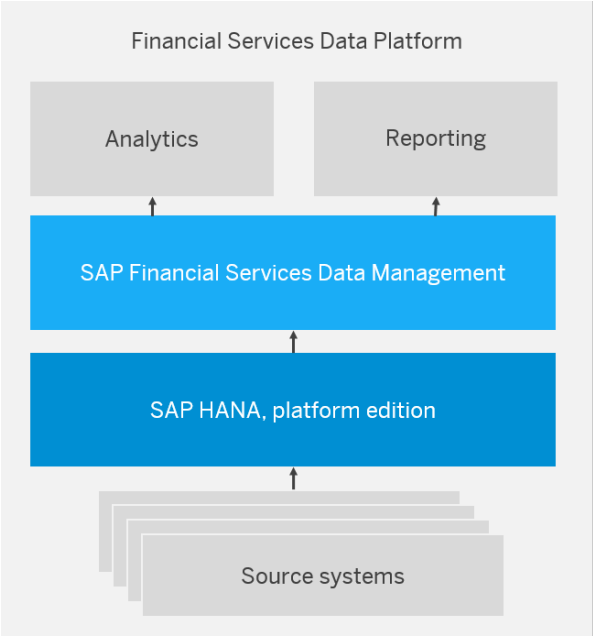
The database technology of SAP HANA, platform edition provides the basic data platform and database services using standardized interfaces to store and access your data in-memory and column-based. The SQL technology, facilitates development, maintenance, and integration. We recommend that you use the following compnents:

- SAP HANA extended application services, advanced model (SAP HANA XS advanced) is required as platform services for runtime environment.
- SAP HANA smart data integration is recommended for loading data from your operational source system into SAP HANA and from SAP HANA to your generated read and write access procedures of the interface generator.
- SAP Web IDE for SAP HANA is a web browser-based development environment for SAP HANA. We recommend that you use it to deploy the SAP HANA CDS views as well as the generated read and write interfaces on the SAP HANA database. SAP Web IDE for SAP HANA is mainly relevant for your own development area.
- SAP HANA data warehousing foundation contains basic components that you can use to build your individual data warehouse.

Data Platform

To build your individual financial services data platform, you can combine the above mentioned components with your operational systems as follows:

- Your individual **source systems**
- A basic database technology including services provided with **SAP HANA, platform edition**
- A basic framework for the format in which the data is stored, provided with the unified data model of **SAP Financial Services Data Management**
- Your individual **analytics solutions** and **reporting tools**



Reporting

More Information

- Deploying SAP Financial Services Data Management [page 75]
- Getting Started with Data Modeling
- Data Model
- Versioning
- Installing or Upgrading the Template to Use or Extend the Data Model [page 59]
- Sample Content Guide for SAP Financial Services Data Management
- Extensibility Information [page 100]

1.2 What You Need to Know About the System Landscape

An SAP HANA system contains one or more tenant databases and one system database. There are different administrative tasks performed at system level than there are at database level. Database clients such as the SAP HANA cockpit connect to specific databases. The SAP HANA XS advanced application server is a layer on top of SAP HANA that provides the platform for running SAP HANA-based Web applications. It is an integral part of the SAP HANA system.

For smaller installations (such as POCs), we recommend that you use only one or two SAP HANA systems instead of multiple systems. In this case, we recommend that you use separate tenant databases and XSA spaces.

System Landscape Recommendations

We recommend that for the system landscape for SAP Financial Services Data Management you set up three separate SAP HANA systems, all with XSA installed. In addition we recommend that you keep these systems separate since they have different requirements for sizing, availability, and scale-out.

Development System

Make sure that your development system has SAP Web IDE for SAP HANA and both FSDM XSA components installed. All developers can use this system to work on the data model and applications in SAP Web IDE for SAP HANA, as well as deploy and test locally without any risk to the data in the other systems. The coordination between different developers is ideally achieved using the version management system Git as described under Managing Your Data Model [page 12] below.

The system needs at least two different XSA spaces:

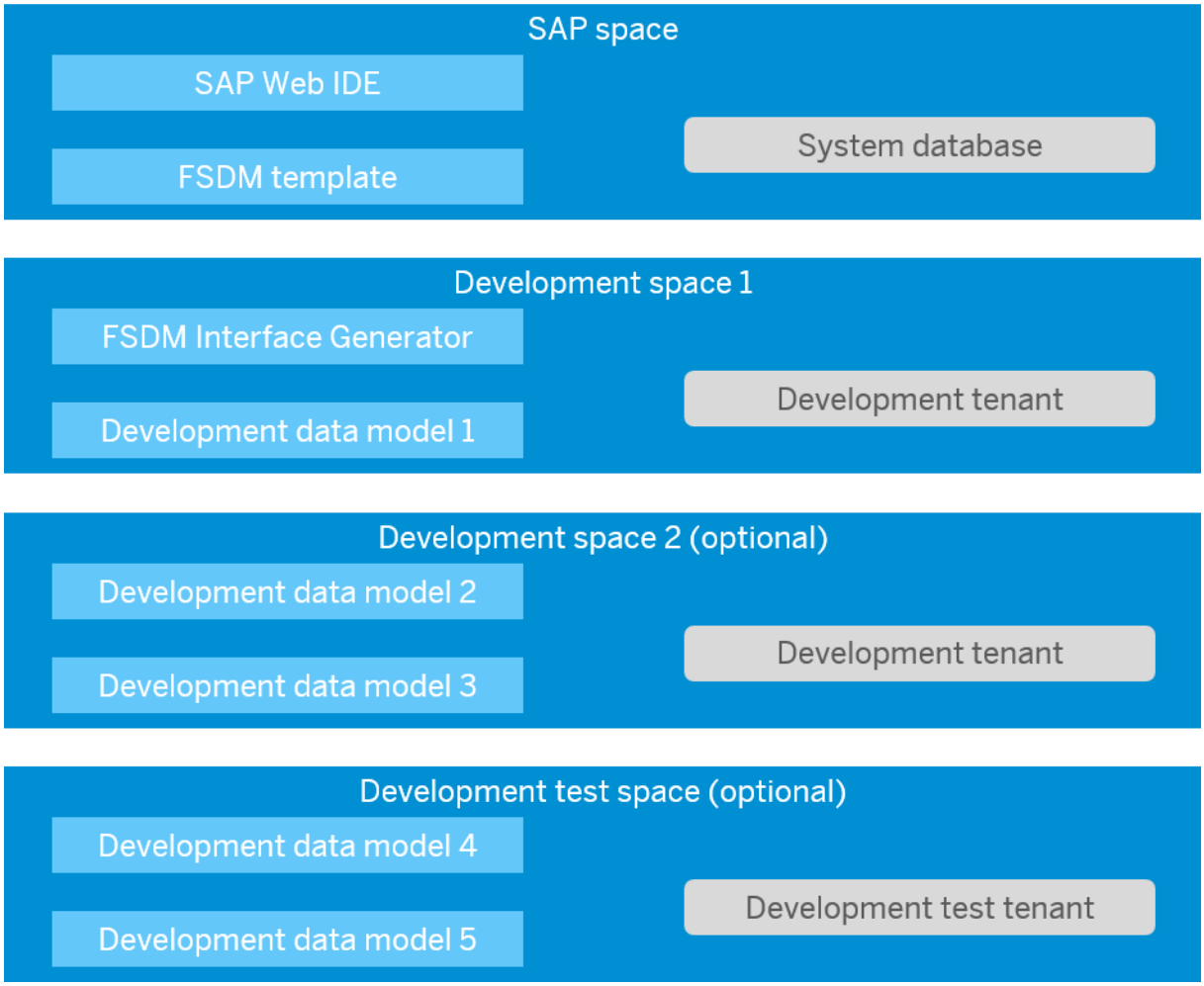
- One space for SAP Web IDE and the FSDM template plug-in
- One development space for the data model in SAP Web IDE
  We recommend the following:
  - Use a separate tenant database for the development space instead of reusing the system database.
  - Work with **small volumes of data**.
  - Developers need to use their own data model copy and synchronize the changes using Git.
  - During development testing, configure the system so that you get an alert when migrations are required.
    Use the following technical configuration:

    Sample Code

    ```
    {
    migration disabled;
    };
    ```

    For more information, see CDS Entity Syntax Options in XS Advanced.
  - If there are a lot of developers, create more than one development space. Multiple spaces are possible.
- Optional: create one space for development testing where you can test with a more stable version of the development artifacts. You can work with slightly more data here.

The following figure shows an example of possible spaces:



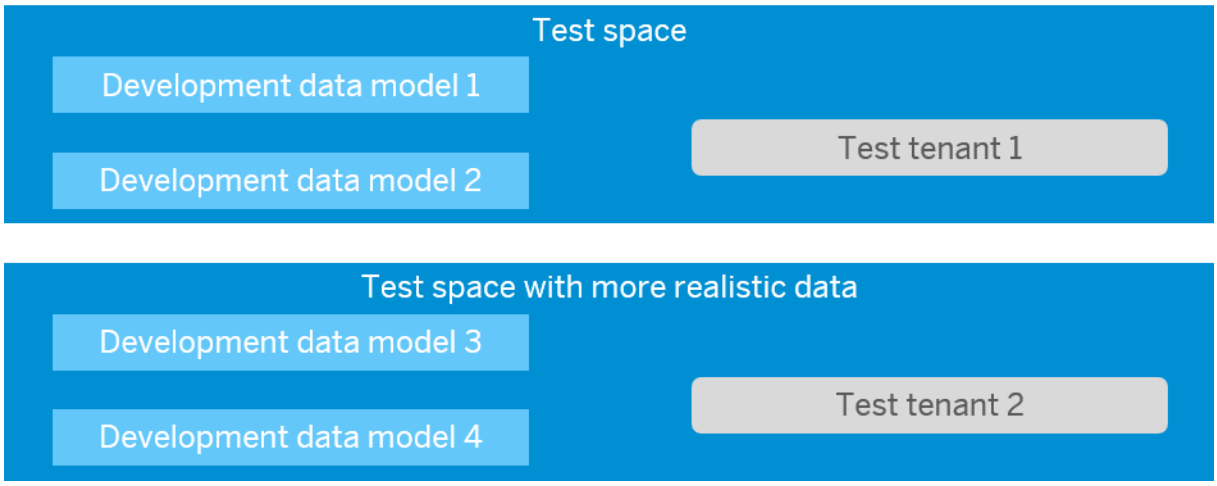
Your development system needs to be sized according to the XSA and SAP Web IDE recommendations depending on the number of developers using SAP Web IDE. See the following SAP Notes:

Location	Content
2618752	Resource consumption of SAP HANA extended application services, advanced model
2624901	SAP Web IDE for SAP HANA 2.0 SPS 03 - Sizing Note
2596466	FAQ: SAP HANA XS advanced Pay special attention to operation step number 21 (**How to configure the application working directory to improve application start performance?**)
2222250	In exceptional cases, it may be necessary to mix different workloads in one system (such as working in SAP Web IDE for SAP HANA and testing flowgraphs and views). Make sure you set adequate SAP HANA workload management parameters as described in this SAP Note.

Test System

The test system needs to have at least one tenant database where developers can test on a larger data set. In larger test spaces, we recommend that you test only with SQL users, deploy the data model, and view changes from development using continuous integration. You can have SAP Web IDE installed and work in one space where people can change views or the data model interactively.

As shown in the following figure, it may be useful to have one tenant database with synthetic data where developers have a lot of authorizations and one tenant database with a subset of the productive data that may be suitably anonymized.



Productive System

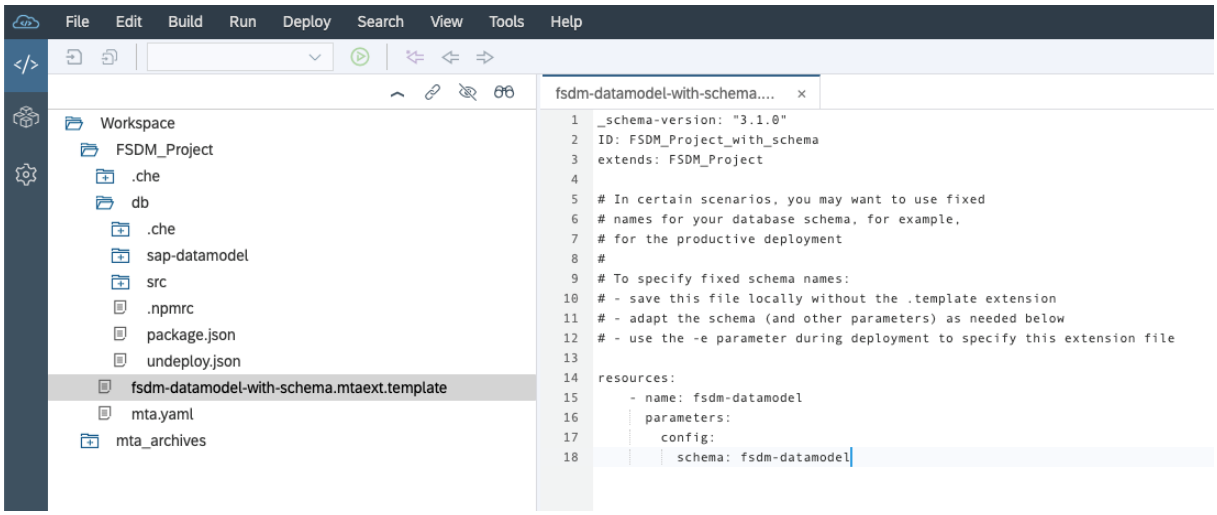
The productive system needs to be sized according to the amount of data you want to load and the number of reporting users that you expect. It may be useful to use scale-out or extension nodes on this system.

Using Named Database Schemas in Your Landscape

In development and test systems, we recommend not using explicit database schemas. Cross-container references are based on container names and container names are unique in each (XSA) space. This enables you to deploy a set of containers that are connected via cross-container references in different spaces on the same HANA tenant.

In a test or productive system, you may want to have one set of containers where some of the containers have explicitly named schemas. To do this, we recommend that you specify the schema in a MTA deployment extension file and using the `-e` option during deployment. For more information see The MTA Deployment Extension Descriptor.

As an example of how such an extension file may look, SAP Financial Services Data Management provides the template `fsdm-datamodel-with-schema.mtaext.template` as shown in the following figure:



Managing Your Data Model

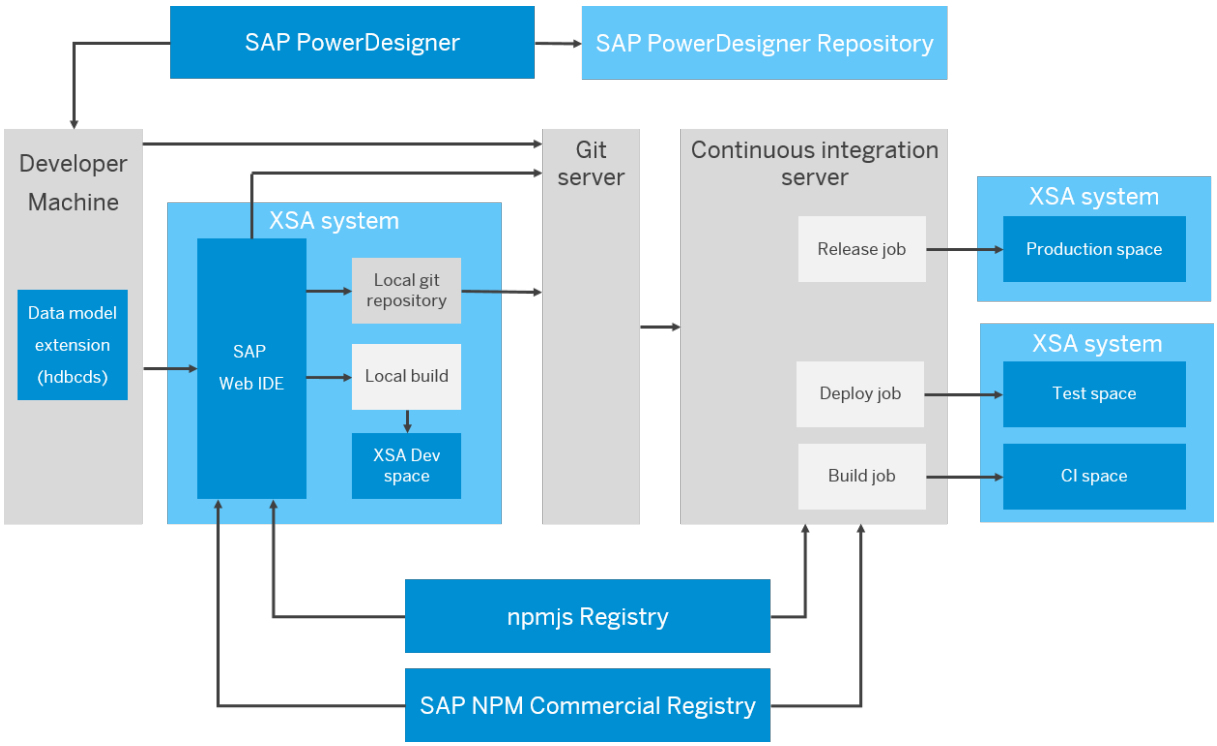
We recommend that you keep your data models in a Git repository. SAP HANA XS advanced supports Git as a source control management system. If you are already using a central Git repository for other developments, we recommend that you use this repository.

For more information, see Installing Repository and Continuous Integration Tools [page 38].

Distributing New Versions of the Data Model in Your Landscape

We recommend that you test and distribute your data model changes using a continuous integration (CI) approach. If you are developing your own XSA applications on top of the data model, you need to follow the same approach.

The setup of your continuous delivery process for SAP Financial Services Data Management may look as shown in the following figure. Note that this is just an example and that your setup may differ significantly depending on your needs.



In this example the set up consists of the following fragments:

- Your **SAP PowerDesigner** installation including the Repository. You can extend the CDM with additional fields, regenerate the PDM, and export the new CDS definition files (hdbcds) that includes your extensions.
- On your **developer machine** you can store your extended CDS definition files (hdbcds)
- **XSA development system** organization including:
  - **SAP Web IDE** for the template plug-in. Here you deploy your extended CDS definition files (hdbcds).
  - A **local git repository**
  - **Local build** processes
  - An **XSA development space**
- **Git server** for version control including a git repository
- **Continuous integration server**, for example a Jenkins server.
- We recommend that you use the **NPM registry** as described in Using the NPM Registry for Installing and Upgrading an Existing Data Model [page 66] to automate the necessary steps to deploy the data model and the generated write/read interface to SAP HANA XS advanced Runtimeproject.
- **XSA systems** for production, test, and CI space

Related Links

Location	Content
Continuous Integration and Delivery Best Practices Guide	Provides general information about how to set up a continuous delivery process for SAP HANA XS advanced model applications. Note the CI process for XS advanced development as shown in figure 1.
Installing Repository and Continuous Integration Tools [page 38]	Installation description within this guide

1.3 Important SAP Notes

This list directs you to further important information for the following topics:

SAP Note	Title	Description
2559644	Release Information Note (RIN) for SAP Financial Services Data Management	Provides central release information including updates for patches. Please check on a regular basis for updates.
2567713	Release Information Note (RIN) for FSDM PD DATA MODEL	Provides central release information for the FSDM data model including updates for patches. Please check on a regular basis for updates.
2872911	Guide: Domain Checks for Field Values	The attached document provides information about domain checks for field values in SAP Financial Services Data Management. It explains different approaches for checking field values against a domain which can be implemented in the ETL process.
2771618	FSDM: Definition of field values	The attachment provides all the field values for SAP Financial Services Data Management (no delta). Please check on a regular basis for updates.
2729620	Data Ageing of Database Tables in FSDM to Hadoop	The attached document provides information about how to perform data aging in SAP Financial Services Data Management by relocating data to Hadoop/SAP Vora using SAP Data Hub data pipelines or the SAP HANA DWF DLM tool.
2832082	Data Volume Management in SAP Financial Services Data Management Using SAP HANA Native Storage Extension	The attached document provides information about data volume management using SAP HANA native storage extension (NSE). It describes how to enable and use NSE for data tiering in SAP Financial Services Data Management.
2990730	Concepts for Partitioning in SAP Financial Services Data Management	The attached documents provide information about how you can manage your data volume with different partitioning concepts.
2832068	Where used list for impact analysis of artefacts in SAP Financial Services Data Management	The attached document provides information about use cases for a where-used list for the impact analysis of artifacts in SAP Financial Services Data Management.

SAP Note	Title	Description
2771449	Example for accessing the Task Orchestration Engine API	The attached document provides information about how to call the Task Orchestration Engine API of the Data Warehouse Foundation.
2785178	Guide : FSDM integration with SAP S/4HANA for financial products subledger for data replication	The attached document provides additional information about the integration process between SAP Financial Services Data Management and SAP S/4HANA for financial products subledger.
2868940	Field Length Restrictions for the Integration of FSDM with FPSL	The attached document provides information about field length restriction relevant for the integration with SAP S/4HANA for financial products subledger.
2770223	Partitioning in FSDM in line with FPSL - Synthetic Partitioning Approach	The attached document provides information about the partitioning in SAP Financial Services Data Management in line with SAP S/4HANA for financial products subledger using the synthetic partitioning approach. A partitioning strategy can be used to improve system performance and to overcome the restriction in SAP HANA that a non-partitioned table cannot store more than two billion records. If you have SAP Financial Services Data Management and SAP S/4HANA for financial products subledger in your landscape, we recommend that you use the synthetic partitioning approach in SAP Financial Services Data Management, which is similar to the approach used in SAP S/4HANA for financial products subledger.
2989584	FSDM 1.11: Versioning enabled for GeneralLedgerAccountBalance and SubledgerAccountbalance entity	

SAP Note	Title	Description
2837614	Guide : Data replication from SAP Loans Management for Banking, SAP S/4HANA (FS-CML) to SAP Financial Services Data Management	The attached document provides information about the integration with SAP Loans Management for Banking, SAP S/4HANA edition. It describes how to set up the inbound integration between SAP Loans Management for Banking, SAP S/4HANA edition and SAP Financial Services Data Management by implementing the extraction, transformation, and load (ETL) process for integration.
2931864	Guide: Back-to-Standard Migration for Data Model Extensions	The attached document provides information about what to do if you have already created your own extensions to the standard data model of SAP Financial Services Data Management and want to return to the standard data model.
3219581	Disabling System Versioning	The attached document provides information about what to do if you want to disable SAP HANA System Versioning for a set of entities or for all the entities in case there is no auditing or traceability requirement.

2 Security Information

Protecting corporate information is one of the most important topics for you. You need to meet ever increasing cyber-security challenges, keep your systems secure, and stay on top of the compliance and regulatory requirements of today's digital world. Since SAP Financial Services Data Management is based on SAP HANA, platform edition, this platform allows you to securely run and operate SAP HANA in a variety of environments and to implement your specific compliance, security, and regulatory requirements.

The high flexibility of SAP Financial Services Data Management means a wide range of options for your security settings. We recommend that you use different HDI containers to manage your security settings for different use cases. This allows you to control security settings for an entire HDI container, which makes the setup much easier and more efficient, and allows you to respond individually to different requirements. For more information, see the User Administration and Authorization [page 28] section of this admin guide.

Data Encryption

Please note that SAP Financial Services Data Management can collect the data only as you submit it. Data encryption is not the focus of the product. You must, therefore, guarantee the encryption and decryption of the data in your source and target systems and during the data loading process. You can use the data encryption services provided by SAP HANA, platform edition.

For example, to comply with the Payment Card Industry Data Security Standard (PCI DSS), you have to encrypt the credit card number. Here you can choose between the following options:

- You can control the data encryption in your source and target systems, and use the data in SAP Financial Services Data Management as delivered. This is the option we recommend.
- You can use the SAP HANA data encryption services on the server side. Since the entire database is encrypted in this option, this may have a negative impact on performance.
- You can use the client-side data encryption. Column-based data encryption is possible in this option, but in combination with multitenant areas, the complexity can increase considerably.

Related Links

Location	Content
SAP HANA Security Guide	Provides all information relating to the secure operation and configuration of SAP HANA, platform edition.
Server-Side Data Encryption Services	Section in the SAP HANA Security Guide that describes the SAP HANA features for encrypting data that protects data on operational system level.
Client-Side Data Encryption	Section in the SAP HANA Security Guide that describes the column-level data encryption capability managed by the client driver.

Location	Content
Recommendations for Data Encryption	Section in the SAP HANA Security Checklists and Recommendations.
pcisecuritystandards.org	PCI Security Standards Council

2.1 Data Protection and Privacy

Data protection is associated with numerous legal requirements and privacy concerns. In addition to compliance with general data protection and privacy acts, it is necessary to consider compliance with industry-specific legislation in different countries. SAP provides specific features and functions to support compliance with regard to relevant legal requirements, including data protection, which are documented in these templates along with the assumptions that have been guiding the implementation in the software. By nature of legal requirements the conclusion whether these features are covering customer specific demands as well as the conclusion whether additional measures have to be taken is solely with the customer.

> **Note**
>
> SAP does not provide legal advice in any form. SAP software supports data protection compliance by providing security features and specific data protection-relevant functions, such as simplified blocking and deletion of personal data. In many cases, compliance with applicable data protection and privacy laws will not be covered by a product feature. Definitions and other terms used in this document are not taken from a particular legal source.

> **Caution**
>
> The extent to which data protection is supported by technical means depends on secure system operation. Network security, security note implementation, adequate logging of system changes, and appropriate usage of the system are the basic technical requirements for compliance with data privacy legislation and other legislation.

SAP Financial Services Data Management is mainly designed to consolidate data from different source systems into one single source of truth. We assume that you have already taken care of the relevant legal requirements and privacy concerns within the operational systems that provide the data you are using.

Please be aware that because of the consolidatation of data from different source systems, a connection between the data might bring it into a new context. This may affect access permissions. In addition transferred data might not be up to date because changes during the import process or because the data is not configured to be transferred. Depending on your configuration, it might be possible that some information is not transferred at all. This may also affect information retrieval requests from data subjects.

A wise allocation of your data for dedicated use cases into different HDI containers allows you to individually control security settings for the entire container. A well-thought out distribution into HDI containers makes the set up of the following more efficient:

- SAP HANA authorization concepts to configure roles and authorizations to control access even to sensitive data

- Audit trails and audit policies for change logging to track read access directly in the SAP HANA database. This allows you, for example, to log and monitor read access to sensitive data.
- Your own purpose management, by using one single HDI container to store all data that is related to one dedicated purpose such as European regulatory reporting or subledger accounting.

More Information

- Data Protection in SAP HANA
- SAP HANA Data Anonymization

2.1.1 Glossary

Term	Definition
Artificial Intelligence (AI)	The simulation of human intelligence processes by machines and computer systems – typically by learning, coming to its own conclusions, appearing to understand complex content, engaging in natural dialogs with people, enhancing human cognitive performance (also known as cognitive computing) or replacing people on execution of nonroutine tasks. Applications include autonomous vehicles, automatic speech recognition and generation and detecting novel concepts and abstractions (useful for detecting potential new risks and aiding humans to quickly understand very large bodies of ever-changing information)
Automated Decision Making	The ability to make decisions by technological means without human involvement.
Blocking	A method of restricting access to data for which the primary business purpose has ended.
Business Purpose	The legal, contractual, or in other form justified reason for the processing of personal data to complete an end-to-end business process. The personal data used to complete the process is predefined in a purpose, which is defined by the data controller. The process must be defined before the personal data required to fulfill the purpose can be determined.
Consent	The action of the data subject confirming that the usage of his or her personal data shall be allowed for a given purpose. A consent functionality allows the storage of a consent record in relation to a specific purpose and shows if a data subject has granted, withdrawn, or denied consent.

Term	Definition
Data Subject	Any information relating to an identified or identifiable natural person ("data subject"). An identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier, or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural, or social identity of that natural person.
Deletion	Deletion of **personal data** so that the data is no longer available.
End of Business	Defines the end of active business and the start of residence time and retention period.
End of Purpose (EoP)	The point in time when the processing of a set of personal data is no longer required for the primary business purpose, for example, when a contract is fulfilled. After the EoP has been reached, the data is blocked and can only be accessed by users with special authorizations (for example, tax auditors).
End of Purpose (EoP) check	A method of identifying the point in time for a data set when the processing of **personal data** is no longer required for the primary **business purpose**. After the **EoP** has been reached, the data is **blocked** and can only be accessed by users with special authorization, for example, tax auditors.
Personal data	Any information relating to an identified or identifiable natural person ("data subject"). An identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier, or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural, or social identity of that natural person.
Processing of Personal Data	Processing means any operation or set of operations which is performed on personal data or on sets of personal data, whether or not by automated means, such as collection, recording, organisation, structuring, storage, adaptation or alteration, retrieval, consultation, use, disclosure by transmission, dissemination or otherwise making available, alignment or combination, restriction, erasure or destruction.
Purpose	The information that specifies the reason and the goal for the processing of a specific set of personal data. As a rule, the purpose references the relevant legal basis for the processing of personal data.

Term	Definition
Residence period	The period of time between the end of business and the end of purpose (EoP) for a data set during which the data remains in the database and can be used in case of subsequent processes related to the original purpose. At the end of the longest configured residence period, the data is blocked or deleted. The residence period is part of the overall retention period.
Retention period	The period of time between the end of the last business activity involving a specific object (for example, a business partner) and the deletion of the corresponding data, subject to applicable laws. The retention period is a combination of the residence period and the blocking period.
Sensitive personal data	A category of personal data that usually includes the following type of information: • Special categories of personal data, such as data revealing racial or ethnic origin, political opinions, religious or philosophical beliefs, trade union membership, genetic data, biometric data, data concerning health or sex life or sexual orientation. • Personal data subject to professional secrecy • Personal data relating to criminal or administrative offenses • Personal data concerning insurances and bank or credit card accounts
Technical and organizational measures (TOM)	Some basic requirements that support data protection and privacy are often referred to as technical and organizational measures (TOM). The following topics are related to data protection and privacy and require appropriate TOMs, for example: • **Access control:** Authentication features • **Authorizations:** Authorization concept • **Read access logging** • **Transmission control / Communication security** • **Input control / Change logging** • **Availability control** • **Separation by purpose:** Is subject to the organizational model implemented and must be applied as part of the authorization concept.

2.1.2 User Consent

Any personal data collected or processed must be linked to a specific, pre-defined purpose, such as the fulfilment of a contract or legal obligation. If there is no other legal basis for the lawful processing of personal data or if the data is to be sent to a third party, you must obtain consent from the data subject to use their personal data. SAP Financial Services Data Management uses data from operational source systems and does not collect any data directly. The assumption for SAP Financial Services Data Management is that all imported data is covered by the contractual purposes of the operational source systems.

2.1.3 Read Access Logging

Read Access Logging is considered as an additional safeguard in the protection of personal data, because it helps to identify potential illegitimate access to personal data. Read access to sensitive personal data is partially based on legislation, and it is subject to logging functionality. Read access logging (RAL) is used to monitor and log read access to sensitive personal data that was disclosed via user interface, which can be extended to read access to other personal data. Data may be categorized as sensitive by law, by external company policy, or by internal company policy. When these read accesses are logged, you should be able check which user accessed personal data on which access channel and the date and time, depending on the configuration. Read access logging enables you to answer questions about who accessed particular data within a specified time frame. That logging also includes downloading attachments or files, logs for such events shall contain information to identify the attachment. Additionally, as for Read Access Logging across system boundaries, the respective “access” shall be logged as soon as sensitive personal data crosses the boundary from a trusted to an un-trusted area. Here are some examples of such questions:

- Who accessed the data of a given business entity, for example a bank account?
- Who accessed personal data, for example of a business partner?
- Which employee accessed personal information, for example religion?
- Which accounts or business partners were accessed by which users?

Furthermore, log records can be viewed and queried, but access to them is restricted by adequate authorizations. The personal data for which read access shall be logged and the retention period of logs, can be configured.

Read Access Logging in SAP Financial Services Data Management

To track read access directly in the SAP HANA database of SAP Financial Services Data Management, we recommend that you use the audit trails and audit policies for read access logging. You can setup the audit trail and audit policies on HDI container level. For more information, see the Auditing Activity section in the SAP HANA Security Guide.

More Information

- Auditing Activity in the SAP HANA Database
- Audit Trails

2.1.4 Information Report

Data subjects have the right to get information regarding their personal data undergoing processing. In SAP Financial Services Data Management it is possible to display all the information stored about a certain data subject.

Since the operational data is replicated from operational source systems, SAP Financial Services Data Management assumes that information reporting is implemented in these operational source systems.

If you need information reporting directly for the SAP HANA database of SAP Financial Services Data Management, you can use SQL to retrieve the required information. You can use SAP WebIDE / SAP HANA runtime tools to create views for the required database tables to fetch the information.

2.1.5 Deletion of Personal Data

Simplified Blocking and Deletion

The processing of personal data is subject to applicable laws related to the deletion of this data when the specified, explicit, and legitimate purpose for processing this personal data has expired. If there is no longer a legitimate purpose, that requires the retention and use of personal data, it must be deleted irrecoverably. Blocking is necessary when the original retention period has expired but additional applicable extended and overruling (mandated by law) retention periods are still in place. After the expiration of the longest retention period, the data must be deleted.

Deletion of Personal Data

When considering compliance with data protection regulations, it is also necessary to consider compliance with industry-specific legislation in different countries. A typical potential scenario in certain countries is that personal data shall be deleted after the specified, explicit, and legitimate purpose for the processing of personal data has ended, but only if no other retention periods are defined in legislation, for example, retention periods for financial documents. Legal requirements in certain scenarios or countries also often require blocking of data in cases where the specified, explicit, and legitimate purposes for the processing of this data have ended, however, the data still has to be retained in the database due to other legally mandated retention periods. Sometimes, transactional data are personal data with relation to a master data object, e.g.

a sales order with reference to a business partner. Therefore, the challenge for deletion and blocking is first to handle transactional data and finally other data, such as business partner data.

SAP HANA supports the deletion of data in tables using SQL deletion commands. Applications running on SAP HANA must make use of such commands to implement deletion requirements of personal data. For more information see the Data Protection and Privacy section in the SAP HANA security guide.

Data Loading and Deletion in SAP Financial Services Data Management

SAP Financial Services Data Management relies on the source system to provide data. It requires a replication infrastructure that you need to set up to provide data from the source system. The infrastructure consists of a set of tools offered by the SAP HANA platform, such as SAP Smart Data Integration flowgraphs. They can be used to load data from various source systems to the SAP HANA database. The interface generator provides procedures for loading and deleting data. These procedures can be used by the replication infrastructure that you set up for loading or deleting data.

The retention policy maintained in the source system is also applicable to data in SAP Financial Services Data Management. After the completion of the retention period when the data is deleted in the source system, this information has to be provided to SAP Financial Services Data Management by the source system using the replication infrastructure. The delete procedures provided by the interface generator can be used by the replication infrastructure (such as the ETL process) to delete the data from SAP Financial Services Data Management.

> **Note**
>
> You need to make sure that the replication infrastructure properly supports the update and deletion of data in accordance with data protection and privacy, see also Data Provisioning, Loading, and Reporting [page 93] in this administrator's guide.

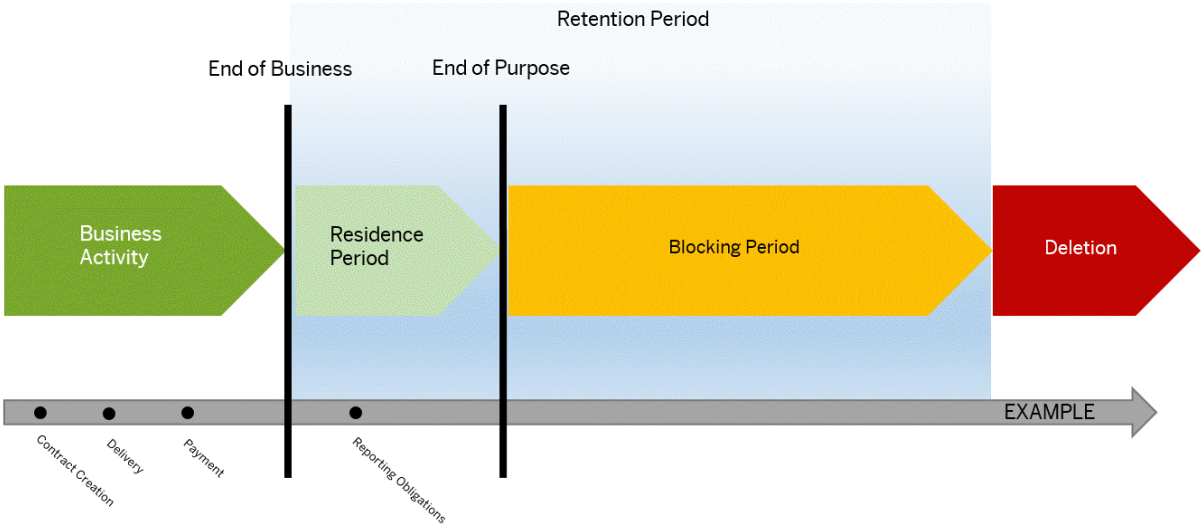
If data stored in SAP Financial Services Data Management reaches the end of its residence period or retention period and is used as content for products outside of SAP Financial Services Data Management, the system does not provide any notification service about the end of purpose, end of retention, or deletion. The product outside must ensure that the legal requirement of deletion and blocking is fulfilled.

End-of-Purpose Check

An end of purpose (EoP) check determines whether data is still relevant for business activities based on the retention period defined for the data. The retention period is part of the overall lifecycle of personal data which consists of the following phases:

- **Business activity**: The relevant data is used in ongoing business, for example contract creation, delivery or payment.
- **Residence period**: The relevant data remains in the database and can be used in case of subsequent processes related to the original purpose, for example reporting obligations.
- **Blocking period**: The relevant data needs to be retained for legal reasons. During the blocking period, business users of SAP applications are prevented from displaying and using this data; it can only be processed in case of mandatory legal provisions.

- **Deletion**: The data is deleted and no longer exists in the database.



Blocking

The blocking policy maintained in the source system is also applicable to data in SAP Financial Services Data Management.

Blocking of data can impact system behavior in the following ways:

- **Display**: The system does not display blocked data.
- **Change**: It is not possible to change a business object that contains blocked data.
- **Create**: It is not possible to create a business object that contains blocked data
- **Copy/Follow-Up**: It is not possible to copy a business object or perform follow-up activities for a business object that contains blocked data.
- **Search**: It is not possible to search for blocked data or to search for a business object using blocked data in the search criteria.

Prevent Access to an Object Which Has Reached the End of Purpose

If you are using the SAP Financial Services Data Management interface generator, the following artifacts are relevant for the generated read interface to facilitate the authorization checks that prevent access to a objects (like business partner (BP) and financial contract) which has reached the end of purpose (EoP):

Artifact	Description	Technical Information
Generated: view related to the objects, like business partner and financial contract	Projects the `IsEndOfPurposeBlocked Reached` indicator from the operational source system.	• View name `DataProtectionAndPrivacyDetails_<objectName>_ViewWithCheck` • Using the SQL-based analytic privilege syntax… `with structured privilege check` Example: • Business Partner: `DataProtectionAndPrivacyDetails_ViewWithCheck` • Financial Contract: `DataProtectionAndPrivacyDetails_FinancialContract_ViewWithCheck`
Generated: two structured privileges per object	They refer to the `IsEndOfPurposeBlocked Reached` indicator.	• To allow administrators to access blocked business partners: `DataProtectionAndPrivacyDetails_<Object>_Blocking` • To provide regular users with access only to business partners that are not blocked: `DataProtectionAndPrivacyDetails_<Object>_NonBlocking` Example: • Business Partner: `DataProtectionAndPrivacyDetails_Blocking` and `DataProtectionAndPrivacyDetails_NonBlocking` • Financial Contract: `DataProtectionAndPrivacyDetails_FinancialContract_Blocking` and `DataProtectionAndPrivacyDetails_FinancialContract_NonBlocking`

Artifact	Description	Technical Information
Generated: two roles per object	When you deploy the read interface, you create roles for required objects: For accessing business partner data for example: • Role 1: `Only BP with IsEndOfPurposeBlocked = TRUE` • Role 2: `Only BP with IsEndOfPurposeBlocked = FALSE`	You can access the roles in SAP HANA Studio by choosing *Security* > *Roles*, right-click *Roles*, choose *Find Role* and search for a term with the prefix *DataProtectionAndPrivacyDetails*.
Manual step: create joined view	You can join the generated view related to the required object to other views that have personal data. This join would now ensure that the necessary privilege is applied to it.	You can use: • SQL join statements • SAP HANA graphical calculation views
Manual step: assign generated roles to users	You can then assign the generated roles to users depending on their need to access blocked objects. Example for business partner: If access is required, you need to assign the roles as follows: • For regular users: only assign the role that allows access to business partners that are not blocked. • For administrators: both roles need to be assigned so that they can access blocked business partners as well.	

More Information

- Examples: Securing Views Using SQL-Based Analytic Privileges
- CDS Access Policies in XS Advanced

2.1.6 Change Log

Creation and change of personal data need to be documented. Therefore, for review purposes or as a result of legal regulations, it may be necessary to track the changes made to this data. When these changes are logged,

you should be able to check which user made which change, the date and time, the previous value, and the current value, depending on the configuration. Furthermore, log records can be viewed and queried, but access to them is restricted by adequate authorizations. The personal data for which changes shall be logged and the retention period of logs, can be configured..

Change Log in SAP Financial Services Data Management

SAP Financial Services Data Management deploys a data model. The operational data is replicated from operational source systems. SAP Financial Services Data Management assumes that change logging is implemented in these operational source systems. In addition we recommend that you track changes directly in the SAP HANA database of SAP Financial Services Data Management. You have the following options:

- To support the SQL:2011 standard, you can use most database tables in the data model with system time versioning. Application-time (business time) versioning is also an option, as well as a combination of both, which is called bitemporal versioning. For more information, see the Versioning section in the user guide.
- You can use the audit trails and audit policies for change logging. We recommend that you set up the audit trail and audit policies on HDI container level. For more information, see the Read Access Logging [page 22] section of this administrator's guide or the Auditing Activity section in the SAP HANA Security Guide.

More Information

- Read Access Logging [page 22]
- Auditing Activity in the SAP HANA Database
- SAP HANA Audit Trail

2.2 User Administration and Authorization

Since SAP Financial Services Data Management is based on the platform edition of SAP HANA, the authorization concept for SAP HANA, platform edition as described in the SAP HANA Administration Guide is in principle valid.

The following user types are relevant:

XSA Users

To manage XSA users, we recommend that you use the XS Advanced Cockpit or the xs command line **xs create-user**. By default, XSA uses SAP HANA as an identity provider. For more information, see XS Advanced User Management.

If you are using the SAP Financial Services Data Management interface generator, you need to take additional authorization aspects into account as described in Installing or Upgrading the Interface Generator [page 68].

Individual developers usually have XSA users in the development system to use SAP Web IDE and are assigned to SpaceDeveloper roles in one or multiple spaces. For more information about the WebIDE Developer role collection, see User Authorization and Authentication in the SAP HANA Security Guide. In test and productive systems, you would typically use technical XSA users for deployment from a CI/CD tool as described in the section Operations Information [page 71].

As described under What You Need to Know About the System Landscape [page 8], we recommend that you install SAP Web IDE for SAP HANA in your development system. Then the developers should have the relevant authorizations. You can have several spaces and assign authorizations to different developers in different spaces.

SQL Users

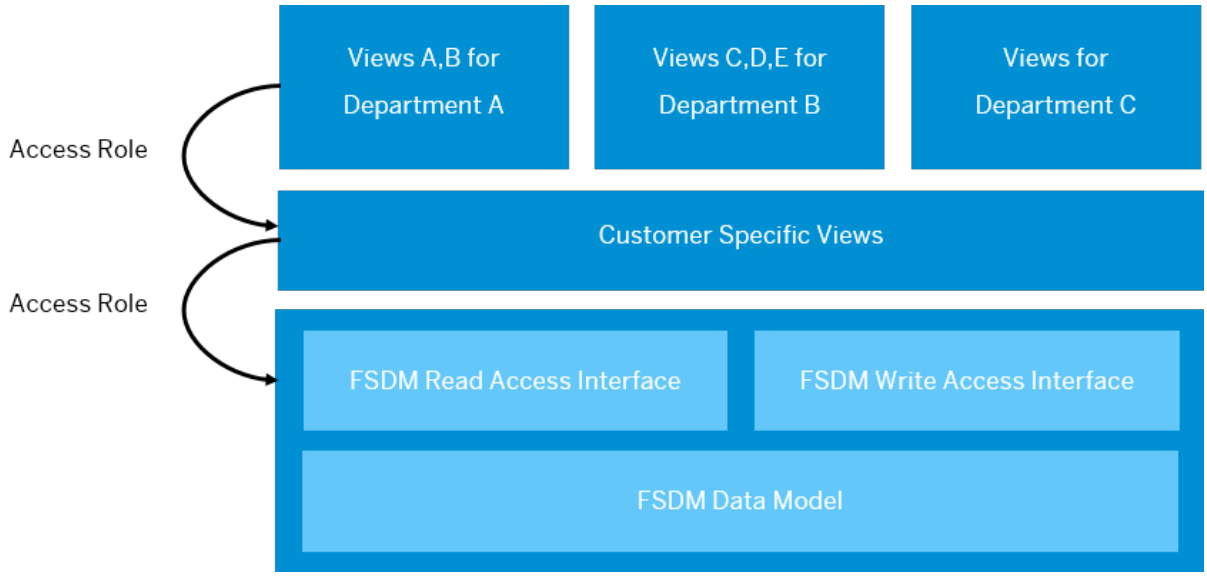
SQL users in tenant databases are used for reporting. You can manage these users using SAP HANA Cockpit.

For test and productive systems, see the sections Assign Roles to a Database User and Create a Database User in the SAP HANA administration guide. As an alternative, you can use provided roles for HDI containers as described under Grant a User a Role from the Container's Schema to grant a user access to the development system.

When you model SQL artifacts for read access, we recommend that you only use objects in SQL DEFINER mode. This has the following advantages:

- You do not need to grant SELECT privileges for the underlying tables. Instead, you can use a view to determine what a user is allowed to display. For example, you can use the view to restrict access to special table columns or to display the data only for certain regions.
- You can use separate containers with separate views if you have groups of users with different access privileges. This makes it easy to understand the privileges by looking at the artifacts in the container.

The following image illustrates how the access control using views works:



More Information

- 4. Installing or Upgrading SAP Web IDE for SAP HANA (required for development) [page 36]

3 Installation and Upgrade Information

The following sections provide an overview of the required and recommended software for SAP Financial Services Data Management. It is first necessary to install SAP HANA, platform edition and SAP HANA extended application services, advanced model. Some recommended components facilitate the installation steps for SAP Financial Services Data Management. Therefore, please follow the order of the installation sections in this guide. Where applicable, you can find the upgrade information included in the installation section of the corresponding component.

For the latest information on patch level, see SAP Release Information Note (RIN) 2559644.

Software Versions

We recommend that you always use the latest version of SAP Financial Services Data Management, as well as the latest version of required and recommended additional software. If you do not follow our recommendation, please note that you may encounter difficulties with the combination of software versions. Below you can find the list of software versions that we used for our internal tests:

Component	Version	Usage	Requirements and Recommendations
SAP HANA, platform edition	2.0, SPS07 REV 77 (or higher)	Basic technology platform For more information, see 1. Installing or Upgrading SAP HANA, platform edition (required) [page 34]	**Required**
SAP HANA extended application services, advanced model		Platform services for runtime environment For more information, see 2. Installing or Upgrading SAP HANA XS advanced (required) [page 34]	**Required**

Component	Version	Usage	Requirements and Recommendations
SAP HANA smart data integration		Loads data, in batch or real-time, from your operational source system into SAP HANA and from SAP HANA to the generated read and write access procedures of the interface generator For more information, see 3. Installing or Upgrading SAP HANA smart data integration (recommended) [page 36]	Recommended
SAP Web IDE for SAP HANA	SAP Web IDE for HANA 2 SPS 06 Patch 5, Build 4.6.6 (or higher)	Web browser-based development environment for SAP HANA For more information, see 4. Installing or Upgrading SAP Web IDE for SAP HANA (required for development) [page 36]	**Required for development**
SAP HANA data warehousing foundation	2.0 SP 05 (or higher)	Contains basic components for building a data warehouse For more information, see 5. Installing SAP HANA data warehousing foundation (recommended) [page 37]	Optional
A central Git repository and other continuous integration tools	-	Used as a front-end tool for versioning For more information, see Installing Repository and Continuous Integration Tools [page 38]	Recommended
SAP PowerDesigner	16.7 SP05 PL00	Used as a front-end tool for your conceptual and physical data models For more information, see Installing or Upgrading SAP PowerDesigner [page 39]	Recommended

Component	Version	Usage	Requirements and Recommendations
SAP Financial Services Data Management	1.23	Add-on for SAP HANA, platform edition For more information, see Installing and Upgrading SAP Financial Services Data Management [page 56]	**Required**
Additional front-end and reporting tools such as SAP BusinessObjects Analysis for Microsoft Office	2.5 SP1 (or higher)	Used for reporting purposes For more information, see Installing Reporting Tools [page 70]	Recommended

More Information

For more information about availability matrix, release strategy, upgrade strategy, and maintenance strategy, see the SAP Support Portal.

3.1 Installing or Upgrading SAP HANA Components

SAP Financial Services Data Management is based on SAP HANA platform technology. Therefore, the installation of SAP HANA, platform edition is a basic requirement.

Versioning and Upgrade

We recommend that you always use the latest version of SAP HANA, platform edition. Please always check the version of linked references, since all the links to SAP HANA references below lead you to the latest version.

Before you upgrade to the latest version of SAP Financial Services Data Management, make sure you have upgraded step-by-step to the latest version before it, and have done the same with your SAP HANA components, ensuring that you upgrade each SAP HANA version before each version of SAP Financial Services Data Management.

1. Installing or Upgrading SAP HANA, platform edition (required)

The installation of SAP HANA comprises several steps. The central part is the installation of the SAP HANA, platform edition. You need to check specific installation information, depending on the data provisioning technology you use and the other components you want to add to your SAP HANA landscape.

> → Recommendation
>
> When you install SAP HANA, platform edition, the system offers to install SAP HANA XS advanced in one step. We recommend that you do so including XSA content (XSA_CONT) since this procedure is much easier than installing each XSA component step-by-step. To upgrade, proceed in the same way and see SAP Note 2422421.

Related Links

For detailed installation or upgrade steps for SAP HANA, see the related links below.

Location	Content
SAP HANA Platform	SAP Help Portal documentation for SAP HANA, platform edition
SAP HANA Server Installation and Update Guide	Describes how to install or update an SAP HANA system and its components with the SAP HANA database lifecycle manager (HDBLCM)
Software Download	Describes how to find the installation media and components for SAP HANA.
SAP HANA Administration Guide	Central operations documentation for the deployment of the SAP HANA platform.
User Administration and Authorization [page 28]	Describes how to use the authorization concept for SAP HANA, platform edition for SAP FSDM
3426637 RIN for SAP HANA Platform 2.0 SPS07, Revision 077.00 3379631 - RIN for SAP HANA Platform 2.0 SPS07, Revision 074.00 3221209 RIN for Revision 59.05 3093250 RIN for revision 58 2844322 RIN for SAP HANA Platform 2.0 SPS05	Release Information Notes (RIN) for the important revisions
SAP Note 2456225	Describes how to include SAP HANA EPM-MDS (EPMMDS plug-in) to improve the update process for SAP HANA components.

2. Installing or Upgrading SAP HANA XS advanced (required)

SAP HANA extended application services, advanced model, (XS advanced) provide a comprehensive platform for the development and execution of native data-intensive applications. SAP HANA functions as

a comprehensive platform for the development and execution of native data-intensive applications that run efficiently in SAP HANA, taking advantage of its in-memory architecture and parallel execution capabilities. Structured accordingly, applications can profit from the increased performance provided by SAP HANA due to the integration with the data source.

The installation of SAP HANA extended application services, advanced model 1.0.152 (or higher) is a required installation step for SAP Financial Services Data Management. Make sure that you have installed SAP HANA XS advanced (XSA) runtime including content (XSA_CONT), as recommended above, during the installation of SAP HANA, platform edition.

The SAP HANA extended application services, advanced model run-time platform provides a number of services for managing the various container instances and their application run times. Containers are used to manage run times and allow for isolation, resource management, and shared service injection. The XS advanced application run times are lightweight processes that are invoked over HTTP and communicate remotely with the database. Execution environments such as JavaScript, including Node.js, and Java are supported. Node.js Data Services (node-cds) provide a native JavaScript client for using Core Data Services functionality in the JavaScript (node.js) run time container on the XS advanced platform. Make sure to perform the optional installation of Node 6 since it is not installed by default.

Related Links

Location	Content
Continuous Integration and Delivery Best Practices Guide	Describes best practices for continuous integration (CI) with SAP HANA XS advanced on SAP HANA
SAP HANA Developer Guide for SAP HANA XS Advanced Model	Explains how to build applications using SAP HANA, including how to model persistent and analytic data, how to write procedures, and how to build application logic in SAP HANA XS advanced.
2347931	Explains the relation between SAP HANA and release versions of the SAP HANA XS advanced components.
2939461	Central note for the latest release of SAP HANA XS advanced.
3250940	Explains upgrade steps for EXTENDED APPLICATION SERVICES 1 Release Collection 1.0.152
Maintaining the SAP HANA XS Advanced Model Run Time	Describes installation steps for SAP HANA XS advanced within the SAP HANA Server Installation and Update Guide.
2711421	Provides additional information regarding the installation of SAP HANA XS advanced using the XS advanced installation media.
Installing or Updating SAP HANA Components	Provides additional information about the installation and update process of SAP HANA Components.
2422421	Provides a guideline for upgrading a SAP HANA system with extended application services, advanced model.
Update XS Advanced Components	Describes the patch process for SAP HANA XS advanced
Proxy Server Settings for SAP HANA Installations with XS Advanced Runtime	Provides the values of environment variables required for an SAP HANA system with SAP HANA XS advanced installed, that is using a proxy server.

Location	Content
The SAP HANA XS Advanced JavaScript Run Time	Privides information about the business logic to retrieve data from the database using run time envronment (JavaScript, node.js).

3. Installing or Upgrading SAP HANA smart data integration (recommended)

SAP HANA smart data integration is used for SAP Financial Services Data Management as the recommended tool to load data, in batch or real-time, into SAP HANA from a variety of sources. SAP Financial Services Data Management uses only smart data integration and not smart data quality for the following:

- Data Provisioning Agent (DP Agent)
- Data Provisioning delivery unit (DP DU)
- Enterprise Semantic Services (ESS)
- SDI REST API

Related Links

Location	Content
Modeling Guide for SAP HANA Smart Data Integration and SAP HANA Smart Data Quality	Modeling guide for SAP HANA smart data integration
2091095	Central note for the latest release of SAP HANA smart data integration

4. Installing or Upgrading SAP Web IDE for SAP HANA (required for development)

SAP Web IDE for SAP HANA is a browser-based integrated development environment (IDE) for the development of SAP HANA-based applications comprised of web-based business logic, and extensive SAP HANA data models. SAP Web IDE for SAP HANA supports developers who use SAP HANA XS advanced, by providing a variety of tools, such as syntax-aware editors for code and SAP HANA artifacts, graphical editors for CDS data models and calculation views, as well as inspection, testing and debugging tools.

The installation of or upgrade to SAP Web IDE for SAP HANA is recommended for SAP Financial Services Data Management. The Web IDE plug-ins are needed only in your development system. The following components are included:

- `XSAC_SAP_WEB_IDE`
- `XSAC_HRTT`

We recommend that you use the latest version (SAP Web IDE for HANA 2 SPS 06 Patch 5, Build 4.6.6 or higher). For detailed installation or upgrade steps for SAP Web IDE for SAP HANA, see the related links below. Make sure you take the post-installation steps into account (such as enabling access).

Related Links

Content	Location
SAP Web IDE for SAP HANA	SAP Help Portal documentation for SAP Web IDE for SAP HANA
What You Need to Know About the System Landscape [page 8]	Describes the system landscape for SAP Financial Services Data Management.
3104090	SAP Web IDE for SAP HANA 2.0 SPS 06 - Central Release Note
Software Installation Matrix	Provides a list of the compatible versions and installation information of all the components required to set up and smoothly run your system.
SAP Web IDE for SAP HANA - Installation and Upgrade Guide	Provides the installation, post-installation and upgrade instructions for SAP Web IDE for SAP HANA
Access the SAP Web IDE Administration and Development Tools	Explains the browser access to the SAP Web IDE for SAP HANA and its administration tools.
Customizing the Environment	Describes how to maintain the settings to access npm according to customize your SAP Web IDE for SAP HANA Environment
2624901	Sizing note for SAP Web IDE for SAP HANA
2618752	Sizing: resource consumption of SAP HANA XS advanced
2596466	FAQ for SAP HANA XS advanced

5. Installing SAP HANA data warehousing foundation (recommended)

The installation of SAP HANA data warehousing foundation is recommended for building a data warehouse in SAP Financial Services Data Management.

We recommend that you use the latest version (2.0 SP 05 or higher). For detailed installation steps for SAP HANA data warehousing foundation, see the related links below.

Related Links

Location	Content
SAP HANA Data Warehousing Foundation	SAP Help Portal documentation for SAP HANA data warehousing foundation
SAP HANA Data Warehousing Foundation Installation Guide for XS Advanced Components	Describes how to install SAP HANA data warehousing foundation XSA tools, such as Data Warehouse Scheduler. These tools are SAP HANA-based applications on SAP HANA XS advanced.
2435452	Central note for the latest release of SAP HANA data warehousing foundation

3.2 Installing Repository and Continuous Integration Tools

Repository

SAP Web IDE for SAP HANA includes the Git source control system, letting you connect and interact with remote Git repositories. This is fully supported by SAP HANA XS advanced.

We recommend that you use a Git source control system for versioning purposes in your data model. You have the following options:

- If you are already using a central Git repository for other developments, we recommend that you use this repository.
- If you do not already have a Git server in your landscape, check the best practices description for SAP HANA XSA for information about how to set this up. You can also use Gerrit Version Control included as an optional component of SAP HANA XS advanced.
- If you set up a Jenkins pipeline after Git check-in, you can automate the necessary steps to deploy the data model and the generated write/read interface to SAP HANA XS advanced Runtimeproject.

Related Links

Location	Content
Continuous Integration and Delivery Best Practices Guide	Provides setup instructions for all relevant components.
Using Source Control (Git)	Describes how to use Git in SAP Web IDE for SAP HANA.
Setting Up Git	Describes how to set up Git.
Working with Gerrit Version Control in XS Advanced	Describes how to set up Gerrit version control in SAP HANA XS advanced

Continuous Integration (CI)

To distribute the changes tracked in your version control system of your system landscape, we recommend that you set up a continuous integration process. For more information, see the related links below.

Related Links

Location	Content
Continuous Integration and Delivery Best Practices Guide	Provides set up instructions for all relevant components.
Distributing New Versions of the Data Model in Your Landscape [page 12]	Provides an example how the set up of a continuous delivery process for SAP Financial Services Data Management may look like.

Location	Content
Continuous Delivery for the SAP Ecosystem	You can use the continuous delivery for your Jenkins pipeline.

3.3 Installing or Upgrading SAP PowerDesigner

We recommend that you use SAP Financial Services Data Management to display, understand, or extend the delivered data models. For more information about the use of SAP PowerDesigner, see the Operations Information [page 71].

Recommended version

We recommend that you always use the latest version of SAP PowerDesigner. Please always check the version of linked references, since all the links to SAP PowerDesigner references below lead you to the latest version.

Installation and Upgrade Steps

Before you install SAP Financial Services Data Management or upgrade to the latest version, make sure to install SAP PowerDesigner or upgrade as required. For detailed installation or upgrade steps for SAP PowerDesigner, see the related links below.

You can install SAP PowerDesigner directly on individual client machines, or a network server. We recommend that you use a local installation on the administrator's workstation for testing and experimentation. Use the SAP PowerDesigner repository to store your data model for SAP Financial Services Data Management as described under Installing the Repository and Installing the Repository to the Server. An administrator must set up the repository, which is stored in a database on a server.

Related Links

Location	Content
Planning Your PowerDesigner Installation	Installation of SAP PowerDesigner Please always check the version of this reference, since the link leads you to the latest version. There may be deviations if you are using an older version.
2616428	SAP PowerDesigner Central SAP Note
3122475	SAP PowerDesigner v16.7 SP05 Release Note
2908504	SAP PowerDesigner v16.7 Release Note

Location	Content
2567713	Release information note (RIN) FSDM PD DATA MODEL
Installing the Repository	Installation of a SAP PowerDesigner repository.

3.3.1 Setting up Your SAP PowerDesigner Repository

Context

After you have installed your SAP PowerDesigner repository, you can set it up. This is necessary only if you want to use the SAP PowerDesigner repository to store your data model for SAP Financial Services Data Management. To do so, please follow the steps below:

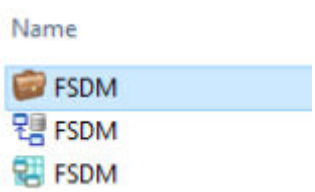
Procedure

1. Download the SAP PowerDesigner data model for SAP Financial Services Data Management and the data model extensions.

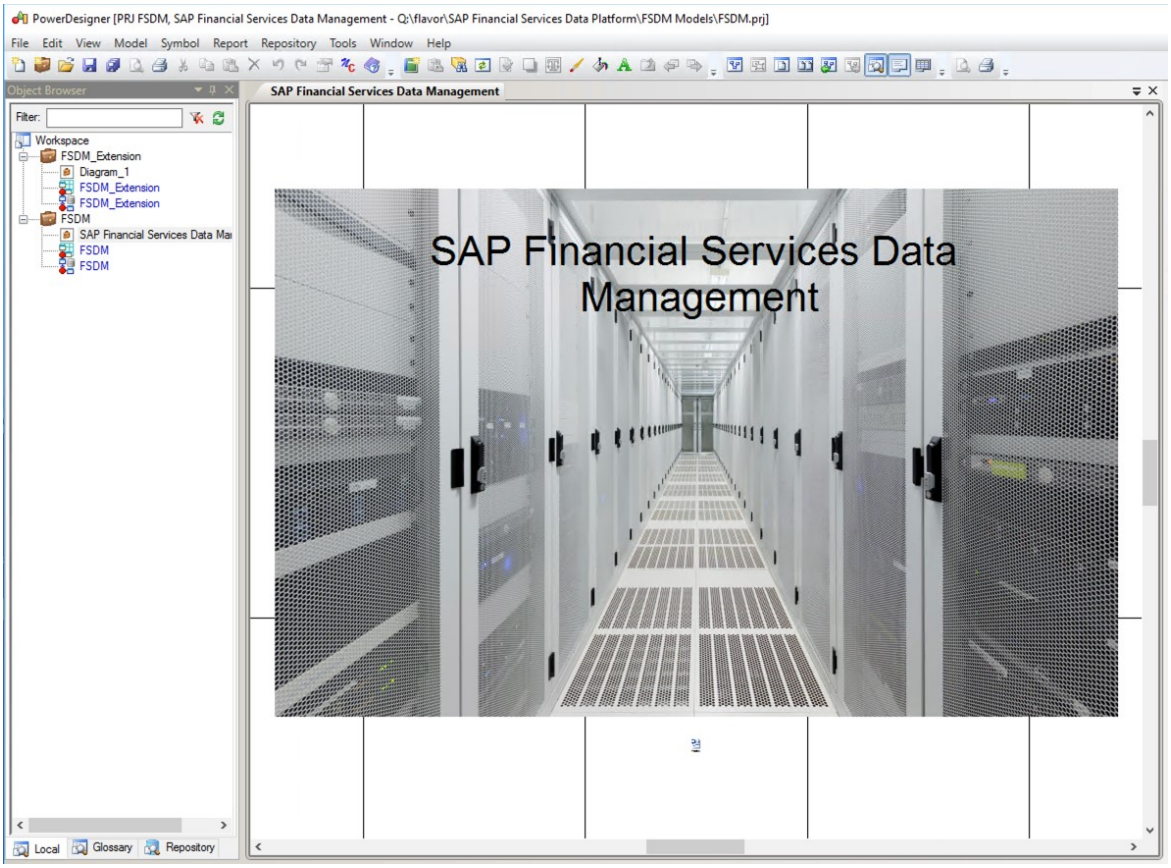
   Download the software component FSDM PD DATA MODEL (FSDM_PD_DM) at SAP Software Download Center > *Support Packages & Patches* > *Access Downloads* > *By Alphabetical Index (A-Z)* > *F* > *SAP FS DATA MANAGEMENT* > *COMPRISED SOFTWARE COMPONENT VERSIONS* >.
2. Extract the zip file containing the model files:

3. Double-click the project file to open it in SAP PowerDesigner:

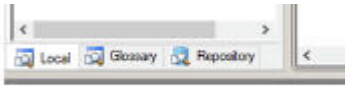
When you open the model, it should look like this:



3.3.2 Check-in Data Model

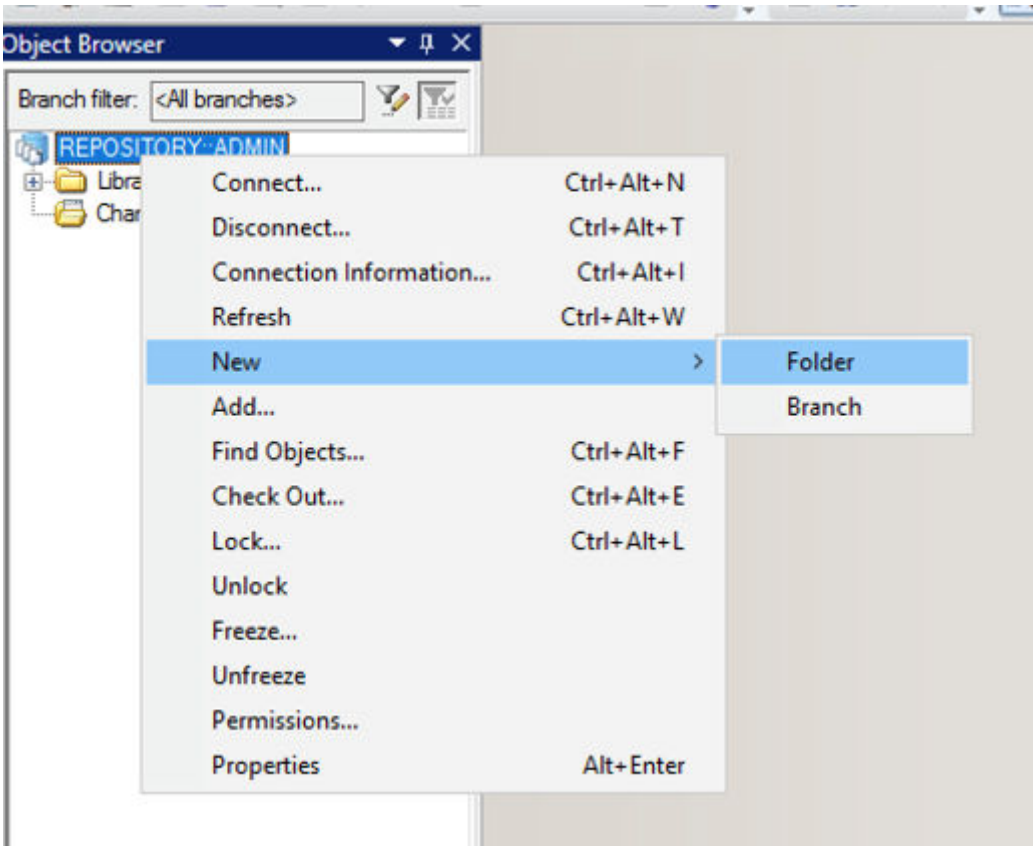
Procedure

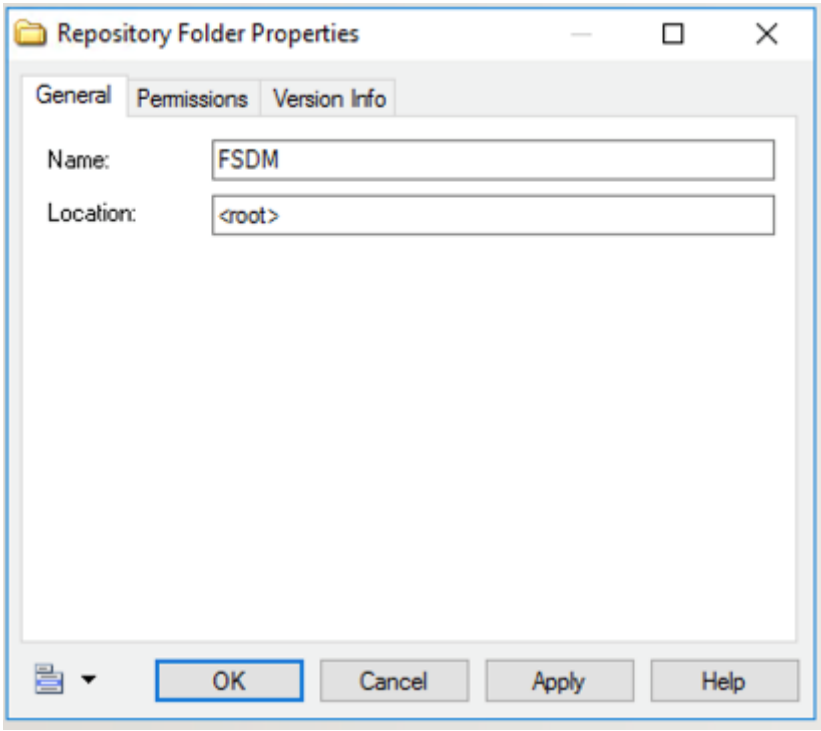
1. Create a new folder



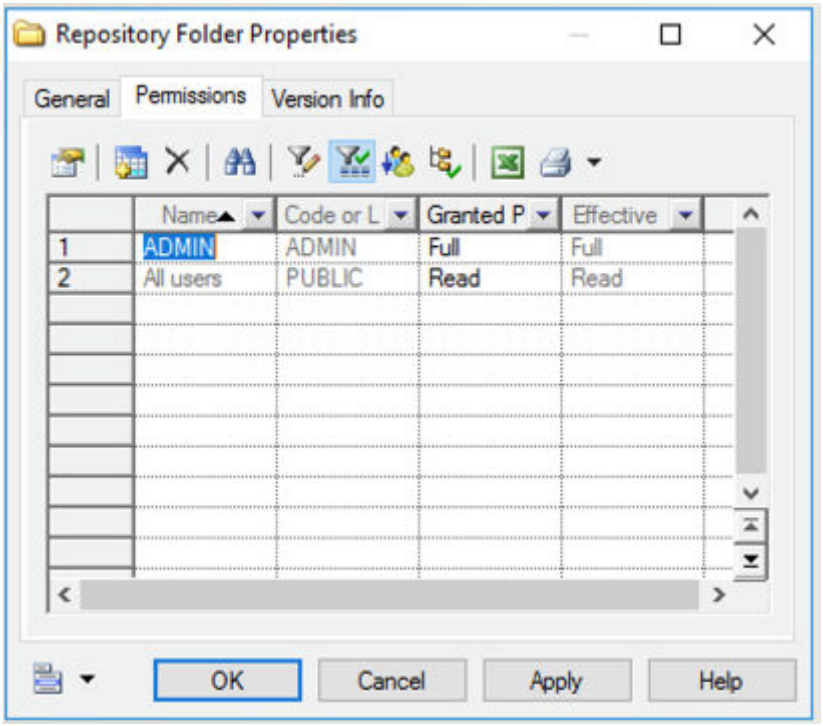
   a. Click the *Repository* tab:

b. Create a new folder (suggested name: *FSDM*):

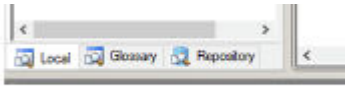




c. Keep the permissions for all users on *Read* only.

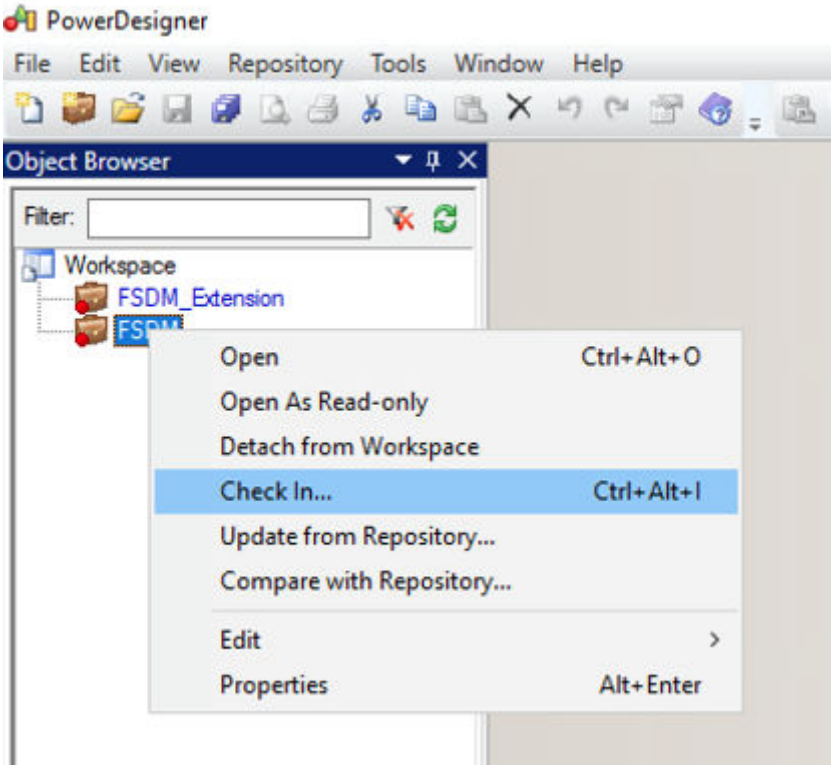


2. Check in the data model for SAP Financial Services Data Management

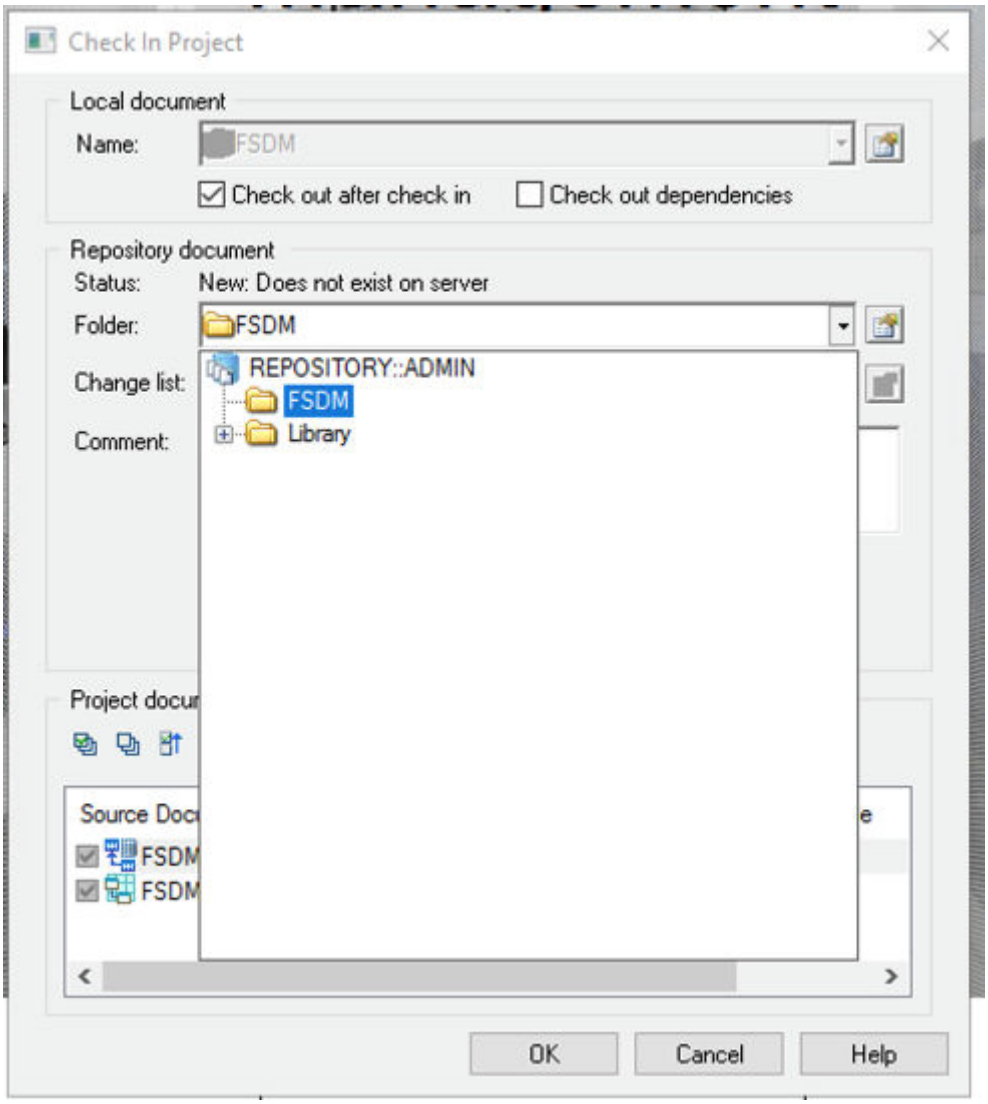


a. Click the *Local* tab:

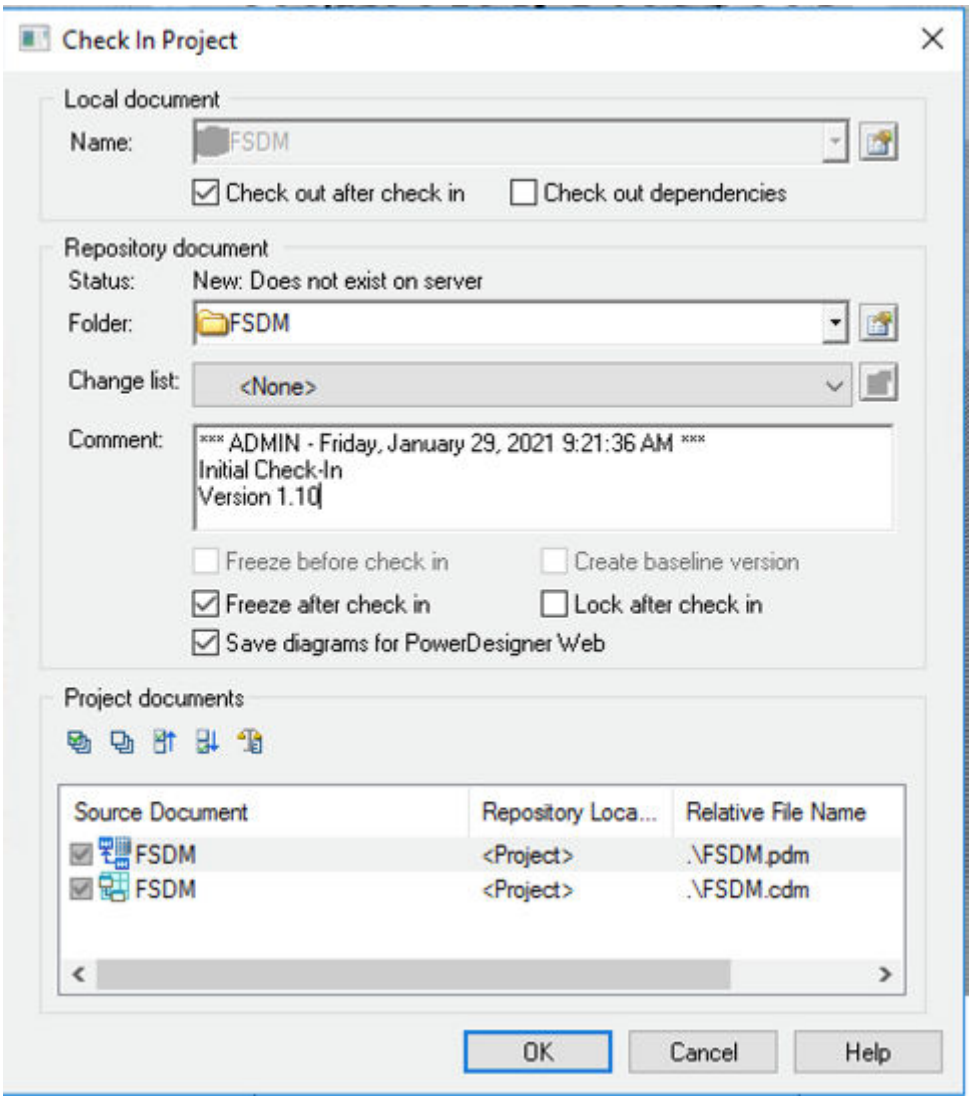
b. Right-click the project *FSDM* and select *Check In*.



c. Choose the folder *FSDM*.

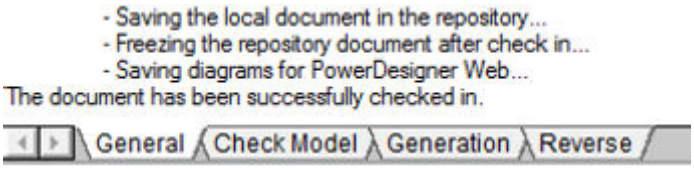


d. Check in the project.

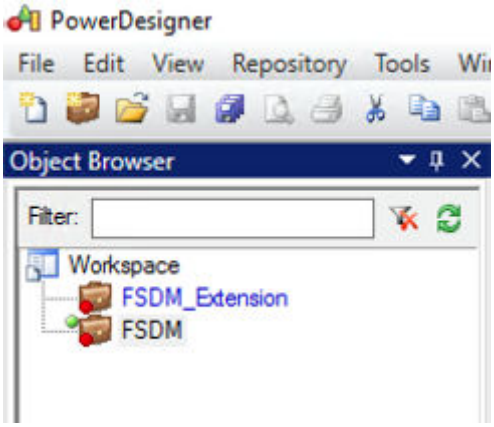


3. Check the following:

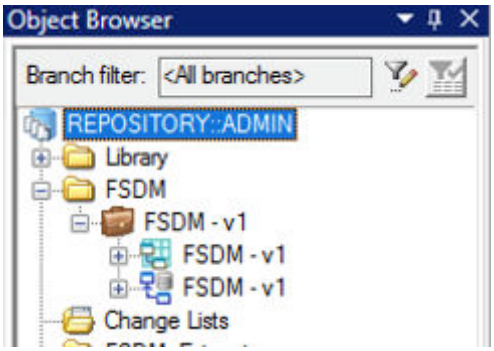
a. Check if you received the following message in the log:



b. Check if the local copy that you just checked in has a green mark next to it:



c. Check if the model is available in the repository object browser:

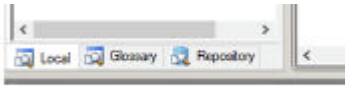


3.3.3 Check-in Data Model Extensions (Optional)

Procedure

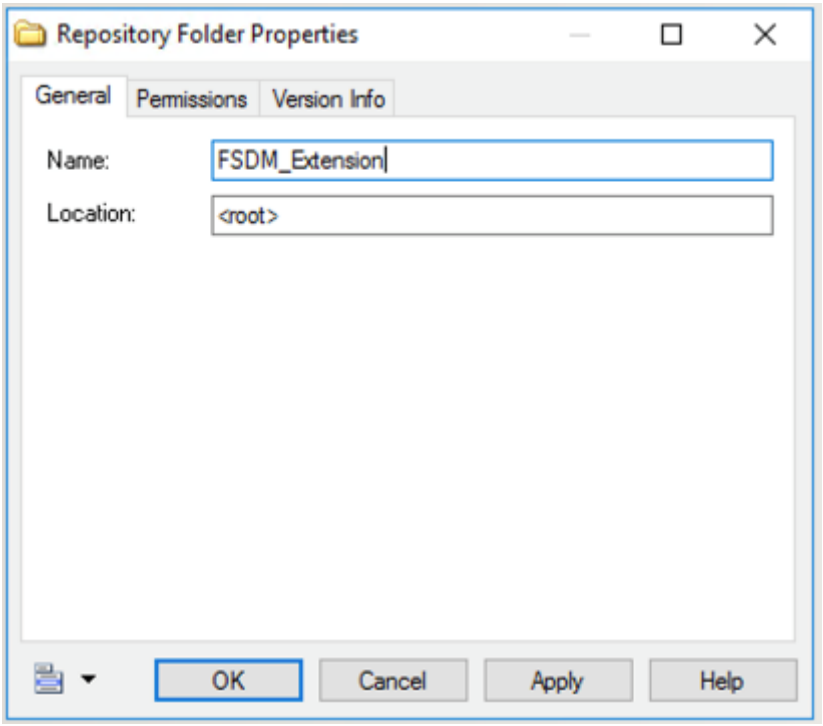
1. Check in data model extensions.

   Prerequisite for this optional step is that you have created extensions as described under Extensibility Information [page 100].

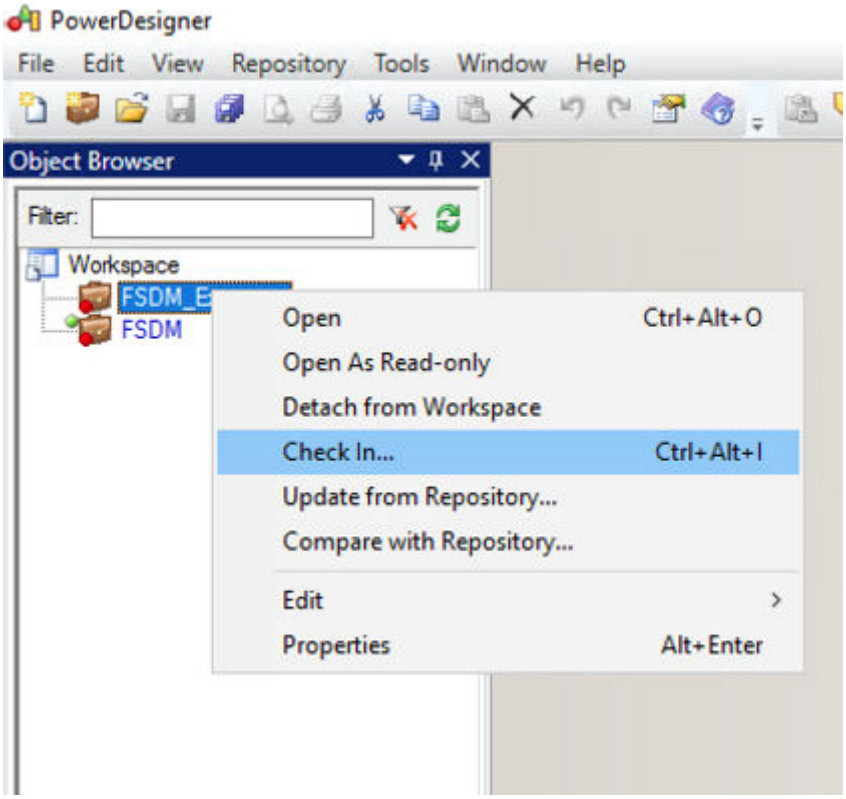
   

   a. Click the *Local* tab:

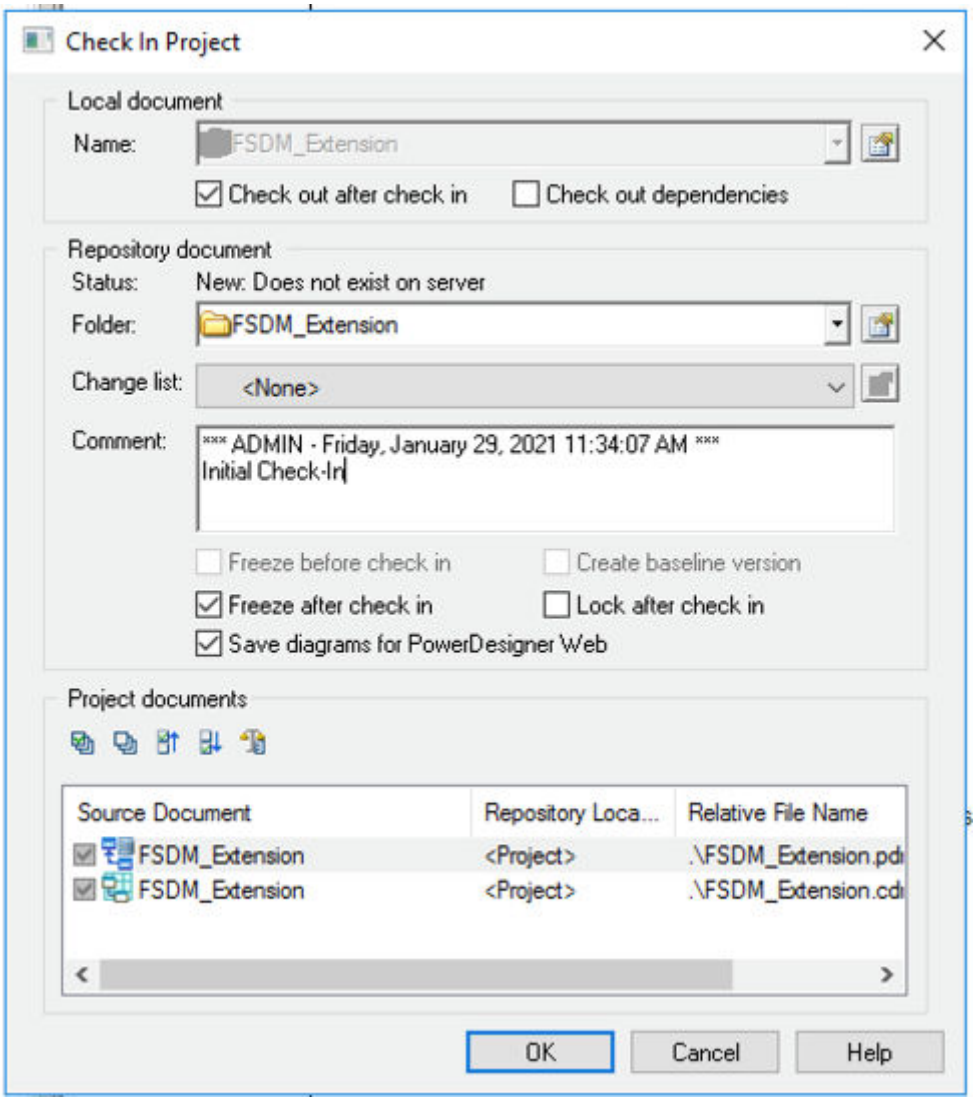
b. Create a folder for your data model extensions and choose an appropriate name.



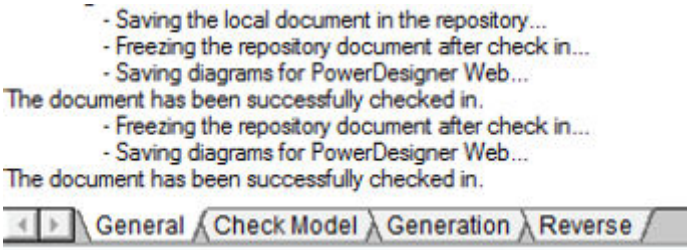
c. Open the project file for your CDM and PDM extensions and select *Check In*.



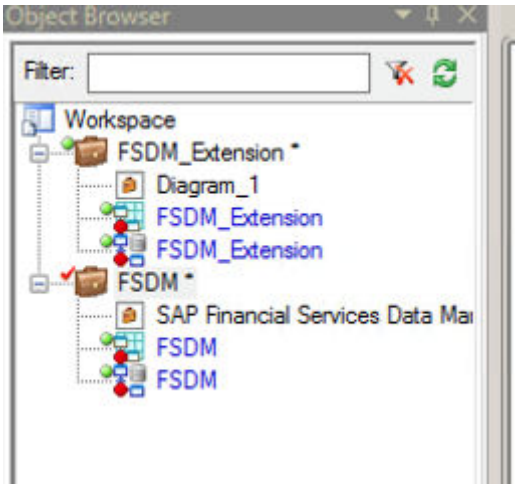
d. Choose the folder you just created and provide a meaningful check-in message:



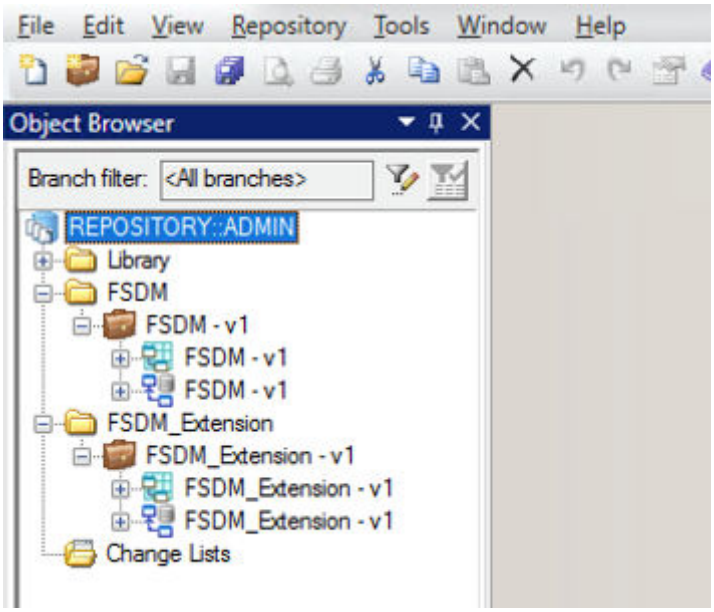
2. Make the following checks:
   a. Check if you received the following message in the log:



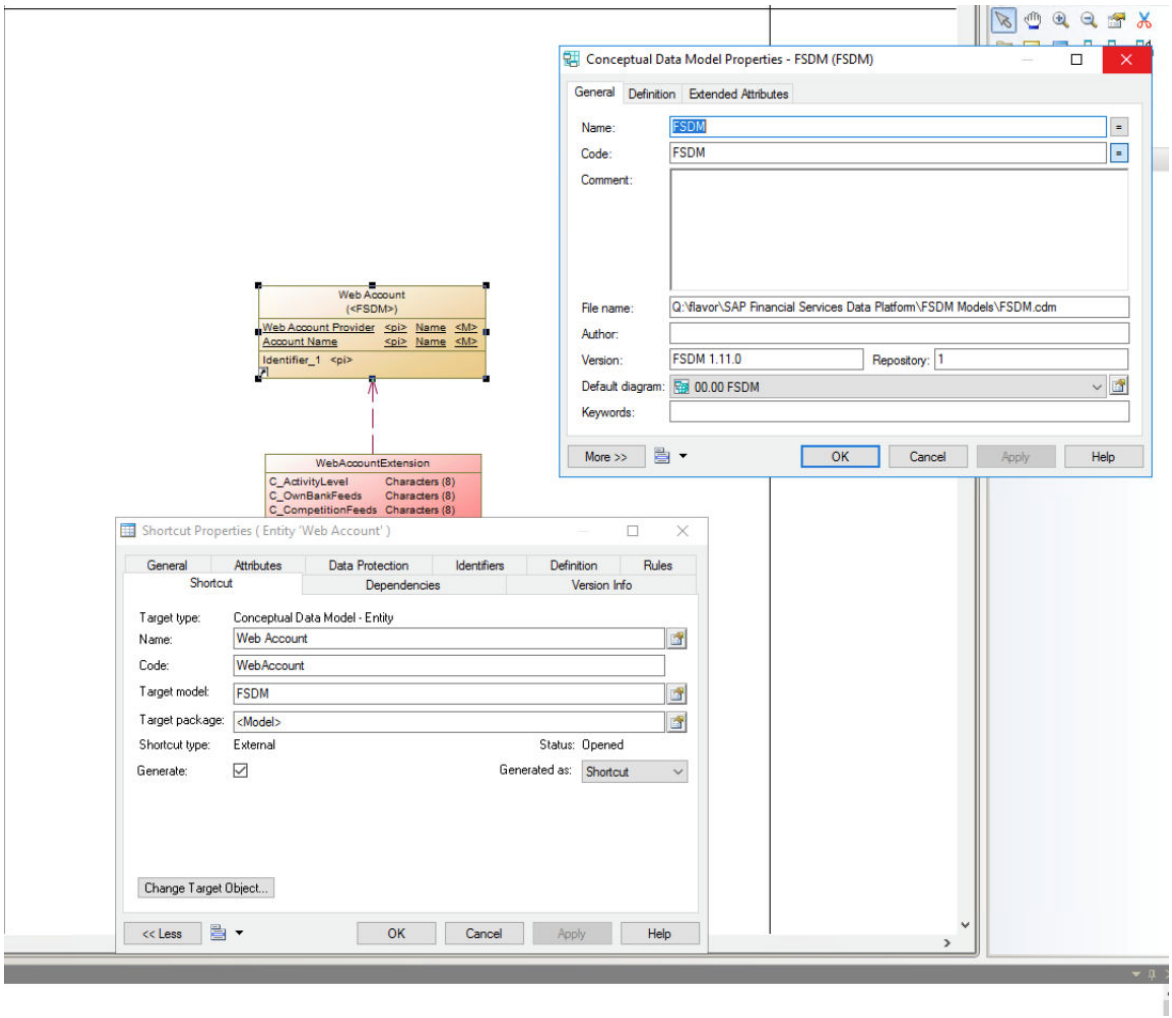
b. Check if the local copy that you just checked in has a green mark next to it:



c. Check if the model is available in the repository object browser:



3. To test your extension, open an entity in the extension that is a shortcut to the FSDM model and check that the references are still correct:



3.3.4 Upgrade Your SAP PowerDesigner Repository to a New Data Model Version

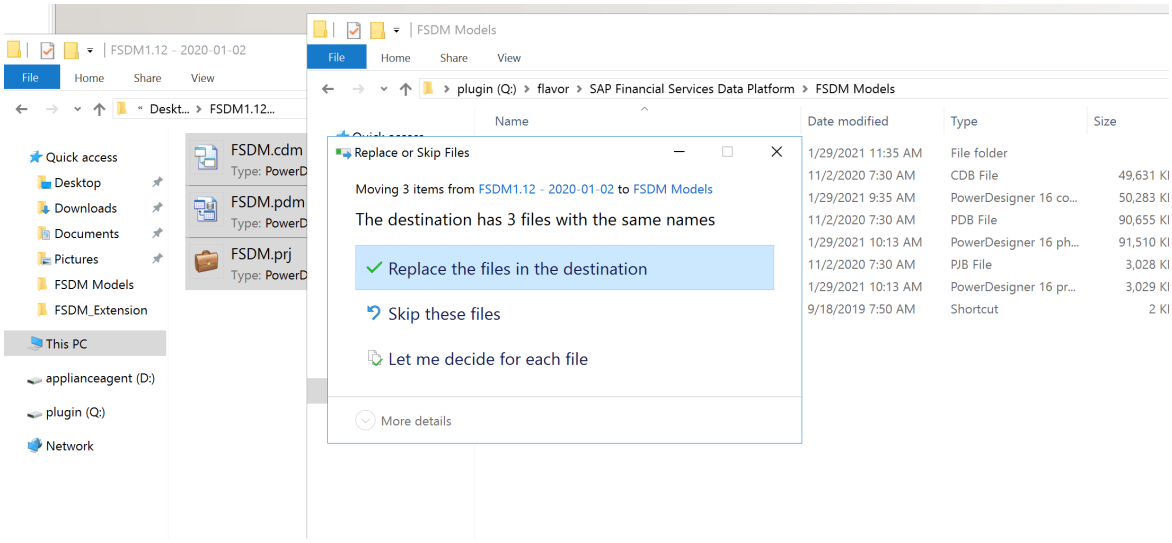
Context

You have installed a previous version of the SAP FSDM PD Data Model in your repository and want to upgrade to a new version. To upgrade the data model version, follow the steps below:

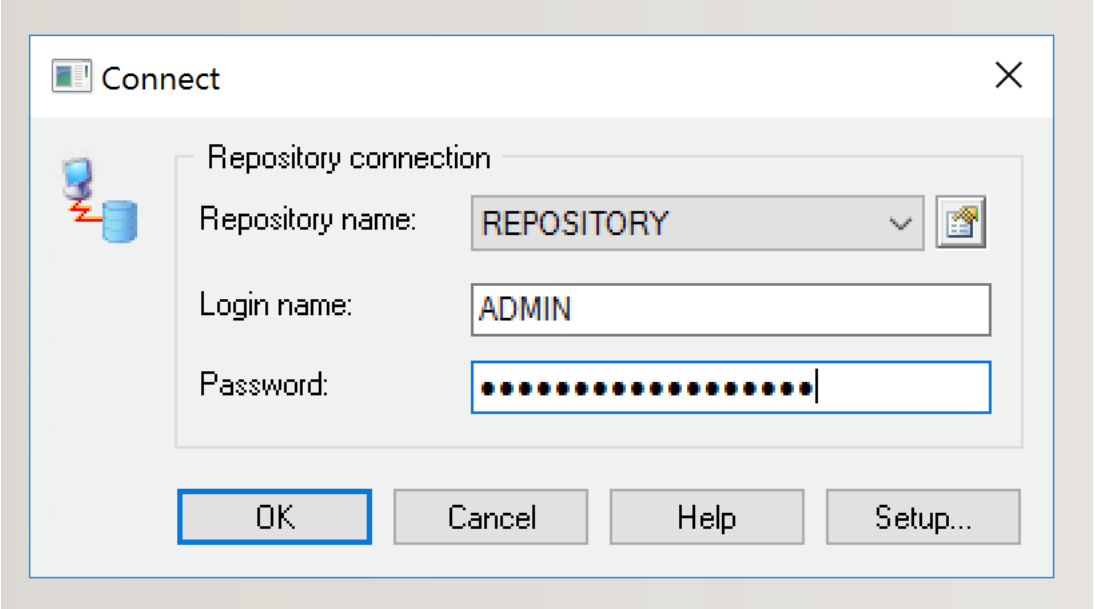
Procedure

1. Download the new version of the data model and the data model extensions.

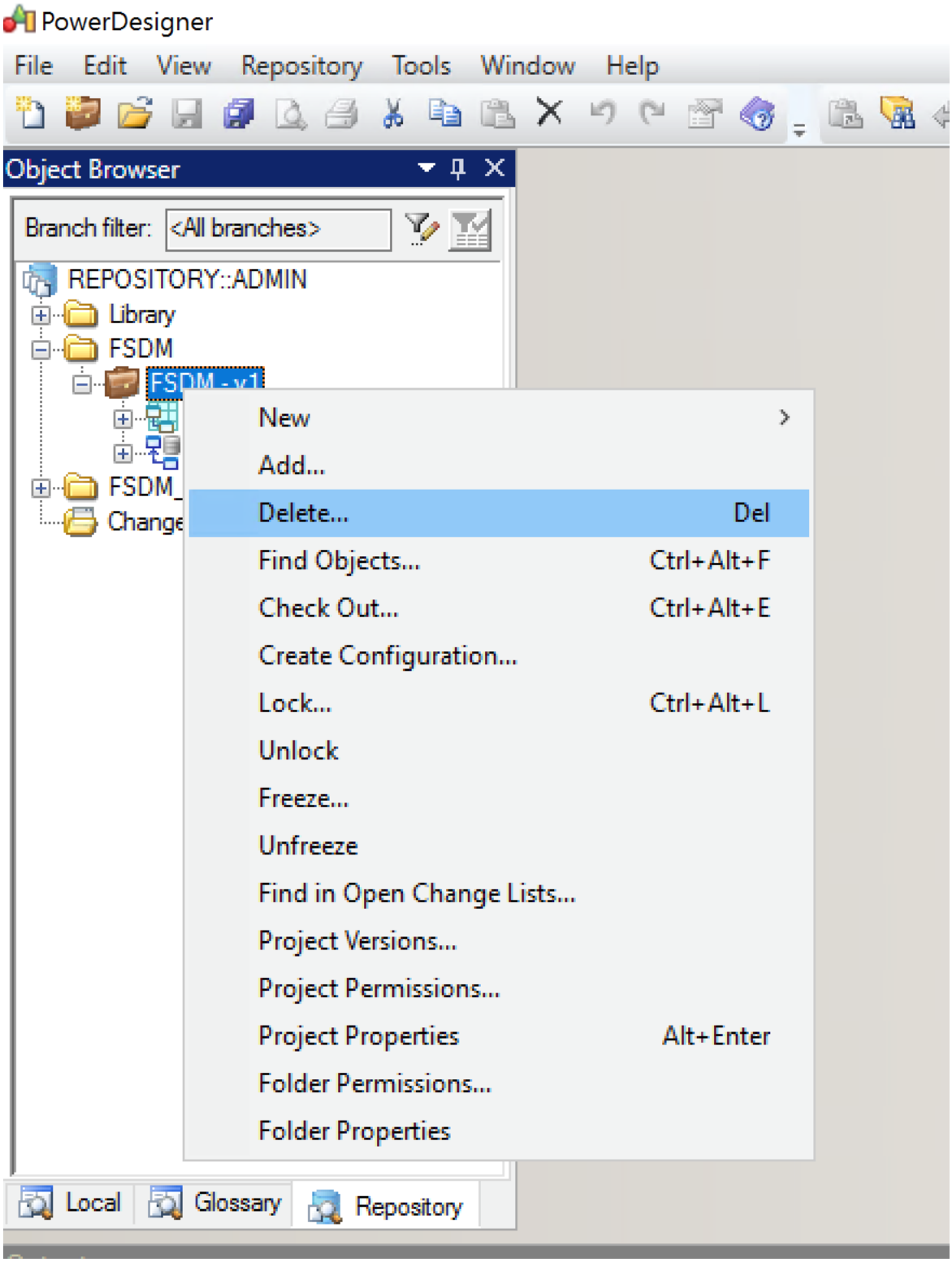
   Download the software component FSDM PD DATA MODEL (FSDM_PD_DM) at SAP Software Download Center > *Support Packages & Patches* > *Access Downloads* > *By Alphabetical Index (A-Z)* > *F* > *SAP FS DATA MANAGEMENT* > *COMPRISED SOFTWARE COMPONENT VERSIONS*.
2. We recommend that you save your current version in a backup before you check in the new data model version.
3. In your local file system, overwrite the SAP model files.

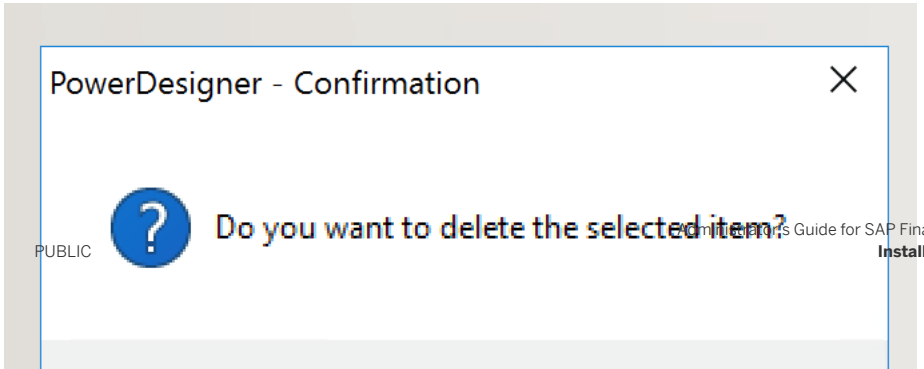


4. Connect to the repository using an admin user:

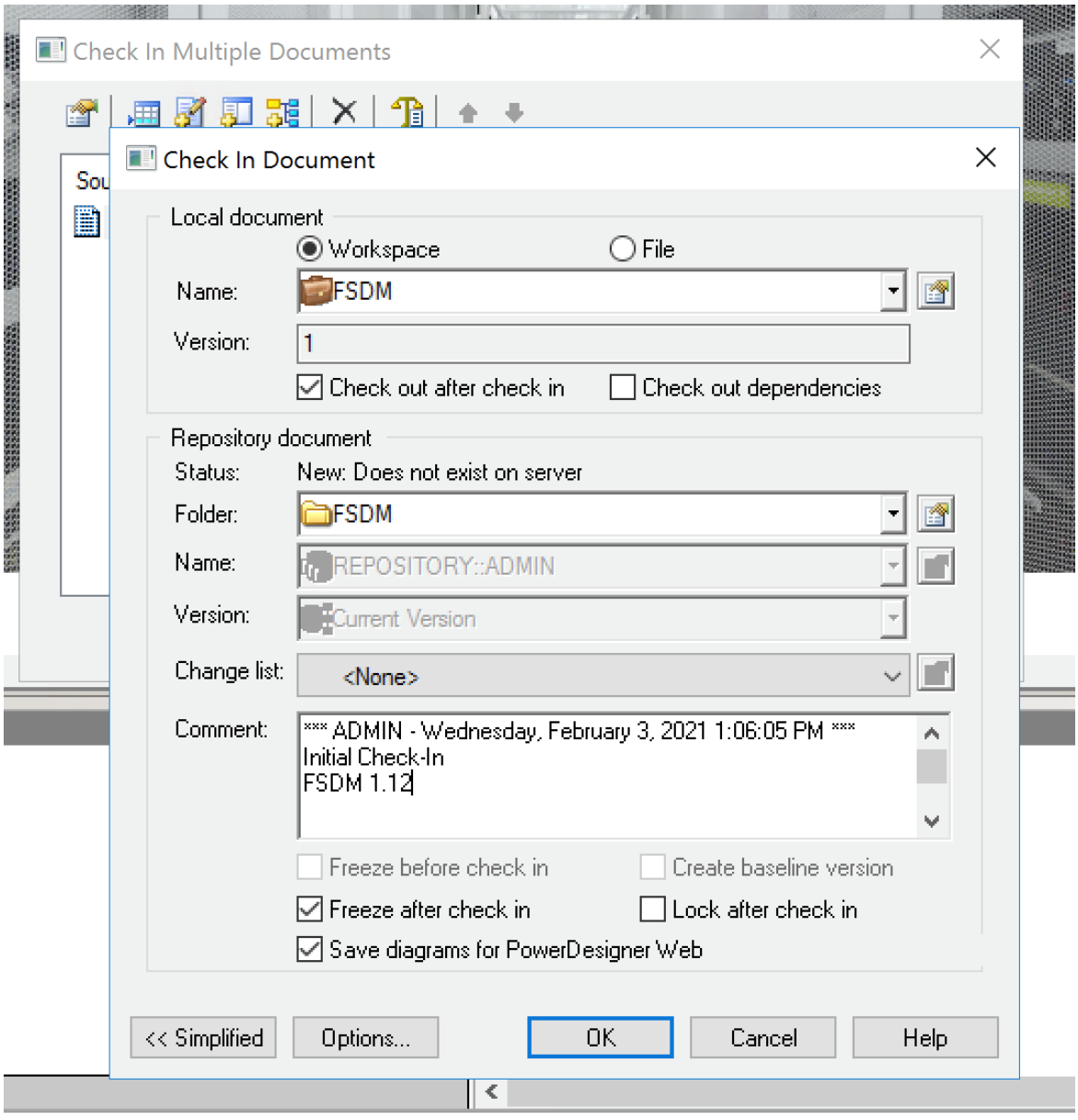


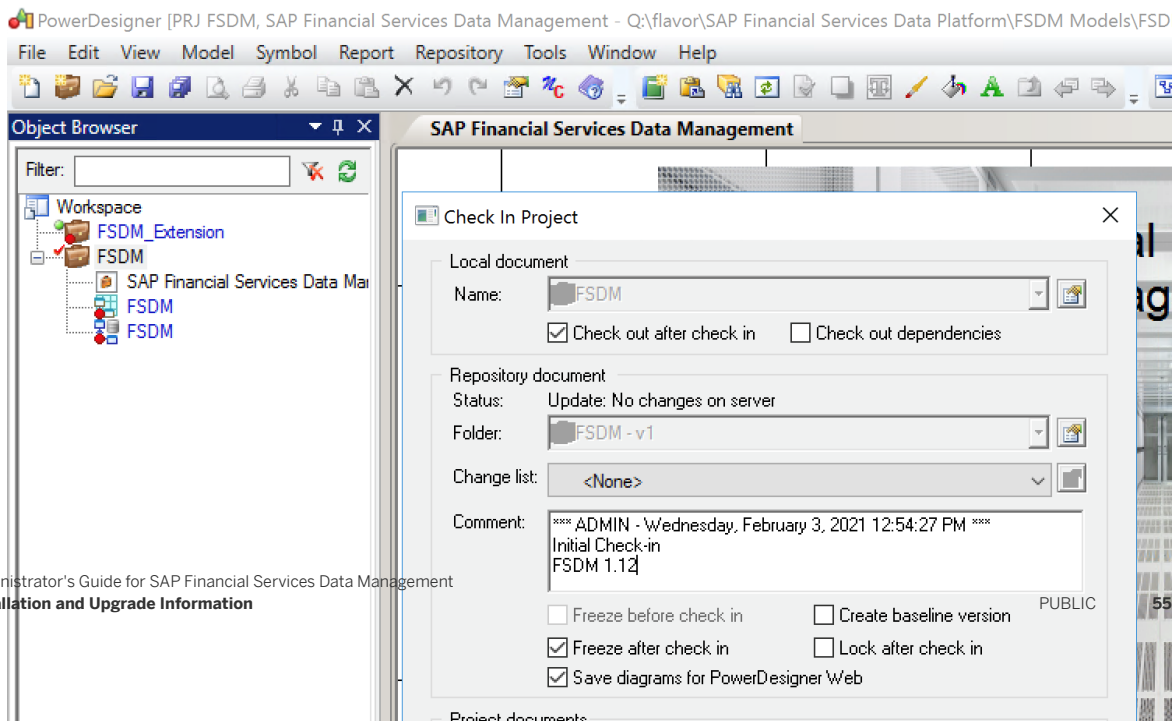
5. Delete the old FSDM model.





6. Check in the new model in the FSDM folder:





3.4 Installing and Upgrading SAP Financial Services Data Management

Software Download

Download software media at SAP Software Download Center.

The following download packages are available for the current version of SAP Financial Services Data Management. Please check the release information note 2559644 more information about latest patches:

Software Component Name	Technical Name of Software Component	Description
FSDM PD DATA MODEL	`FSDM_PD_DM`	Contains the SAP PowerDesigner data model for SAP Financial Services Data Management

FSDM DATA MODEL TEMPLATE	`XSAC_FSDM_TL`	Provides SAP WebIDE templates in form of projects:
		• *Financial Services Data Management Project*: template to facilitate the installation and extension of the data model for SAP Financial Services Data Management
		• Projects for sample content in form of template views to support you in providing data on top of the data model:
		  • *FS Data Provisioning for European Regulatory Reporting*
		  • *FS Data Provisioning for Subledger Accounting*
		  • *FS Data Provisioing from SAP Consumer Mortgage Loans*
		  • *Financial Services Data Management Conceptual Views*

Software Component Name	Technical Name of Software Component	Description
FSDM CDS DATA MODEL	`FSDM_CDS_DM`	Contains the CDS data model for SAP Financial Services Data Management as a reusable database module. If you use the template, this data model is already included and does not need to be installed again.
FSDM INTERFACE GENERATOR	`XSAC_FSDM_IG`	Contains the interface generator to generate write interfaces, read interfaces, and roles for SAP Financial Services Data Management

Installation

1. **FSDM PowerDesigner DATA MODEL**
   In SAP PowerDesigner you can display the PD data model to understand the data modeling for SAP Financial Services Data Management. For more information, see Deploying SAP Financial Services Data Management [page 75].
   To upgrade the PD data model, you need to replace the PowerDesigner files (FSDM Project Model, FSDM CDM, FSDM PDM) with the new ones.
   In addition, you can extend the data model to meet your business needs. For more information, see Extensibility Information [page 100].
2. **FSDM DATA MODEL TEMPLATE**
   - The SAP WebIDE template project *Financial Services Data Management Project* contains the data model for SAP Financial Services Data Management and facilitates the installation process and extension of the data model. To install the template project of the data model, proceed as described in Installing or Upgrading the Template to Use or Extend the Data Model [page 59].
   - To install the SAP WebIDE template projects for the sample content, *FS Data Provisioning for European Regulatory Reporting*, *FS Data Provisioning for Subledger Accounting*, *FS Data Provisioing from SAP Consumer Mortgage Loans* or *Financial Services Data Management Conceptual Views* proceed as described in the Sample Content for SAP FS Data Management.

     > ⓘ Note
     >
     > **Example Code**
     >
     > Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

3. **FSDM CDS DATA MODEL**
   We recommend that you use the SAP WebIDE template.

> **Note**
>
> If you use the template, this data model is already included and does not need to be installed again.

4. **FSDM INTERFACE GENERATOR**
   To install the interface generator to generate write interfaces, read interfaces, and roles for SAP Financial Services Data Management, proceed as described in Installing or Upgrading the Interface Generator [page 68].

Upgrade

Before upgrading your SAP Financial Services Data Management, we recommend that you have done the necessary upgrades for the required software as described in the Installation and Upgrade Information [page 31] section in this administrator's guide. Make sure you have upgraded to the latest patch level of the previous software version of SAP Financial Services Data Management. You can find an overview of the latest patches in the release information note 2559644.

1. **FSDM PD DATA MODEL**
   To upgrade the data model in SAP PowerDesigner, you have to install the new version. This replaces the existing files. The UUID remains the same. For more information, see Deploying SAP Financial Services Data Management [page 75] in the Operations Information section of this guide.
   If you have extended a previous version of the data model in SAP PowerDesigner, you have to change the target model to the newly downloaded FSDM PD file.
2. **FSDM DATA MODEL TEMPLATE**
   If you have created and possibly extended your own data model project that uses the CDS data model as a reusable database module based on the SAP Web IDE template, we recommend that you use the SAP Web IDE template to upgrade to the newest version. For more information, see Upgrading an Existing Data Model Project [page 62].
3. **FSDM CDS DATA MODEL**

   > **Note**
   >
   > If you have created and possibly extended your own data model project that uses the CDS data model as a reusable database module **based on the SAP Web IDE template**, we recommend that you use the SAP Web IDE template to upgrade to the newest version. The new version of the CDS data model is part of the new template version.

4. **FSDM INTERFACE GENERATOR**
   To upgrade the interface generator, install the new version. These files replace the existing version. For more information, see Installing or Upgrading the Interface Generator [page 68].

Related Links

Location	Content
SAP Software Download Center	You download software media at *Software Download Center* > *Support Packages & Patches* > *Access Downloads* > *By Alphabetical Index (A-Z)* > *F* > *SAP FS DATA MANAGEMENT* > *COMPRISED SOFTWARE COMPONENT VERSIONS*.
2559644	Release Information Note (RIN)
Installing or Upgrading the Interface Generator [page 68]	Installation and upgrade description within this guide
Installing or Upgrading the Template to Use or Extend the Data Model [page 59]	Installation and upgrade description within this guide
Deploying SAP Financial Services Data Management [page 75]	Installation and upgrade description within this guide

3.4.1 Installing or Upgrading the Template to Use or Extend the Data Model

We recommend that you use the SAP WebIDE project *Financial Services Data Management Project* as a template to deploy or extend the data model for SAP Financial Services Data Management. The template is technically installed as a plug-in for SAP Web IDE and can be used as described in the Operations Information [page 71].

If you have already created your own data model project based on the SAP WebIDE template, and plan to **upgrade** to a newer template version, please see Upgrading an Existing Data Model Project [page 62].

You can use the XSA Application Lifecycle Management Graphical User Interface for the installation process. If you prefer to perform the installation manually, follow the steps below.

Install or upgrade `XSAC_FSDM_TL`:

1. Enter **`xs login`**.
2. Enter **`xs target -o <organization> -s SAP`**.

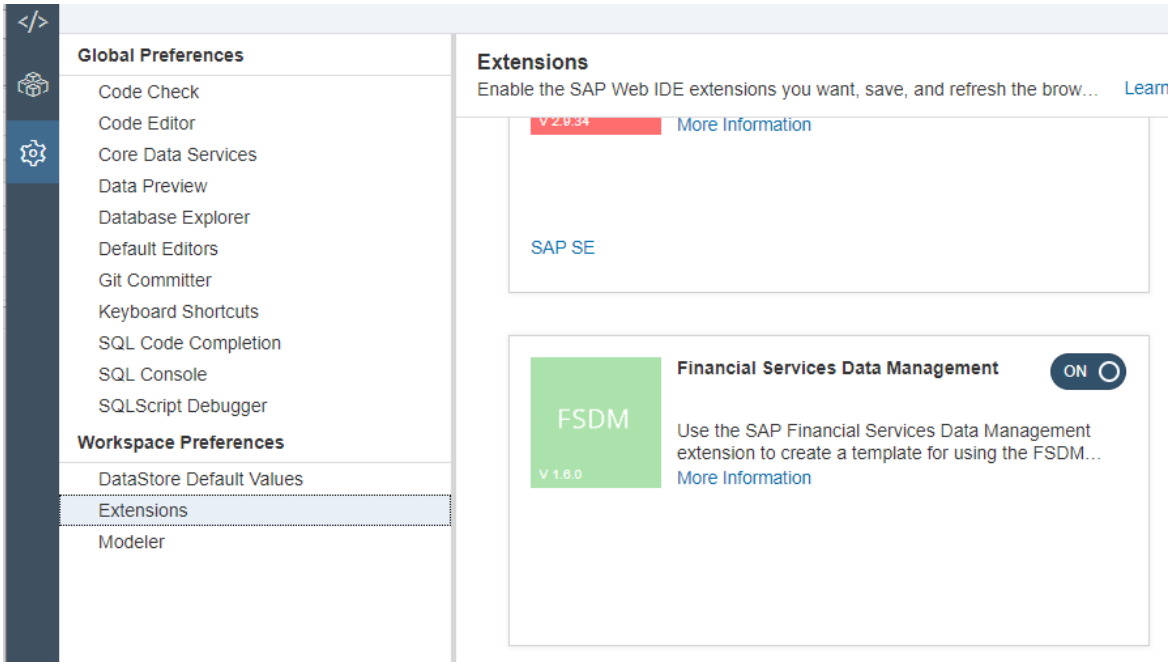
   > **Note**
   >
   > Make sure that you install the SAP Web IDE extension into the same space as SAP Web IDE, which is usually installed in the SAP space.

3. Install or upgrade **`xs install <zip_file_name>`**.
4. You can find the URL of SAP Web IDE by logging on to the SAP space `xs t -s SAP` and running `xs app webide --urls`. If you don't have authorization to the SAP space, you can find the URL under *Registered service URLs* when invoking *xs --version*.

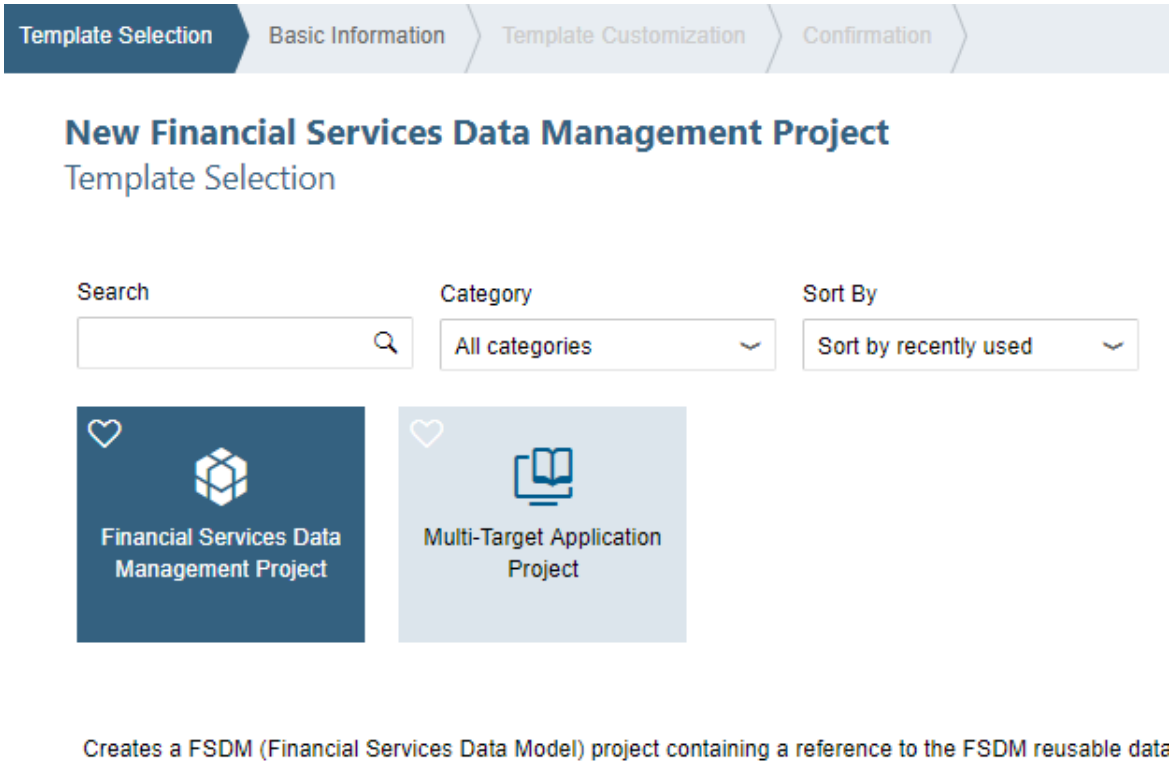
To access the template, start the SAP Web IDE application in the Web browser.

You must enable the template before you use it for the first time.

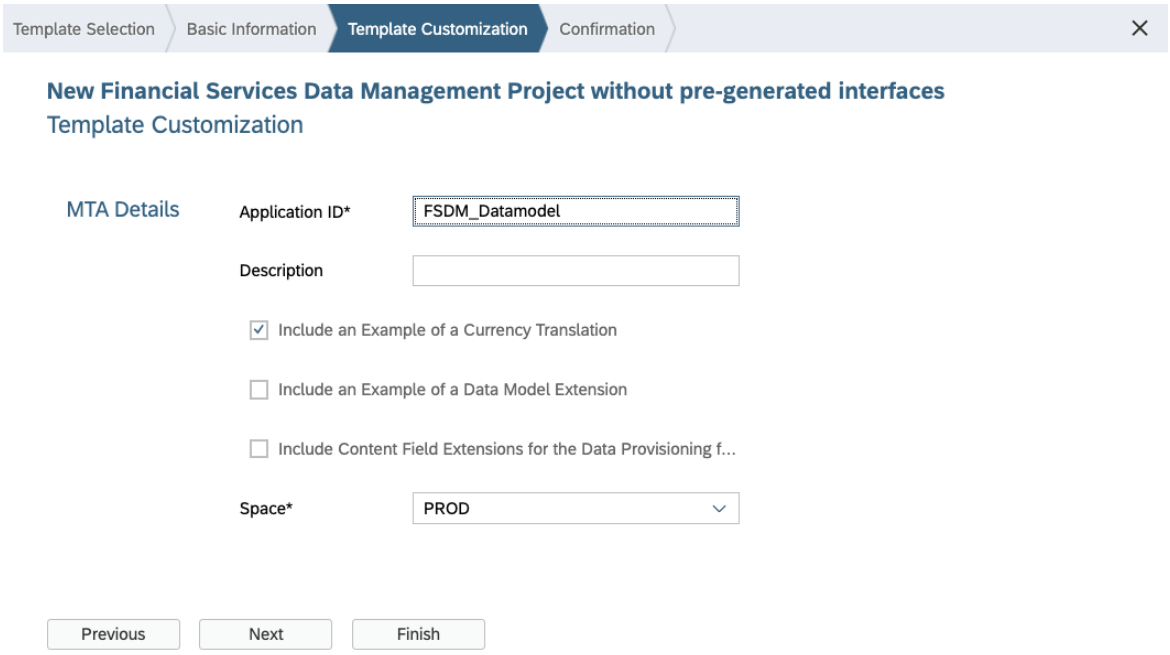
1. Go to the *Preferences* section and open the *Extensions* tab under *Workspace Preferences*. Enable the template by using the switch button and save:



2. Go to the *Development* section and choose *File* > *New* > *Project from Template*.
3. In the new window, choose the Financial Services Data Management project and continue with the wizard:



On the following screen type the name for your project and optionally mark the checkboxes.



- The checkbox *Include an Example of a Currency Translation* is already marked. You can use this example to understand how to integrate the HANA Currency Conversion (*Banking* method) for currency translation in SAP Financial Services Data Management. For more information, see Currency Translation.
- To extend the CDS data model, mark the checkbox *Include an Example of a Data Model Extension*. You can use this example to understand how to perform the extension. For more information, see the section *CDS Extension on SAP HANA* under Adding Data Fields [page 113] or Adding Entities [page 114].
- To use the content for data provisioning for subledger accounting, mark the checkbox *Include Content Field Extensions for the Data Provisioning for Subledger Content*. If you have already installed or upgraded the template and want to add the content field extensions afterwards, copy the folder *db/src/content/DataProvisioningForSubLedger* to your project.
  For more information, see SAP Note 2785178 and the attached document.

4. After you have created the project, search for your new project on the left side of the *Development* section. Expand the project folder, right-click the subfolder *db* and choose *Build*.
5. Optional Step: You can use the NPM registry to install your data model. This makes future upgrades of the data model easier.

Related Links

Location	Content
Installing and Updating Using the XS Advanced Application Lifecycle Management Graphical User Interface	Installation and upgrade description
Extensibility Information [page 100]	Extensibility information within this guide
Using the NPM Registry for Installing and Upgrading an Existing Data Model [page 66]	Describes how you can use the NPM registry to setup reusable components for your SAP Financial Services Data Management data model.

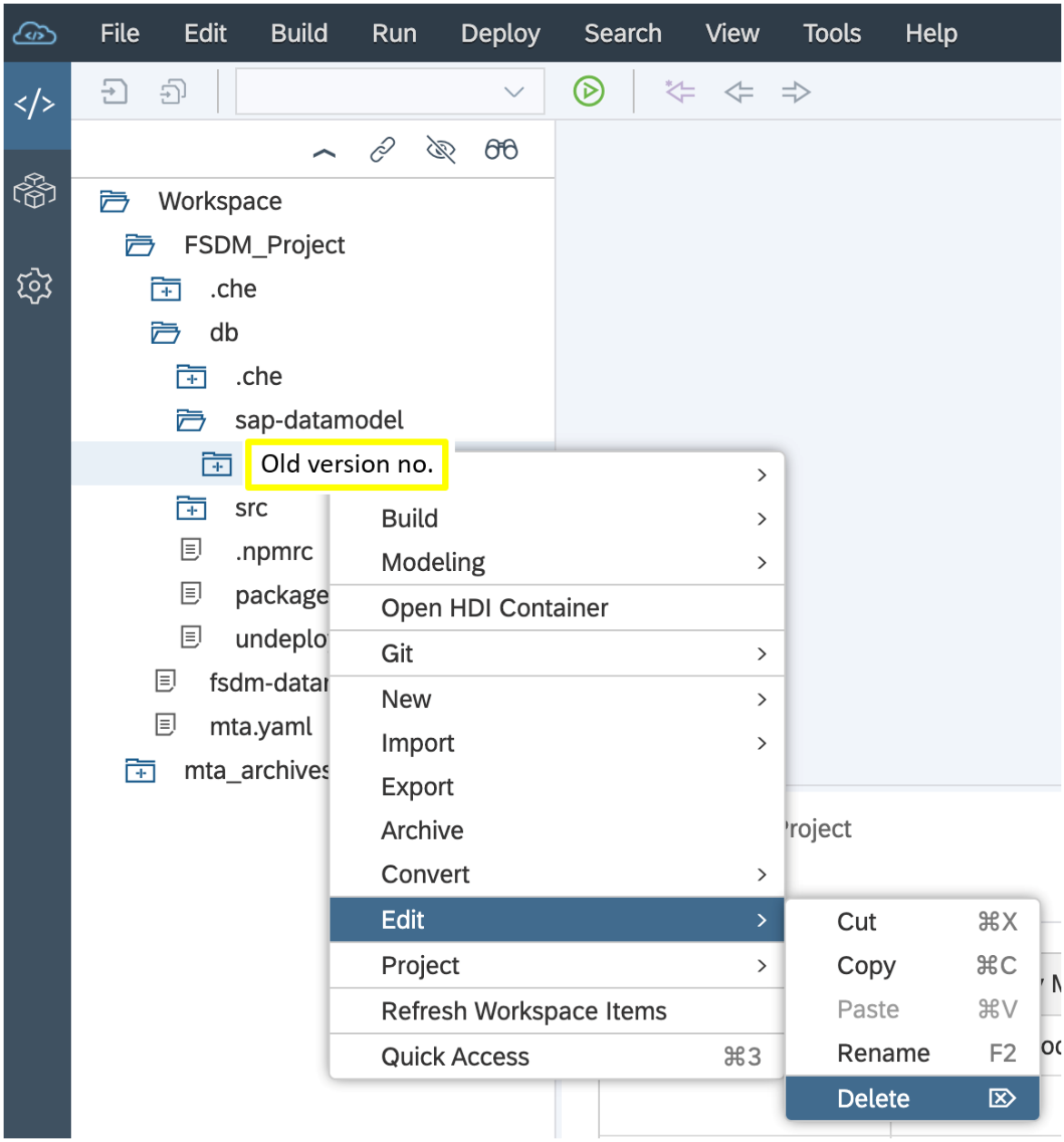
3.4.2 Upgrading an Existing Data Model Project

If you have created your own data model project that uses the CDS data model (as a reusable database module based on the SAP WebIDE template), we recommend that you use the SAP WebIDE template to upgrade to the newest version of the SAP reusable database module.

To upgrade your existing project, do the following steps. We recommend that you first upgrade and test using sample data in SAP Web IDE. You need deploy mtar in a system that already contains test data.

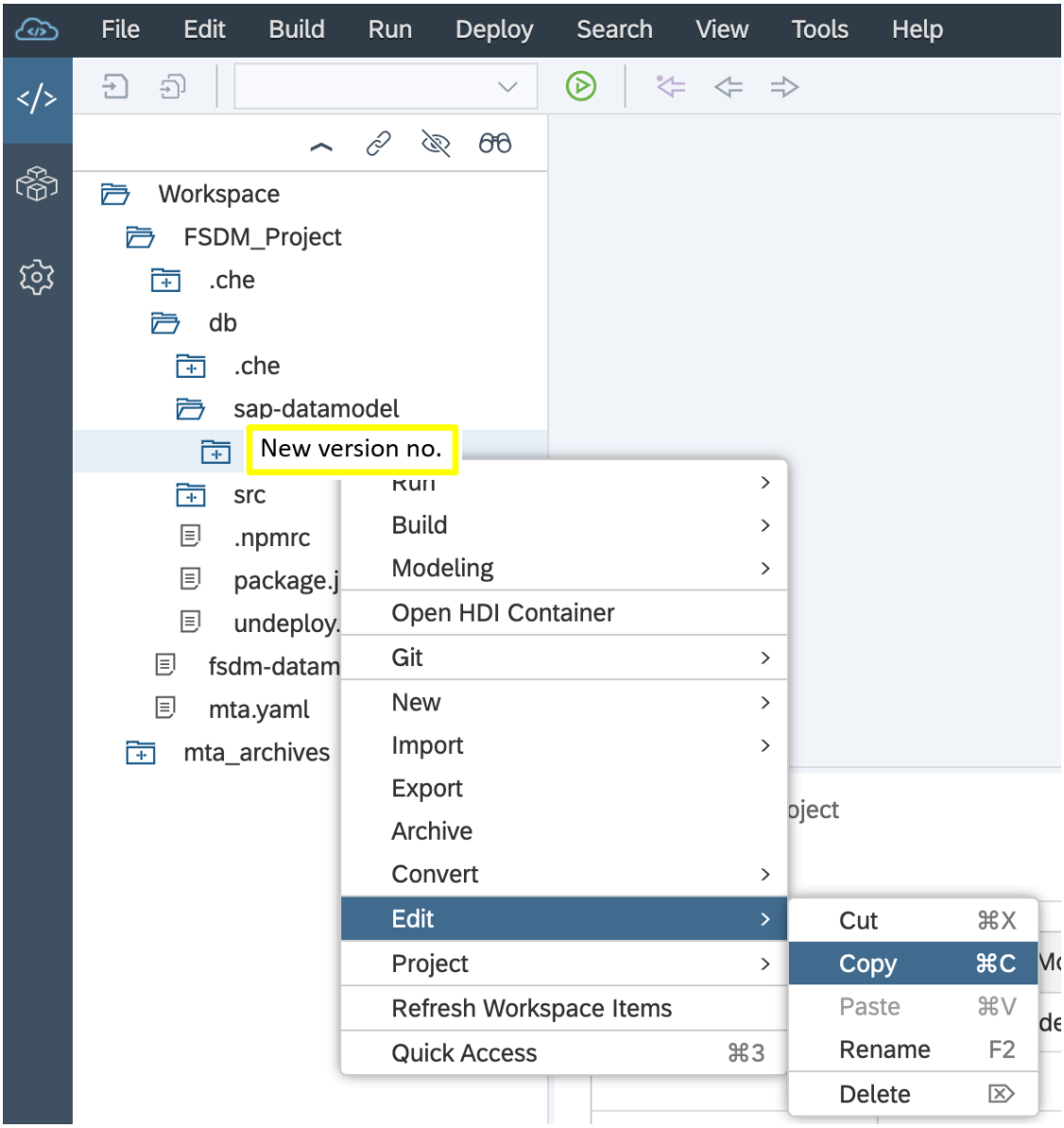
> **Note**
>
> Before you upgrade your data model project to the latest version of SAP Financial Services Data Management, make sure you have upgraded step-by-step to the latest version before it, including the available patches.

1. Check out your existing project into SAP Web IDE (from Git). We recommend that you keep your project in source code versioning.
2. Install the new template in accordance with the installation steps described in Installing or Upgrading the Template to Use or Extend the Data Model [page 59].
3. Delete the folder of the old version in the folder `sap-datamodel` of your existing project (*Right-click* ▸ *Edit* ▸ *Delete*):



> **Note**
>
> For the following steps you can either proceed as described below, or use a public NPM registry, as described under Using the NPM Registry for Installing and Upgrading an Existing Data Model [page 66].

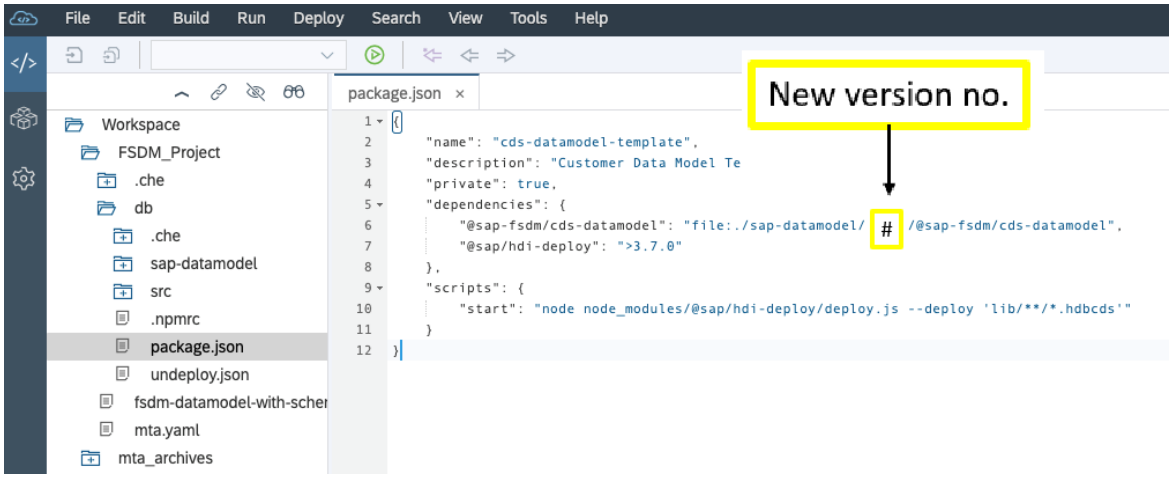
4. Create a new project from the new template. Go to the *Development* section and choose *File* > *New* > *Project from Template*.
5. Copy and paste data from the template to your folder `sap-datamodel`.
   1. Copy and paste the folder for the latest version under *Workspace* > *db* > *sap-datamodel* > *Version number (for example 1.9.0)*.
      Right-click > *Edit* > *Copy*



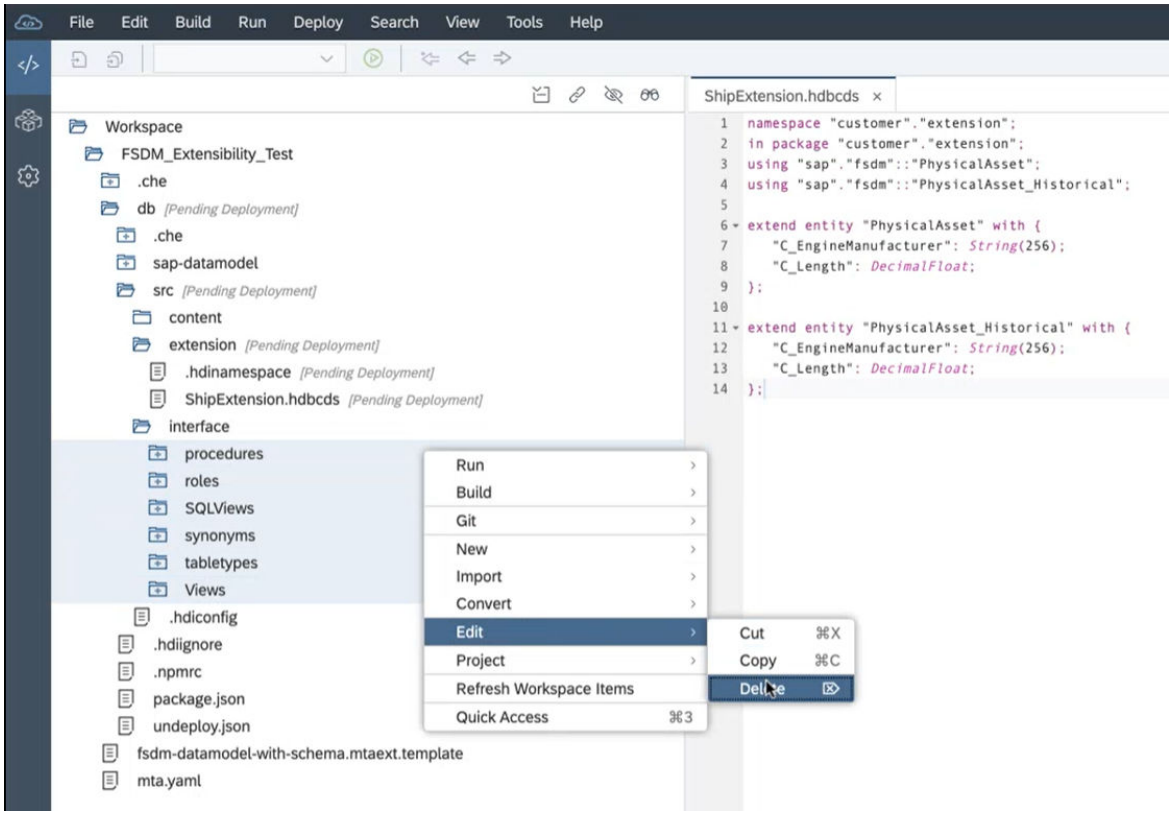
In your folder `sap-datamodel` ▶ *Right-click* ▶ *Edit* ▶ *Paste* ◀.

2. Copy and paste the folder ▶ *Workspace* ▶ *db* ▶ *undeploy.json* ◀ the same way.
3. Copy and paste the folder ▶ *Workspace* ▶ *db* ▶ *.npmrc* ◀ the same way. This folder is relevant, if you use the NPM Registry for installing or upgrading an existing data model.

6. Update `db/package.json` with the new version. If you did not change your `package.json`, you can copy it from the new template. Otherwise, edit `package.json` and replace the existing version of `@sap_fsdm/cds_datamodel` with the new version.



7. Delete the generated `interface` folder under `src`: *Right-click* > *Edit* > *Delete*



8. Test the upgrade of the data model on SAP HANA on a container with test data. You can do this either in SAP Web IDE using a build on the db folder, or on the command line using `xs deploy` on test data.
9. Regenerate the interfaces as described in Deploying the Generated Write or Read Interface to SAP HANA XS Advanced Runtime [page 77].
10. Import the zip file under *db* > *src* > *interface*.
11. Build at DB level.

Related Links

Location	Content
Installing and Updating Using the XS Advanced Application Lifecycle Management Graphical User Interface	Installation and upgrade description
Extensibility Information [page 100]	Extensibility information within this guide

3.4.2.1 Using the NPM Registry for Installing and Upgrading an Existing Data Model

SAP maintains a public NPM registry providing Node.js modules for use by application developers. This NPM registry enables you to set up reusable components for your data model rather than having to process multiple manual steps. The data model for SAP Financial Services Data Management is packaged as a reusable database module. Using the NPM registry for the installation of the data model makes it easier to upgrade the data model later.

Create a Technical User

You can create a technical user in SAP Repositories Management, which allows you to consume data from the repositories.

Prerequisites: You have a valid S-user account. The technical user that you create is automatically granted access to the repositories based on the licenses of your S-user.

> **Note**
>
> The authorizations expire after some time. Therefore, if you use the tool later, check whether your existing user is still valid.

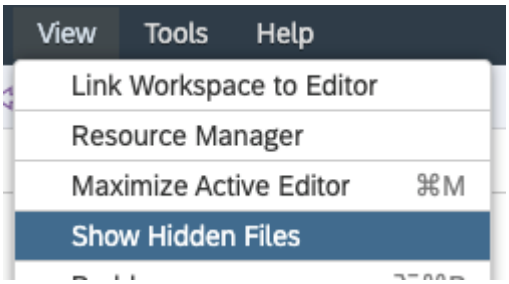
1. Open SAP Repositories Management at https://repositories.sap.ondemand.com/ui/www/webapp/ and log in with your S-user.
2. Choose *Users Management*.
   The *Create a new technical user* window appears.
3. Enter a *name* for your technical user and choose *Submit*.
   The name should contain only alphanumeric characters and can be between 3-10 symbols long. The ID of your company is added as a prefix to this name. A success message is displayed.
4. If you need to create more technical users, choose *Add* at the bottom of the *Technical users* pane.
5. To download the credentials for a technical user that you will need later, click the user to open a list of available credentials. Choose *Download* for *NPM Base64 Credentials* - base64 encoded pair of technical user name and Basic Auth Password. This string is used as the value of the authorization key in the `.npmrc` file.
6. To update the available credentials, choose *Regenerate*.
7. If you want to delete your technical user, select it and choose *Delete*. After you confirm the deletion, the user is deleted from the database as well as from all back-end systems to which it has been distributed.

Configure the NPM registry

> **Note**
>
> If you use SAP Financial Services Data Management in combination with SAP Web IDE for SAP HANA SPS 06 Patch 5, Build 4.6.6, please make sure to follow the steps described under NPM Registry Configuration for Command-Line and Build Tools to fulfill the latest security requirements.

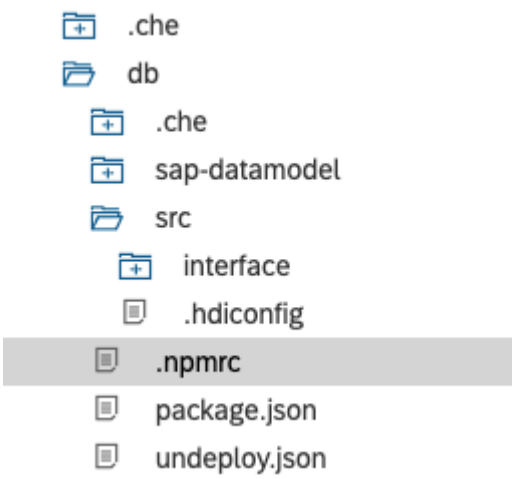
Prerequisites:

- Make sure you have activated *Show Hidden Files*.



- Make sure you have followed the steps 1-4 described in Upgrading an Existing Data Model Project [page 62].

1. Open the file *.npmrc*



2. Uncomment the marked lines in the *.npmrc* while replacing *<AUTH>* with the information downloaded in step 5 above (*NPM Base64 Credentials*)

```
1  strict-ssl=true
2  # These are the standard registries needed for HANA builds
3  registry= https://registry.npmjs.org
4  @sap:registry=https://npm.sap.com/
5  # To use the FSDM Data Model from the SAP repository,
6  # request a technical user
7  # and fill in the _auth information by replacing <AUTH> in the lines below
8  # and uncomment the lines
9  #@sap-fsdm:registry=https://73555000100900002263.npmsrv.repositories.sap.ondemand.com
10 #//73555000100900002263.npmsrv.repositories.sap.ondemand.com/:_auth="<AUTH>"
```

3. Update the version number in your *db/package.json*.


```
{
    "name": "cds-datamodel-
    "description": "Custome
    "private": true,
    "dependencies": {
        "@sap-fsdm/cds-datamodel": "  #  ",
        "@sap/hdi-deploy": ">3.3.1"
    },
    "scripts": {
        "start": "node node_modules/@sap/hdi-deploy/deploy.js"
    }
}
```

> ### Note
>
> To find out the available versions, navigate to https://7355500010090000226 3.npmsrv.repositories.sap.ondemand.com/@sap-fsdm/cds-datamodel/ in your browser or call `npm view @sap-fsdm/cds-datamodel versions` on the command line with the above mentioned `.npmrc` file in your home directory. We recommend that you use the latest version including available patch level.

4. Continue with step 8, following the instructions in Upgrading an Existing Data Model Project [page 62].

Related Links

Location	Content
The SAP NPM Registry	Describes the public NPM registry maintained by SAP.
Reusable Database Modules in XS Advanced Applications	Describes the setup of reusable components in the data persistence model of an application.

3.4.3 Installing or Upgrading the Interface Generator

The **SAP Financial Services Data Management interface generator** is a browser-based application that generates write interfaces, read interfaces, and roles.

- Write interfaces are stored procedures for writing data to the SAP HANA database tables.
- Read interfaces are SAP HANA SQL views for reading data from the SAP HANA database tables.
- Roles are used to provide cross-container access to the underlying HDI containers.

> ### Note
>
> Before installation, make sure you have installed the XSA software component `XSAC_UI5_FESV6`. It is part of the XSA content and can be installed during the XSA installation. You can check the list of installed components by running **`xs lc`**. Usually, you can find it in the SAP space.
>
> Alternatively, you can download the component from SAP Support Portal.

You can use the XSA Application Lifecycle Management Graphical User Interface to install the interface generator. If you prefer to perform the installation manually, follow the steps below. You can install the interface generator in multiple spaces. Make sure you install it in the same space in which the data model will be deployed. You can use the interface generator only if you have also deployed a data model in the same space.

Install or upgrade `XSAC_FSDM_IG.zip` using the XS command line tool:

1. Enter **`xs login`**.
2. Enter **`xs target -o <organization> -s <space>`**.
3. Install or upgrade **`xs install <zip_file_name>`**.
4. Enter **`xs app fsdm-interface-generator-ui --urls`** to identify the URLs of the interface generator.
5. Start the deployed application in the web browser using XSA_ADMIN user.

> **Note**
>
> By default, the interface generator creates all interfaces in one zip file. These interfaces are then intended to be deployed together with the data model in a single container.
>
> If you prefer a deployment in multiple containers, you can adapt the interface generator accordingly. Specify the following script during installation:
>
> **`xs deploy target/<MTA_archive>mtar -e <???>mtaext`**

Role Assignment

To enable the interface generator to download write and read interfaces, you must assign a specific role template that is generated when the interface generator application is deployed. You do this as follows for each space in which the interface generator is installed:

1. Open XS Advanced Cockpit.
2. Choose the Application Role Builder tool.
3. To create a new role collection, choose *Role Collections* under the security branch and click on *New Role Collection*.
4. Enter a name, such as "FSDMInterfaceGenerator", and enter a description for the role collection if desired. Choose *Save*.
5. The new role collection appears in the table in the middle on the right.
6. To assign a template to a role collection, select the collection in the list and choose *Roles* > *Add Role*.
7. Select the application name from the dropdown list. The application name has the instance number as the suffix to *fsdm-interface-generator*. For details on how to find the correct application name, see the following steps:
   1. Open SAP HANA XS Advanced cockpit.
   2. Navigate to the space in which the interface generator is installed.
   3. Select the application *fsdm-interface-generator-js*.
   4. Select *Environment Variables* for the application.
   5. The value of property *xsappname* is the name of the application to be selected during the role building.

   Alternatively, you can use the command **`xs env fsdm-interface-generator-js`** to get the environment variables.
8. Select users from the template name dropdown list.

9. Make the same selection from the application role dropdown list.
10. Choose the *Home* pushbutton.
11. Select the User Management tool.
12. Select the user to which you want to assign the new role collection and choose *Role Collections* > *Add*.
13. A list of role collections appears. Select the role collection that you created and choose *Save*.

Upgrade

To upgrade the interface generator, you have to install the new version in accordance with the installation steps described above.

More Information

- Installing and Updating Using the XS Advanced Application Lifecycle Management Graphical User Interface
- Operations Information [page 71]

3.5 Installing Reporting Tools

You can install additional front-end tools for reporting purposes. We recommend that you use SAP BusinessObjects Analysis for Microsoft Office 2.5 SP1 in addition to the SAP HANA EPM-MDS component.

Related Links

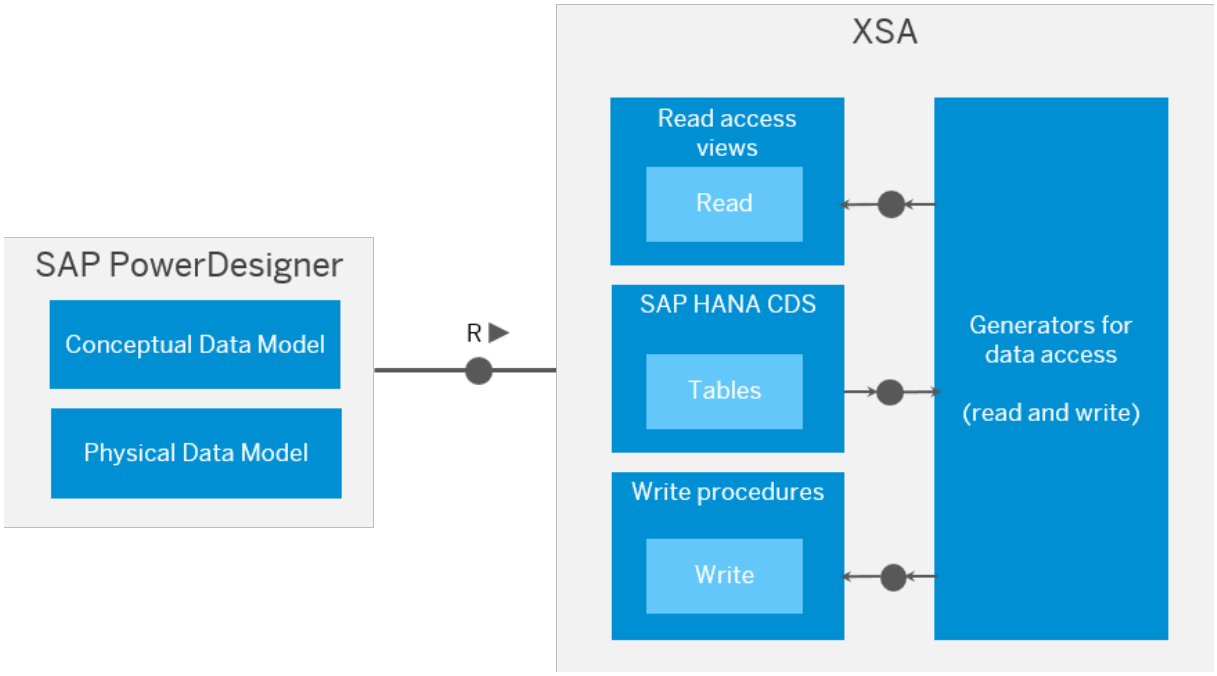
Location	Content
SAP Analysis for Microsoft Office	see there especially the Administrator Guide
2444261	Versioning and delivery strategy of the SAP HANA EPM-MDS component

4 Operations Information

SAP Financial Services Data Management contains the following parts necessary for deployment:

- An **SAP PowerDesigner project** containing a project file, a conceptual data model (CDM), a physical data model (PDM), and an extension file
- A **template** that explains how to start the deployment or extension of the SAP Financial Services Data Management data model
- An **interface generator** to generate and deploy the write and read interfaces for the data model

The following figure is an overview of how data is organized in SAP Financial Services Data Management:



Tools and Environments for Deployment

Before you start, make yourself familiar with the different tools and environments that you will come across:

SAP PowerDesigner for data modeling

You import and open the SAP FS Data Management data model in SAP PowerDesigner. Here you can display the data model and also make your own extensions to the CDM and PDM. For more information about data modeling, see the SAP PowerDesigner documentation.

> **Note**
>
> SAP PowerDesigner is the central point for accessing information about the data model and its constituent parts.

Tools for deploying the data model

- **SAP Web IDE for SAP HANA**: This is a browser-based development environment for development and testing in the SAP HANA XS advanced runtime environment and user-specific test deployments on the SAP HANA database. It includes the SAP HANA database explorer and is tightly integrated with SAP HANA runtime tools, HDI for XS advanced, and the XS advanced runtime platform. The SAP HANA database explorer contains the features and functions required to browse, view, run, and visualize the content of all types of catalog objects in HDI containers, such as tables, views, stored procedures, synonyms, and so on. This tool is intended for viewing the physical data model and adding fields using the CDS extension mechanism.
  You can build your whole project in SAP WebIDE or use the multi-target application archive builder in a CI project. For more information, see the following links:
  - SAP Web IDE
  - Continuous Integration and Delivery Best Practices Guide
  - The Multi-Target Application Model
- **XS command line tool**: You have to install the XS command-line (CLI) client on your development machine to be able to use the command line tool. The XS CLI client tools are installed by default on the SAP HANA server. However, you must download and install the client tools locally to connect to SAP HANA from your local machine.
  This tool is intended for regular deployment, possibly from within a continuous integration pipeline. For more information, see the relevant section in the SAP HANA Developer Guide for XS Advanced Model.

> → Recommendation
>
> We recommend that you use the XS command line tool for deployment. However, you can use SAP Web IDE first to deploy to a user-specific container for testing purposes.
>
> For general guidelines, see Continuous Integration and Delivery Best Practices Guide.

Interface generator for read/write interfaces and roles

The **SAP FS Data Management interface generator** is a browser-based application that generates modeled database artifacts. We strongly recommend that you use these artifacts to read or write data into the database tables.

After you have deployed the data model, you can create the following artifacts with the interface generator:

Interface Type	More information
Read Interfaces: Generated for each entity in the deployed data model. They ensure that the correct version is retrieved from the underlying database tables.	There are two kinds of read views: • **Read view with input parameters (SQL view)**: Depending on the entity, this view can have two parameters, one parameter, or none. You can enter application time and system time. • **Read view without input parameters (SQL view)**: This view returns all the current versions from the table. In addition, you can specify your own queries, for example, for a certain period or time. For more information, see Versioning.

Interface Type	More information
Write Interfaces: Realized as stored procedures. Two sets of procedures are generated: • **Data loading procedures** to load data into the corresponding tables of the entity • **Logical delete procedures** to delete the data for an entity *Write Interfaces for Erase*: Realized as physical delete procedures. They delete the data for an entity from the corresponding tables.	Each procedure has a corresponding read-only procedure, which outputs the records of insert and delete operations. Write procedures do not include delta handling in source data. Therefore, make sure that you send only delta data to the write procedures. Write procedures do not perform any business or application checks. Make sure that you perform these checks or complete the data before you call the procedures. Application/business time versioning fields follow a closed-open period model without overlaps and must be specified by the consumer of the write interface. The consumer of the generated write interface must provide the business start and end date/times of the validity period. The valid-to date is not set to 31-12-9999 by default in the generated write interface. It is, however, possible to have a gap in the application time dimension if there is no data for a certain period. We provide procedures for entities without versioning that insert only new records and update existing ones. This can be useful if you prefer to insert data rather than overwrite it. **Deletion by partial keys:** **Logical delete** and **erase** procedures can be used to delete entries from FSDM tables by specifying partial keys in the input table types. Thus, if a key field is passed as null and a corresponding entry with other key fields is found, the matching records are deleted (moved to history tables in the case of **logical delete** procedures and permanently deleted in the case of **erase** procedures).
Roles: Provide access to underlying HDI containers.	By default, the generated read and write interfaces are deployed in one HDI container with the data model. Roles are provided for accessing these views.

A list of all HDI containers available in the system is displayed on the interface generator UI. Specify the HDI container in which the data model is deployed to generate the corresponding read and write interfaces as well as roles.

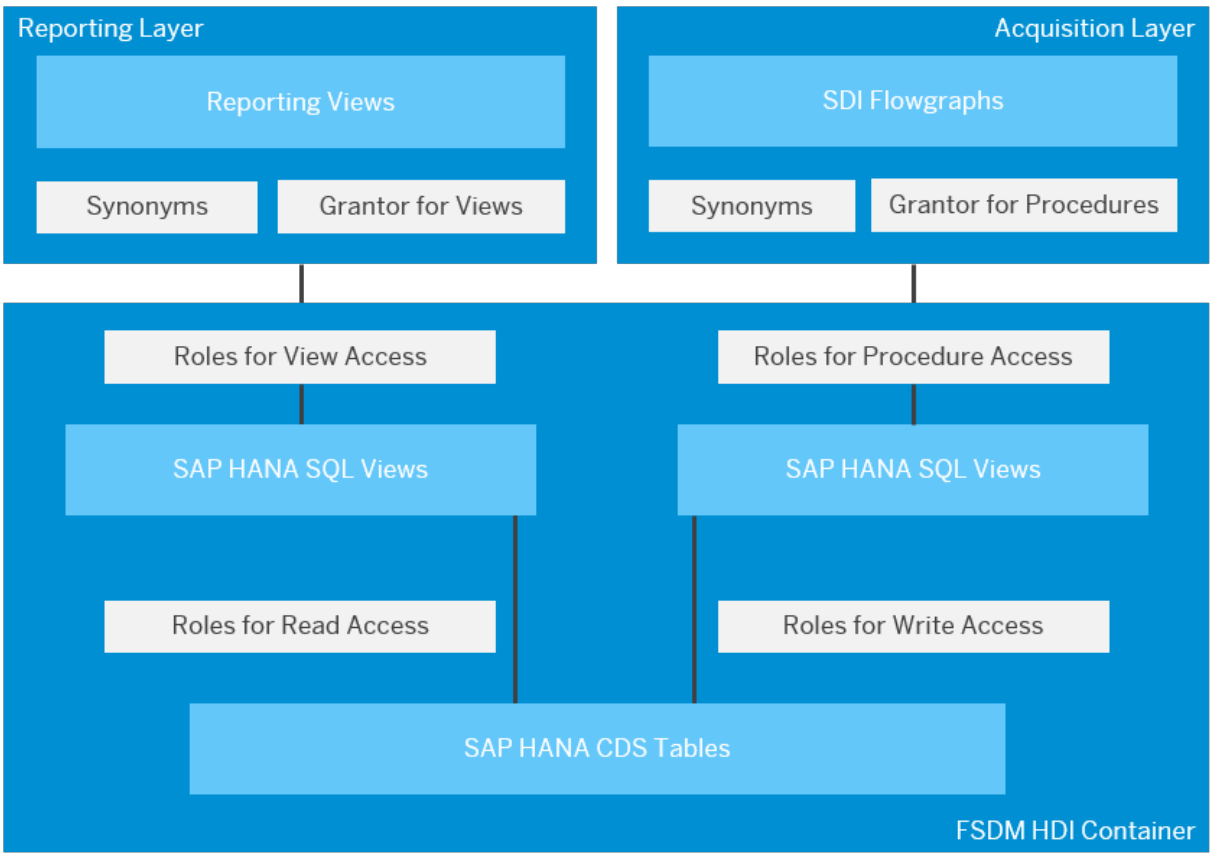
> ⓘ Note
>
> The read and write interfaces are not generated for the tables storing code lists (such as CountryCode, CurrencyCode, BusinessPartnerCategory tables).

Information about the generated artifacts is logged in the interface generator application logs. Each entry is assigned a corresponding log ID and contains details about the user who generated the artifacts.

HDI Container Structure (Single Container Approach)

The generated read and write interfaces and the SAP FS Data Management data model are deployed in a single HDI container. Depending on your security concept, you may want to consider dividing the reporting and acquisition layers. For more information, see User Administration and Authorization [page 28].

The following figure provides an overview of the HDI container structure:



HDI Container Structure (Multiple Container Approach)

The generated read and write interfaces are deployed in separate HDI containers, which are different from the HDI container in which the SAP Financial Services Data Management data model is deployed:

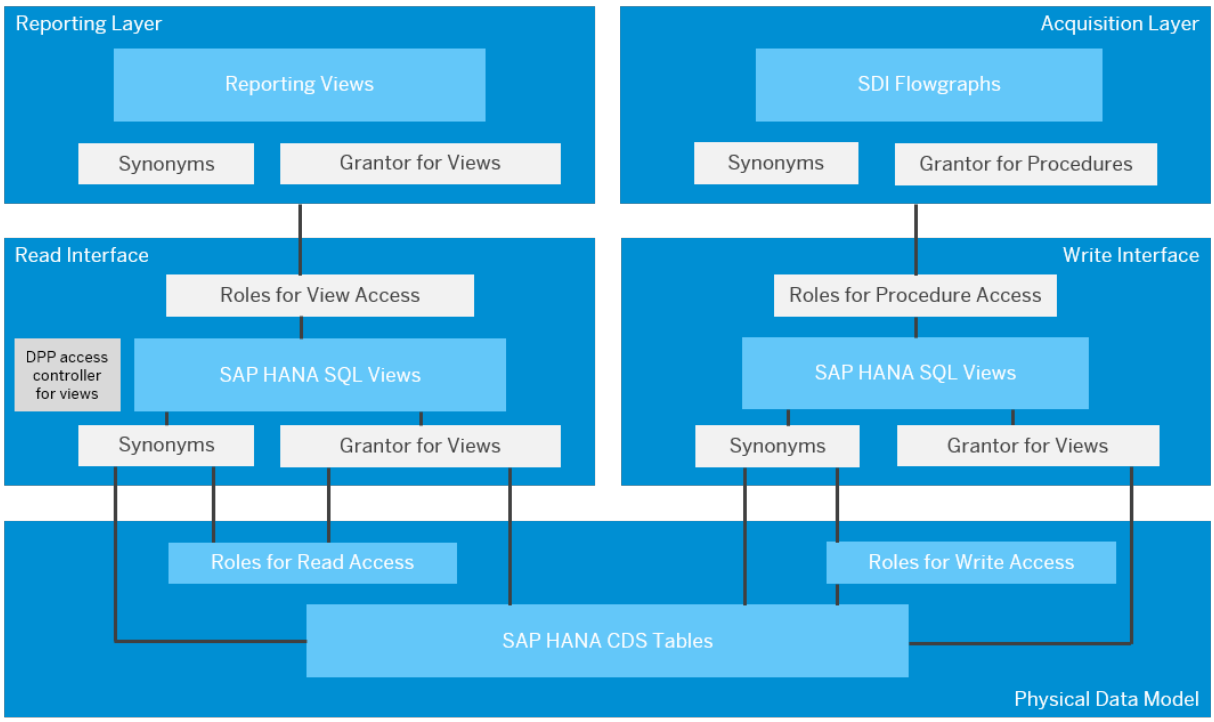
1. HDI container for **tables containing the data model entities** – database tables and two sets of roles granting read and write access to these database tables
2. HDI container for **read interfaces** containing the read interface artifacts – synonyms, grantors, and synonymconfig files for entities in HDI containers for tables, `hdbcds` views consuming the synonyms, DCL access control files for authorization checks for data privacy and protection (optional), and roles to allow access to the views by the reporting layer
3. HDI container for **write interfaces** containing the write interface artifacts – synonyms, grantors, and synonymconfig files for entities in HDI containers for tables, procedures consuming the synonyms, and roles to allow access to the procedures by the data load layer

You can also deploy the following HDI containers:

- HDI containers for **data load** – containing SDI flowgraphs to load data into the database tables using the generated write interface procedures

- HDI containers for **reporting** – containing reporting views (graphical calculation views) for the various reporting scenarios. These views can be used in various reporting applications and tools such as BO Analysis Excel to fetch data for reporting scenarios.

The following figure provides an overview of the HDI container structure:



4.1 Deploying SAP Financial Services Data Management

First Steps

- Download the **SAP PowerDesigner project file** from SAP Support Portal.
  To view the project files, place them in the same folder on your local machine and open them in your local SAP PowerDesigner installation. Double-click the project, CDM, and PDM to see their contents.
  The contents of the extension file can be seen on the *Extended Attributes* tab page of the data model objects (*Right-click* > *Properties* > *Extended Attributes*).
- Download the `zip` **files** from SAP Support Portal and install them as described in Installing and Upgrading SAP Financial Services Data Management [page 56].

Code Lists

You can access the code list values in the following ways:

- In SAP PowerDesigner: Open the CDM and search for a code list entity. You can find the standard values on the *Definition* tab page in the entity's properties.
- In SAP WebIDE: Find the HDI container in which the SAP Financial Services Data Management data model is deployed, go to *Tables* and search for a code list entity. Right-click the entity and choose either *Open Data* or *Generate SELECT Statement* and execute it.

You can extend these code lists as described in Adding Code List Values [page 124].

4.1.1 Deploying the Data Model to SAP HANA XS Advanced Runtime

You can use the SAP WebIDE for SAP HANA to create your initial data model from the template and to deploy the data model project for development and sandbox testing purposes. After this initial step, we recommend that you keep your version of the data model in a source control system such as Git. From the Git repository, other developers can check the data model out into their SAP WebIDE, it can be checked out locally to deploy it using the XS command line tool, and it can be checked out for productive deployment in a CI pipeline using MTA Archive Builder, "xs deploy", and automated tests.

Using SAP Web IDE for SAP HANA to create an initial data model project and build the data model

Prerequisite: You have completed the steps described in Installing and Upgrading SAP Financial Services Data Management [page 56]. You have enabled the FSDM extension in the *Features* section of SAP Web IDE.

1. Launch SAP Web IDE.
2. Choose *File* > *New* > *Project from Template* and select the *Financial Services Data Management Project*. You are prompted to enter a project name of your choice.
   If you want to extend the data model, follow the steps in the Extensibility Information [page 100].
3. To test, you can right-click the db module in the new project and choose *Build*.
4. Display the generated database tables in the database explorer in SAP Web IDE.
5. We recommend that you check your project into a version control system such as a Git repository so it can be versioned, shared with other developers, and used for subsequent deployments.
6. You can build your whole project in SAP WebIDE or check out the project files from your version control system and use the cloud-mta-build-tool locally or in a CI project. In both cases, an `mtar` file is generated.

Using SAP Web IDE for SAP HANA to add more extensions to the data model

1. Launch SAP Web IDE.
2. Select *File* > *Git* > *Clone Repository* to check out the repository created in the first step (multiple users can do this).
3. Change the extension part and follow the steps in the Extensibility Information [page 100].

4. Check in the changes to Git.

Using XS command line tool to deploy the data model project created in the step above

You deploy the artifact that you created in SAP WebIDE from the artifacts folder as follows:

1. Open the command line.
2. Navigate to the location of the `mtar` file created in step 6 above.
3. Enter **`xs login`**.
4. Enter **`xs target -o <organization> -s <space>`**.
5. Enter **`xs deploy <file_name>`**.
6. Optional: specify the schema in a MTA deployment extension file and use the `-e` option during deployment as described in What You Need to Know About the System Landscape [page 8].

We recommend that you automate these steps in a Jenkins pipeline after Git check-in. For more information, see Continuous Integration and Delivery Best Practices Guide.

Display the generated database tables in the database explorer in SAP Web IDE.

More Information

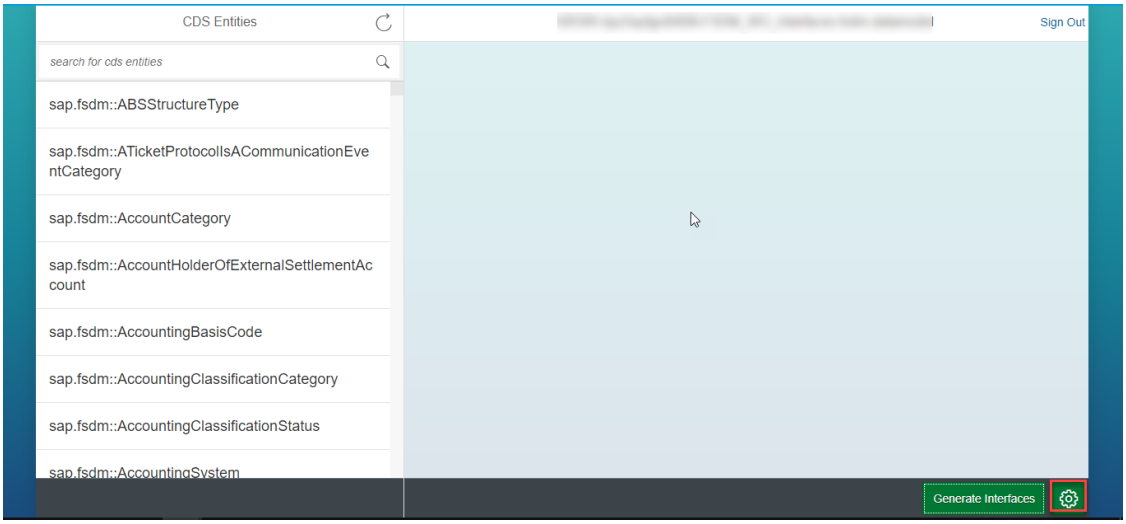
- cloud-mta-build-tool
- Deploy a Multi-Target Application (with Command-Line Tools)
- Continuous Integration and Delivery Best Practices Guide

4.1.2 Deploying the Generated Write or Read Interface to SAP HANA XS Advanced Runtime

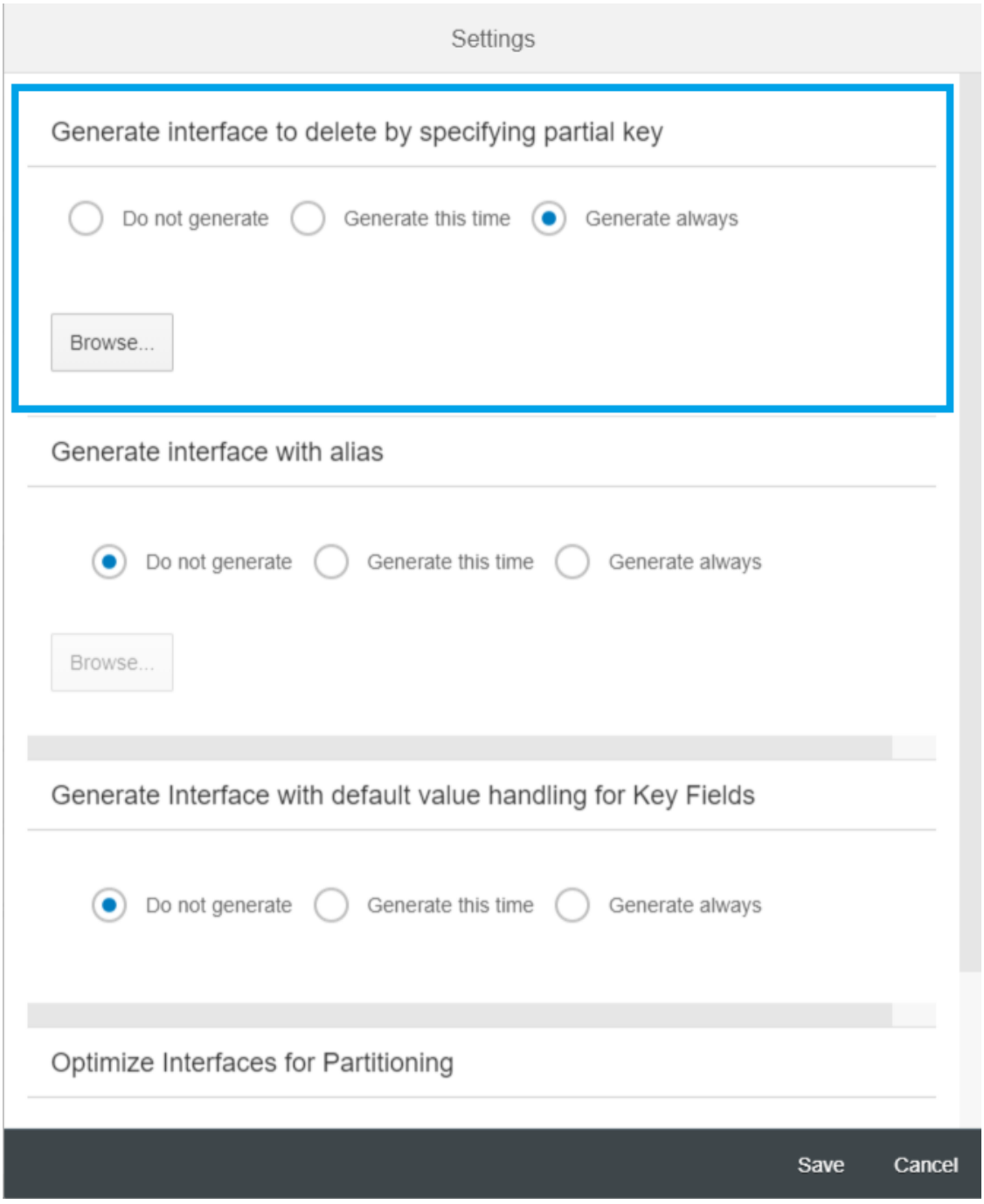
Generate the Interface Using the Interface Generator

1. Log on to the SAP Financial Services Data Management interface generator. You identified the URL in the process for Installing or Upgrading the Interface Generator [page 68].
   We recommend that you use the interface generator in Google Chrome or Mozilla Firefox. Make sure that you have disabled the pop-up blocker within the browser for the interface generator URL.
2. A list of all HDI containers available in the system is displayed on the interface generator UI. Specify the HDI container in which the data model is deployed to generate the corresponding write and read interfaces.
3. Select an entity from the list.

4. If you want to generate delete and erase procedures that can delete/erase records based on partial keys (subset of primary keys of a table), then you need to provide settings as below:
    1. Click on *Settings* on the right bottom corner.



    2. Choose one of the settings as seen in the screenshot below. Any setting that is saved is applicable to all users and HDI containers.



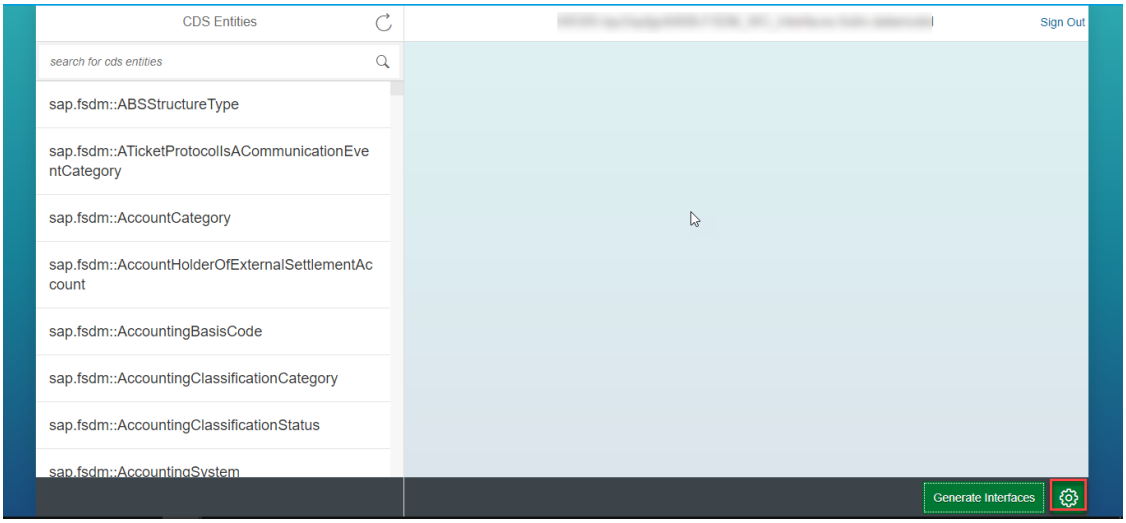
- **Do not generate**: The delete and erase procedures generated by interface generator contain logic to delete/erase based on full key specification. This is the default setting.
- **Generate this time**: The delete and erase procedures generated by interface generator contain logic to delete/erase based on partial key specification. Once you select this option, you can click on *Browse* and select the csv file containing partial key specification. The system validates the csv file input against the data model and provides errors with fields that are not part of the data model. If there are no errors, click *Save*. This setting is relevant for current session of interface generator. To save the global setting as *Do not generate*, click *Save*.

> ### Example
>
> Sample csv file specification for partial key delete
>
> If you want to delete/erase records from a table by specifying partial keys, create a csv file that is comma-separated and does not contain any header. The columns must be as follows:
>
> - The first column must specify the table with the namespace
> - The rest of the columns must be primary key fields of the table that you want to use for deletion/erasure.
>
> If you want to delete from table *AlternativeBusinessPartnerIdentification* based on *Authority*, *IDSystem* and *ASSOC_BusinessPartnerID.BusinessPartnerID*, then the csv content must be as follows without any header:
> *sap.fsdm::AlternativeBusinessPartnerIdentification,Authority,IDSystem,ASSOC_BusinessPartnerID.BusinessPartnerID*.
>
> If you also want to enable deletion based on *ASSOC_BusinessPartnerID. BusinessPartnerID* only, the csv must be as follows:
> *sap.fsdm::AlternativeBusinessPartnerIdentification,Authority,IDSystem,ASSOC_BusinessPartnerID.BusinessPartnerID*
> *sap.fsdm::AlternativeBusinessPartnerIdentification,ASSOC_BusinessPartnerID.BusinessPartnerID*
>
> You can also add more entries for tables similar to the above:
>
> *sap.fsdm::AlternativeBusinessPartnerIdentification,Authority,IDSystem,ASSOC_BusinessPartnerID.BusinessPartnerID*
>
> *sap.fsdm::AlternativeBusinessPartnerIdentification,ASSOC_BusinessPartnerID.BusinessPartnerID*
>
> *sap.fsdm::FinancialContract,IDSystem*

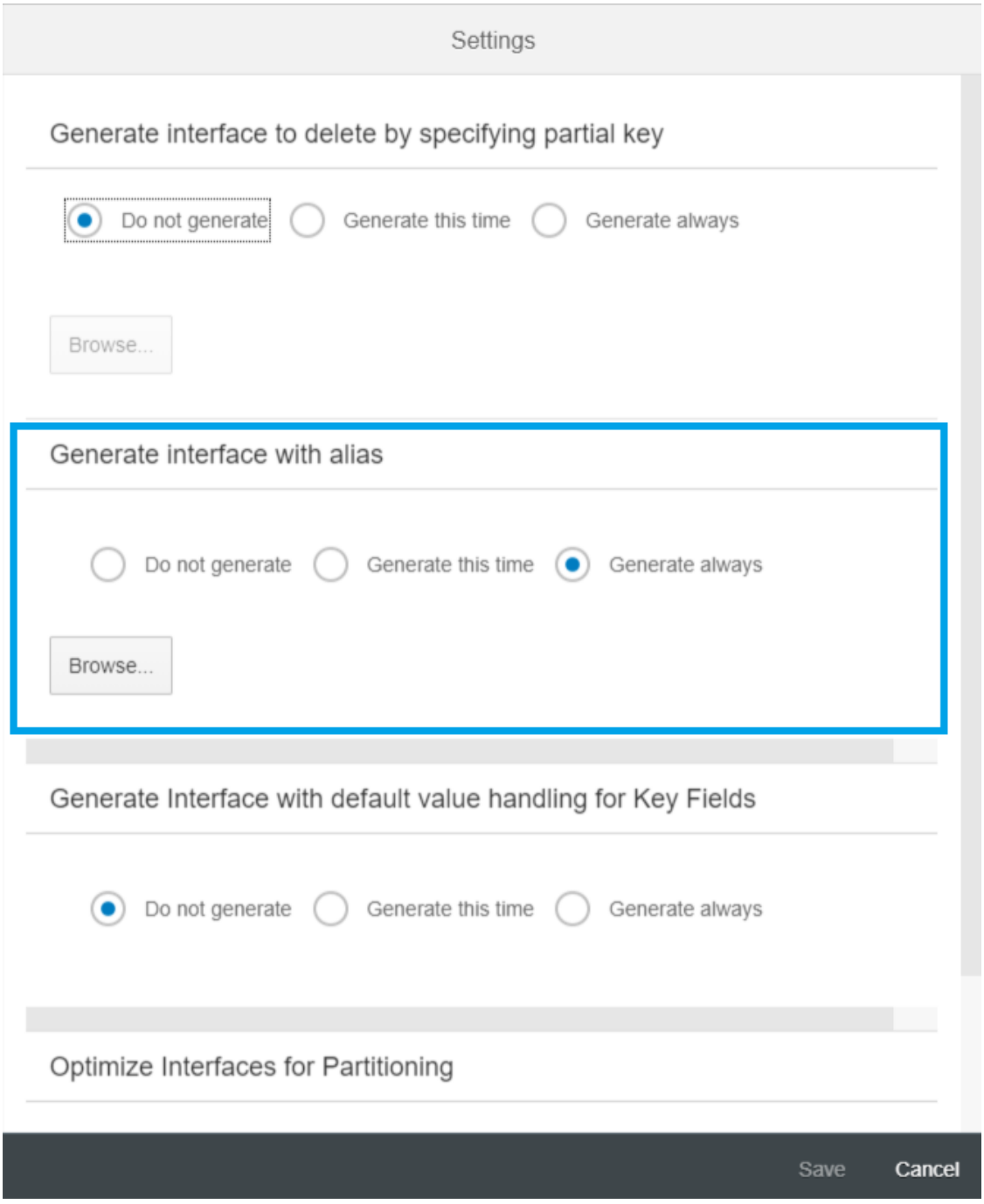
- **Generate always**: This setting works like **Generate this time**. In addition, when you click *Save*, the content of the csv file is saved for further generations.

3. Once the settings are saved and option *Generate this time* or *Generate always* is chosen, the interface files are generated with specified partial key combination in delete and erase procedures.

5. If you want to generate interfaces with an alias, you need to provide settings as below. Based on the settings provided, an extra zip file is generated that handles aliases.
   1. Click *Settings* on the right bottom corner.



2. Choose one of the settings as seen in the screenshot below. Any setting that is saved is applicable to all users and HDI containers.



- **Do not generate**: If this setting is selected, no additional zip file containing interfaces for aliasing is generated. This is the default setting.
- **Generate this time**: Once you select this option, you can click *Browse* and select the csv file containing the alias specification. The system validates the csv file input against the data model and provides errors with fields and associations that are not part of the data model. If there are no errors, click *Save*. This setting is relevant for the current session of the interface generator.
  A sample csv file specification for partial key deletion is described in the example below.

  > ✤ Example
  >
  > Sample csv file specification for alias

If you want to generate an interface with an alias, create a csv file that is comma separated and does not contain any header. The columns must be as follows:

- Deprecated: Indicates whether the attribute is deprecated. Possible values are as follows:
  - *TRUE: Indicates standard data model deprecation.*
  - *FALSE: Replacing extension of data model with standard data model.*
- Type – Indicates the type of extension. Possible values for this field are as follows.
  - *Attribute*: Use this value if you are replacing an extended attribute with a standard attribute in a standard entity.
    o *EntityAttribute*: Use this value if you are replacing an extended attribute with a standard attribute in an extended entity. Here it is assumed that the entire extended entity is replaced by a standard entity.
    o *Relationship*: Use this value if you are replacing an extended attribute created for association with a standard attribute for the corresponding association in a standard entity.
    o *EntityRelationship*: Use this value if you are replacing an extended attribute created for association with a standard attribute for the corresponding association in an extended entity. Here it is assumed that the entire extended entity is replaced by a standard entity.
    o *Association*: Use this value if you are replacing an extended association with a standard association in a standard entity.
    o *EntityAssociation*: Use this value if you are replacing an extended association with a standard association in an extended entity. Here it is assumed that the entire extended entity is replaced by a standard entity.
- *ExtendedTableName*: If the type is Attribute, Relationship, or Association, the ExtendedTableName is the same as StandardTableName. If the type is EntityAttribute, EntityRelationship or EntityAssociation, the ExtendedTableName represents the table that was extended and is being replaced by a standard entity.
  ExtendedArtifact: Indicates the extended artifact (attribute or association) replaced by a standard artifact (attribute or association) .
  StandardTableName
  StandardArtifact
  DataType: Data type of the attribute that you are replacing. This field needs to be filled if the type is Attribute, EntityAttribute, Relationship, or EntityRelationship.
  Length/Precision: Length of the attribute that you are replacing if there are string data types and precision if there are fixed-point decimals like DECIMAL(34.6). This field needs to be filled if the type is Attribute, EntityAttribute, Relationship, or EntityRelationship.
  Scale: It is relevant only for fixed-point decimals like DECIMAL(34.6) of the attribute that you are replacing. This field needs to be filled if the type is Attribute, EntityAttribute, Relationship, or EntityRelationship.
  TargetEntity: Target entity of the association. This field needs to be filled if the type is Association or EntityAssociation.

Example:

The examples below are mentioned based on the extension concept explained under Extensibility Information [page 100]. The assumption is that all attributes/tables are replaced in later releases of SAP Financial Services Data Management with standard attributes/tables with the same semantics.

If you want to replace the extended attributes *C_Length* and *C_EngineManufacturer* in table the *sap.fsdm::PhysicalAsset* with *Length* and *EngineManufacturer*, the csv content must be as below:

FALSE,Attribute,PhysicalAsset,C_Length,PhysicalAsset,Length,DECIMAL,,,

FALSE,Attribute,PhysicalAsset,C_EngineManufacturer,PhysicalAsset,EngineManufacturer,NVARCHAR,256,,

If you want to replace entity *C_Lifeboat* with *Lifeboat*, the csv content should be as below.

FALSE,EntityAttribute,C_Lifeboat,C_LifeboatID,Lifeboat,LifeboatID,INTEGER,,,

FALSE,EntityAttribute,C_Lifeboat,C_CapacityOfPlaces,Lifeboat,CapacityOfPlaces,INTEGER,,,

If you also want to replace the relationship *C_LifeboatsOfShip* (association name in PDM is *ASSOC_PhysicalAsset*) in entity *C_Lifeboat* with *LifeboatsOfShip* (association name in PDM is *_PhysicalAsset*), then the csv content should be as below:

FALSE,EntityAttribute,C_Lifeboat,C_LifeboatID,Lifeboat,LifeboatID,INTEGER,,,

FALSE,EntityAttribute,C_Lifeboat,C_CapacityOfPlaces,Lifeboat,CapacityOfPlaces,INTEGER,,,

FALSE,EntityRelationship,C_Lifeboat,PhyscialAssetID,Lifeboat,_PhysicalAsset.PhysicalAssetID,INTEGER,,,

FALSE,EntityAssociation,C_Lifeboat,ASSOC_PhysicalAsset,Lifeboat,_PhysicalAsset,,,,PhysicalAsset

If you want to replace n:m relationship *C_Motorcycle_compatible_with_Truck* with *Motorcycle_compatible_with_Truck*, then the csv content should be as below:

FALSE,EntityRelationship,C_Motorcycle_Truck_compatibility,C_MotorcyclePhysicalAssetID,Motorcycle_Truck_compatibility,_Motorcycle.PhysicalAssetID,NVARCHAR,168,,

FALSE,EntityRelationship,C_Motorcycle_Truck_compatibility,C_TruckPhysicalAssetID,Motorcycle_Truck_compatibility,_Truck.PhysicalAssetID,NVARCHAR,168,,

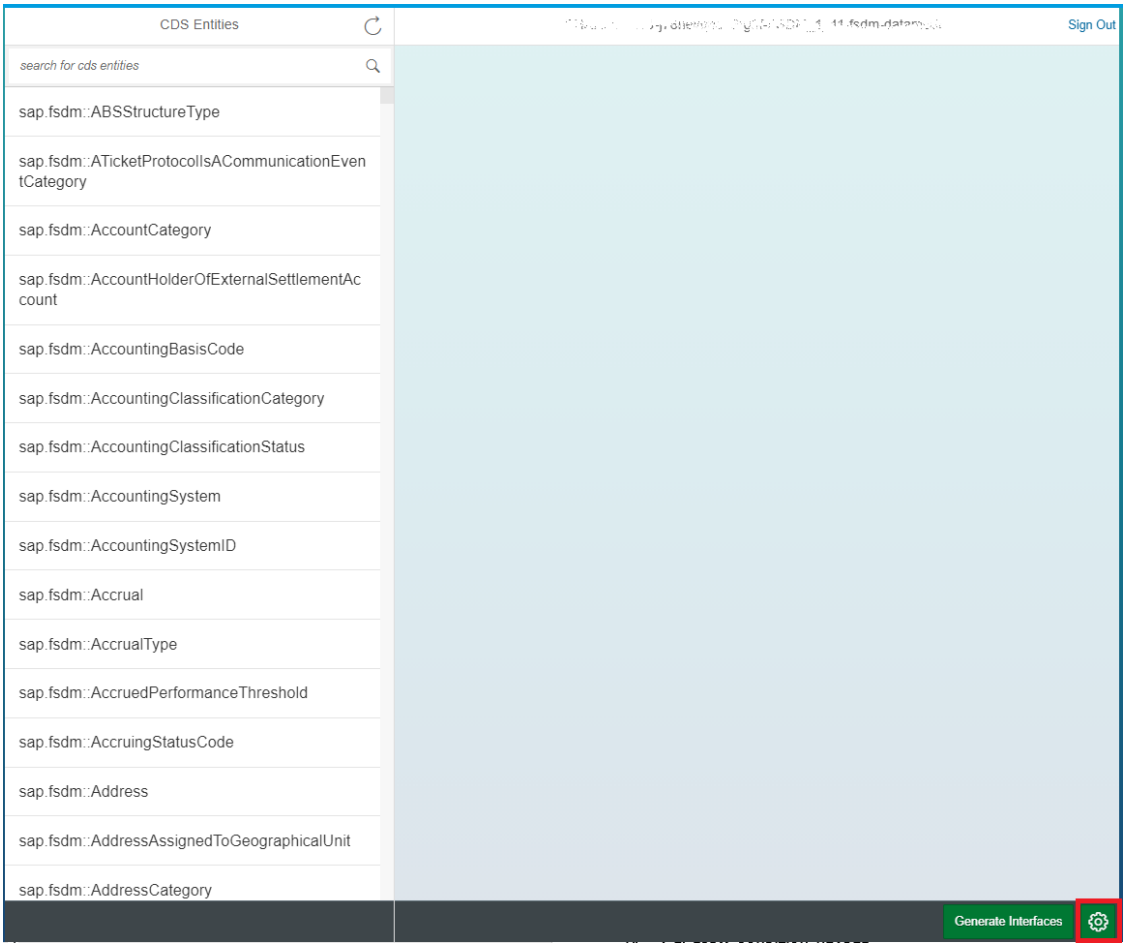
FALSE,EntityAssociation,C_Motorcycle_Truck_compatibility,_C_Motorcycle,Motorcycle_Truck_compatibility,_Motorcycle,,,,PhysicalAsset

FALSE,EntityAssociation,C_Motorcycle_Truck_compatibility,_C_Truck,Motorcycle_Truck_compatibility,_Truck,,,,PhysicalAsset

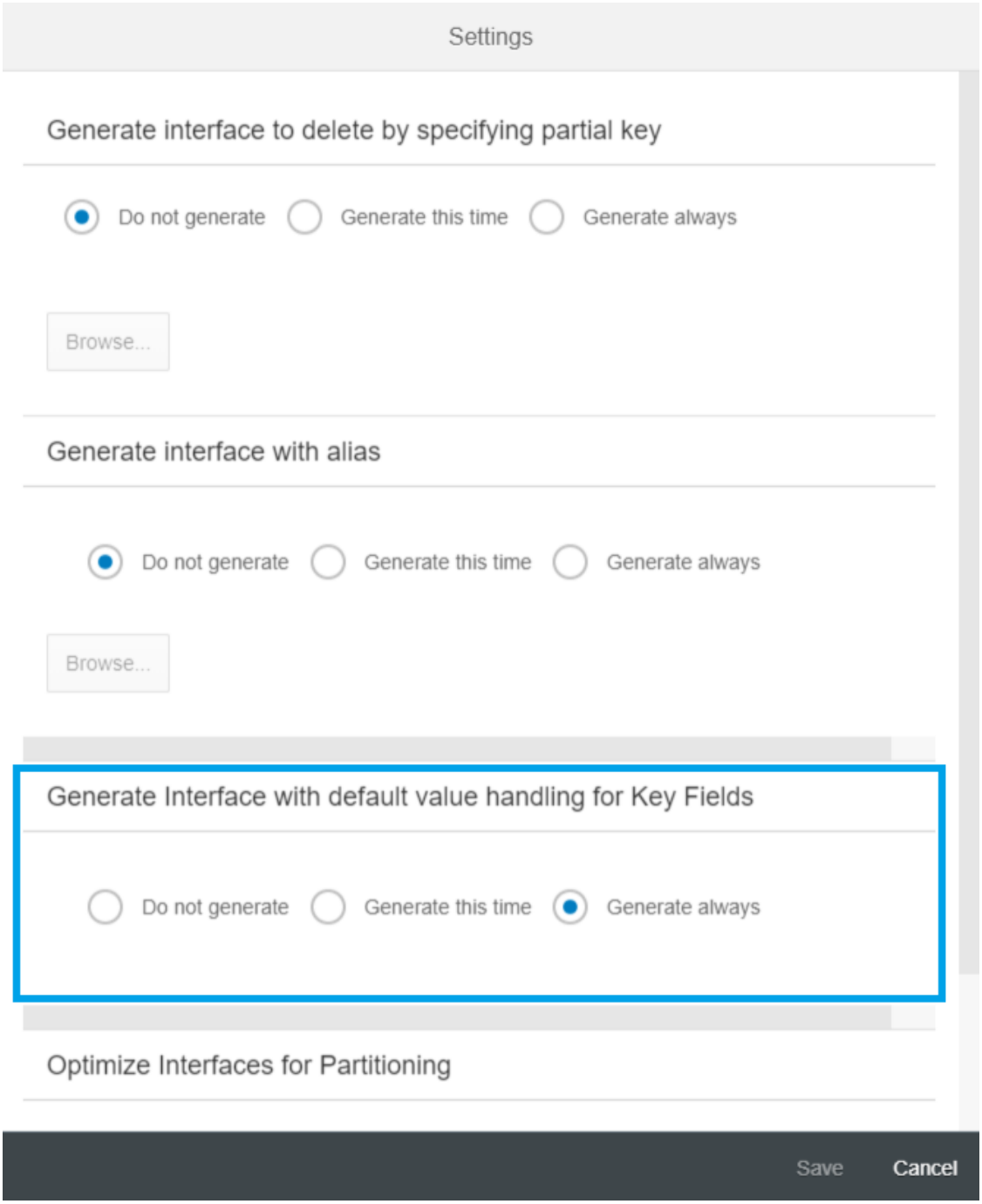
- **Generate always**: This setting works like **Generate this time**. In addition, when you click *Save*, the content of the csv file is saved for further generations.

3. Once the settings are saved and option *Generate this time* or *Generate always* is chosen, the interface generator generates two zip files when you click *Generate* - one for the traditional interfaces and another for the interfaces with an alias.

6. If you want to generate write interfaces without handling for null values in key fields, you need to provide settings as below:
   1. Click on *Settings* on the right bottom corner.


CDS Entities
search for cds entities
sap.fsdm::ABSStructureType
sap.fsdm::ATicketProtocolIsACommunicationEventCategory
sap.fsdm::AccountCategory
sap.fsdm::AccountHolderOfExternalSettlementAccount
sap.fsdm::AccountingBasisCode
sap.fsdm::AccountingClassificationCategory
sap.fsdm::AccountingClassificationStatus
sap.fsdm::AccountingSystem
sap.fsdm::AccountingSystemID
sap.fsdm::Accrual
sap.fsdm::AccrualType
sap.fsdm::AccruedPerformanceThreshold
sap.fsdm::AccruingStatusCode
sap.fsdm::Address
sap.fsdm::AddressAssignedToGeographicalUnit
sap.fsdm::AddressCategory
Sign Out
Generate Interfaces

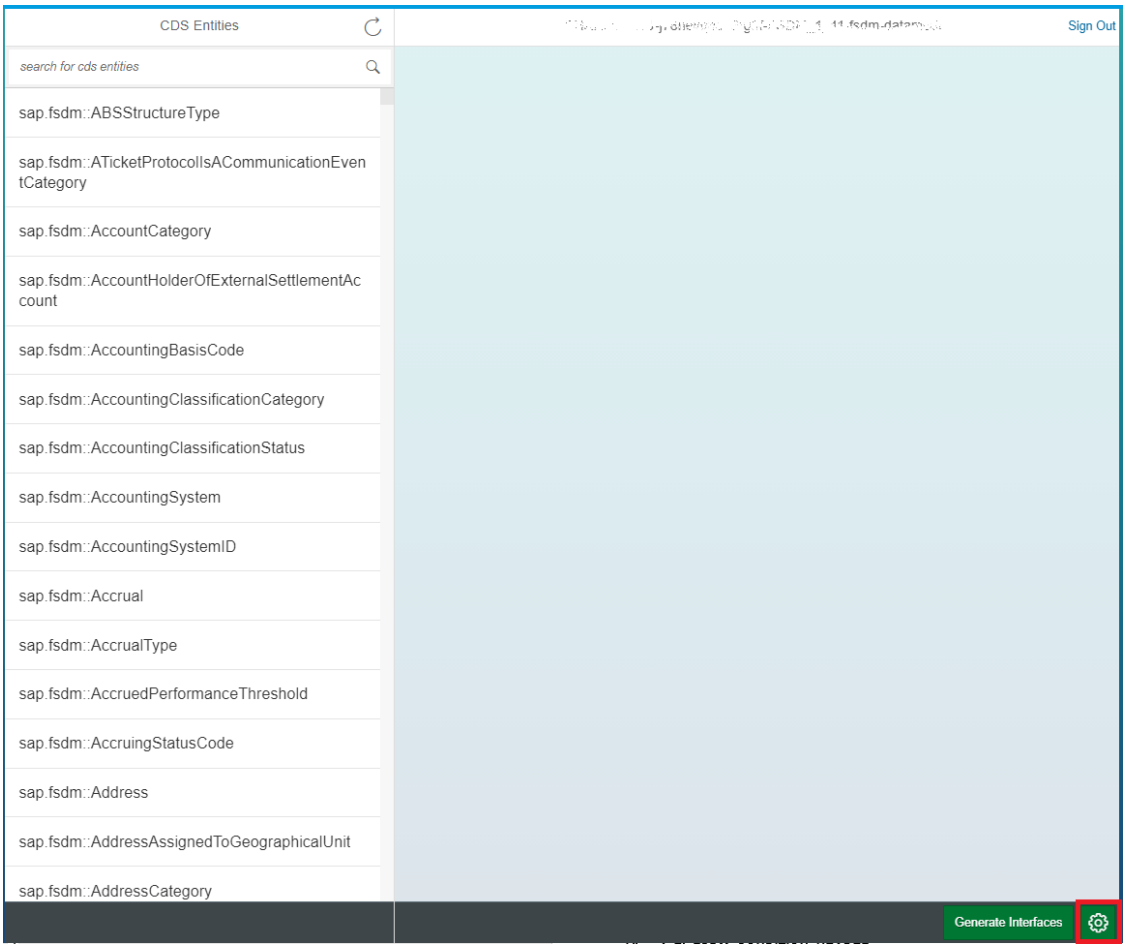
2. Choose one of the settings as seen in the screenshot below. Any setting that is saved is applicable to all users and HDI Containers.



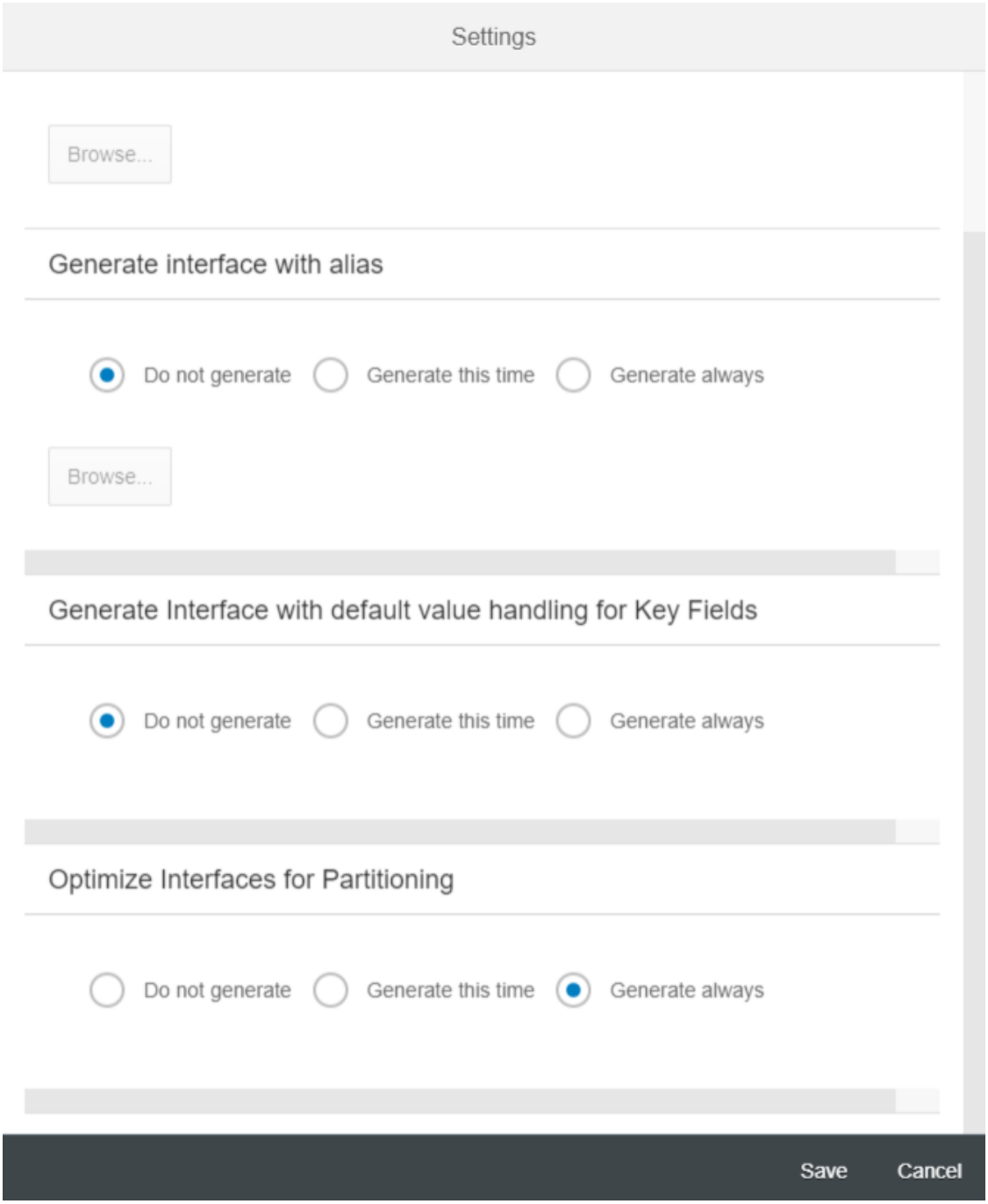
- **Do not generate**: The generated load procedures don't have any logic to handle null values for key fields. When calling these procedures in the input data if any of the key field is null an exception is raised.
- **Generate this time**: This setting works like *Generate always*. Choosing this option is relevant for current session of interface generator.
- **Generate always**: The generated load procedures contains logic to handle null values for key fields. This is the default setting.

3. Once the settings are saved and option *Generate this time* or *Generate always* is chosen, the interface files are generated with logic to handle null values for key fields in load procedures.

7. If you want to optimize write interfaces for tables partitioned using *Business Validity Period* fields (*BusinessValidFrom* or *BusinessValidTo*), you need to provide settings as below:
   1. Click on *Settings* on the right bottom corner.



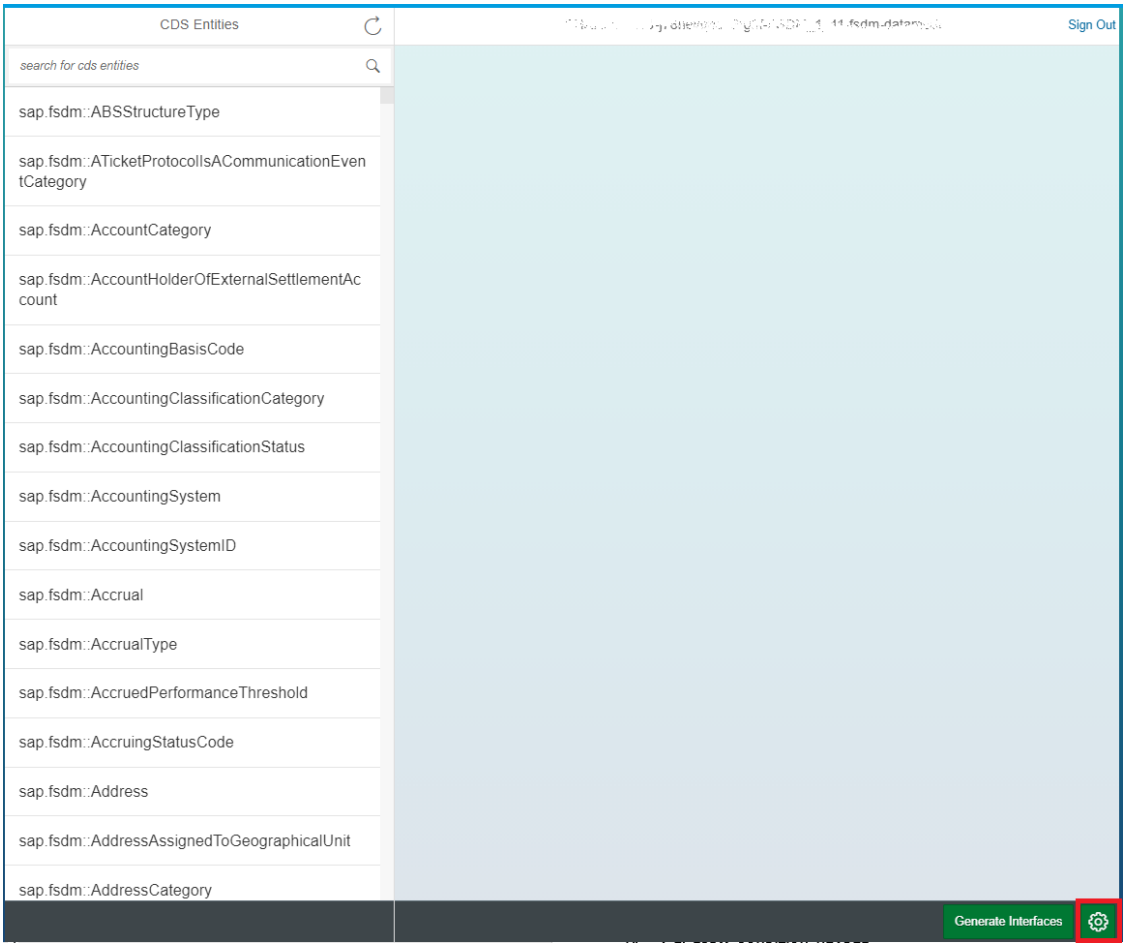
2. Choose one of the settings as seen in the screenshot below. Any setting that is saved is applicable to all users and HDI containers.



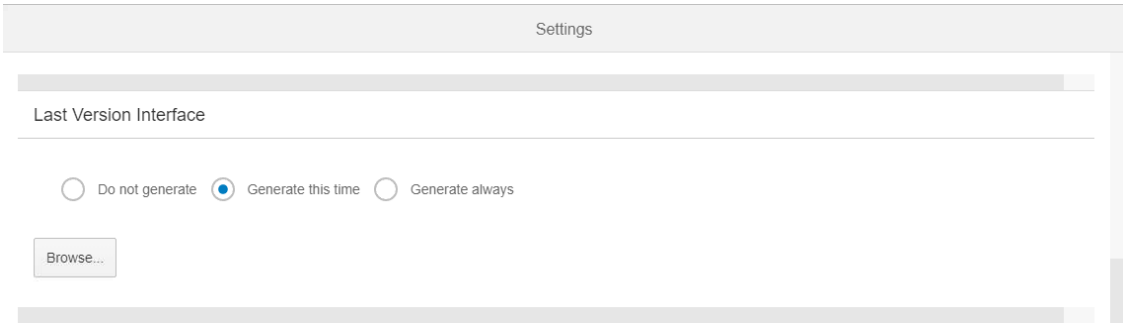
- **Do not generate**: The generated load procedures don't have additional logic to optimize partition access.
- **Generate this time**: This setting works like *Generate always*. Choosing this option is relevant for current session of interface generator.
- **Generate always**: The generated load procedures contains logic to optimally access partitions for tables where partitioning is done using Business Validity Period fields (BusinessValidFrom or BusinessValidTo).

3. Once the settings are saved and option *Generate this time* or *Generate always* is chosen, the load procedures are generated with logic to optimally access partitions.

8. If you want to generate additional SQL views which can fetch records closest to input *Business Date* and *System Timestamp* you need to provide settings as below:
   1. Click on *Settings* on the right bottom corner.



   2. Choose one of the settings as seen in the screenshot below. Any setting that is saved is applicable to all users and HDI containers.



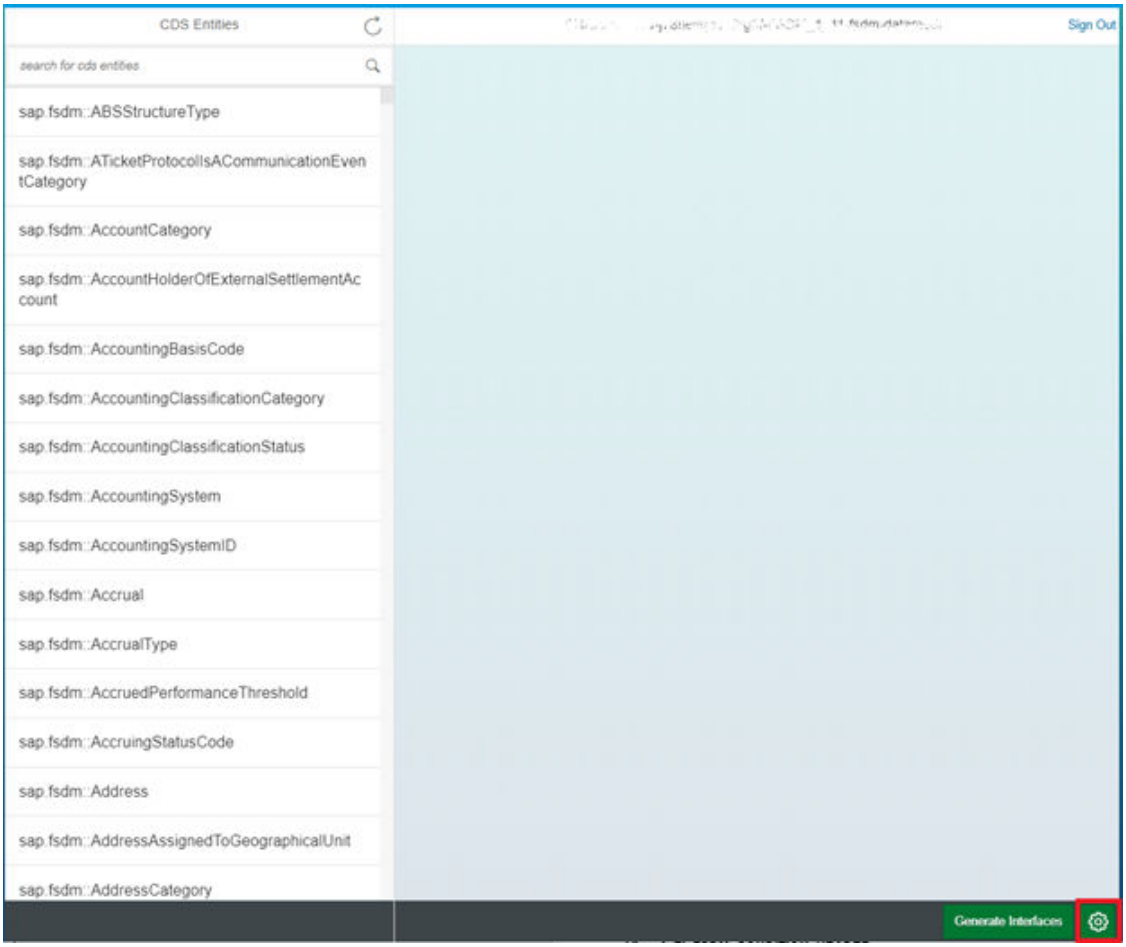
   - **Do not generate**: Additional SQL views for last version handling aren't generated. This is the default setting.
   - **Generate this time**: This setting works like *Generate always*. Choosing this option is relevant for current session of interface generator.

- **Generate always**: If this option is selected then additional SQL views for the last version handling are generated. On selecting this option you can upload a csv file in specified format the file uploader (Browse action).

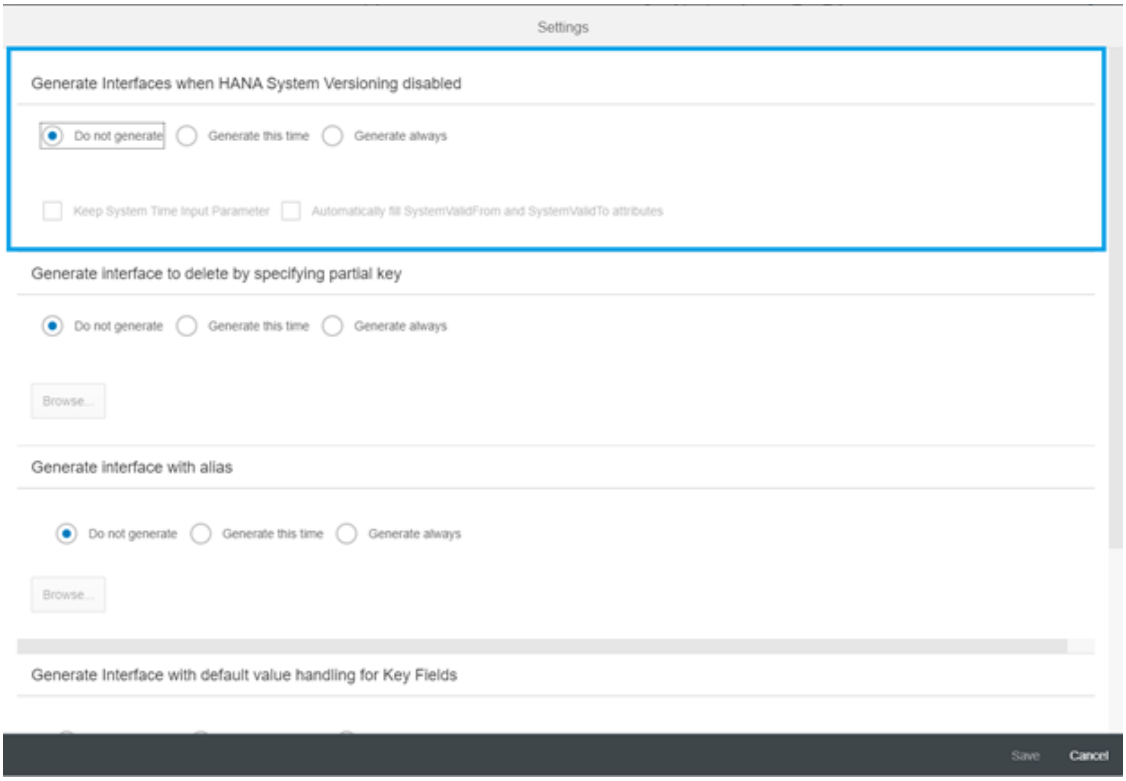
3. Once the settings are saved and option *Generate this time* or *Generate always* is chosen, additional SQL views for last version handling are generated.
4. Each row of csv file represents an entity and optionally attributes (for example semantic key attributes) to be used for determining versions. If attributes aren't specified for an entity, the complete semantic key is considered for version determination.
   Example csv file content for generating last version interface:
   - *BusinessPartner*
   - *FinancialContract*
   - *ReferenceRate,ReferenceRateID,ReferenceRateCurrency*

   In this scenario there will be 3 additional SQL views generated, one each for business partner, financial contract and reference rate entities. For business partner, the semantic key attribute *BusinessPartnerID* will be used for version determination as explicit list of attributes to consider it is not specified in the csv file. Similarly for financial contract entity *FinancialContractID* and *IDSystem* attributes are used in SQL view for version determination. While for reference rate entity explicitly specified attributes *ReferenceRateID* and *ReferenceRateCurrency* will be used for version determination.

9. SAP Financial Services Data Management uses SAP HANA System Versioning to keep track of changes done to the dataset. This means the system creates different versions based on system timestamp when either loading, updating or deleting data in current table in case of write interfaces, or access data from history table in case of read interfaces. This produces a large data volume over the time. To reduce this data volume, you can optionally disable SAP HANA System Versioning for a set of entities or for all the entities in case there is no auditing or traceability requirement.
   If you have disabled SAP HANA System Versioning, but want to generate interfaces for those entities, then you need to provide settings as below:
   1. Click on *Settings* on the right bottom corner.



2. Choose one of the settings as seen in the screenshot below. Any setting that is saved is applicable to all users and HDI containers. Please keep in mind that this selection is only applied if SAP HANA System Versioning is disabled for one or more entities.



- **Do not generate**: If this setting is selected and SAP HANA System Versioning is disabled for one or more entities then *Generate Interfaces* action would not generate interfaces and display error message. This is the default setting.
- **Generate this time**: This setting works like *Generate always*. Choosing this option is relevant for current session of interface generator.
- **Generate always**: The generated interfaces do not handle system versioning and history table of those entities where versioning is disabled.

3. Once the settings are saved *Generate always* is chosen, the generated interfaces for entities where SAP HANA System Versioning is disabled would not create versions based on system timestamp when loading, updating or deleting data in current table (in case of write interfaces) or access data from history table (in case of read interfaces).

10. Click *Generate Interfaces* to download a `zip` file with the generated interface.

> **Note**
>
> For more information about read and write interfaces, see *Interface generator for read/write interfaces and roles* under Operations Information [page 71].

You can now **deploy** the interfaces using SAP Web IDE or the XS command line tool.

Deploy the Interface Using SAP Web IDE for SAP HANA

1. Import and deploy the procedures/views in SAP Web IDE:
   1. Right-click the workspace and choose *Import From File System*.

2. Import the zip file under *db* > *src* > *interface*.
3. Build at DB level.

Display the generated database procedures/views in the database explorer in SAP Web IDE.

4.2 Data Provisioning, Loading, and Reporting

After you have successfully deployed the CDS definitions, you can write or retrieve data in or from tables using only the generated procedures and views provided by the interface generator.

For each data model entity, the following set of procedures is generated:

- Load procedures
- Delete procedures
- Read-only procedures

The generated stored procedures can be used in SAP HANA smart data integration (SDI) flowgraphs to load data into the SAP Financial Services Data Management data model.

> → Recommendation
>
> We recommend that you deploy the SDI flowgraphs as well as reporting views in a separate container to the HDI containers in which the database tables and their corresponding read and write interfaces are deployed. For more information, see User Administration and Authorization [page 28].

> → Recommendation
>
> We recommend that you use plain SQL views to connect with the FS Data Management data model through the read interfaces. If you plan to use analytical tools that can consume only calculation views, use a thin calculation view layer on top of your plain SQL views.
>
> If you need specific features that only calculation views provide, have these as high as possible in the view stack. Always use calculation views running in the SQL engine.

Procedures for Data Provisioning

Load Procedures

Load procedures are used to load data into FS Data Management entities. They behave differently depending on which kind of versioning is used.

- **Entities with one-dimensional versioning**:
  These entities are versioned in system time, which means that the system fills the system timestamp fields of the data records. The data load can be an initial or delta load.
  - Initial load: Data is loaded into FSDM tables whereby the system updates the system timestamp fields (`SystemValidFrom` and `SystemValidTo`).

- Delta load: The previous version of the data records is deleted from the current FSDM table and moved to the FSDM history table while the current table is updated.

The system treats any change in the key fields of a data record as a new record: A new entry is created in the current table, the history table remains untouched.

- **Entities with two-dimensional versioning**:
  These entities are versioned in business and system time: The business time fields (`BusinessValidFrom` and `BusinessValidTo`) must be provided by the source, whereas the system timestamp fields are filled by the system.

  > **Note**
  >
  > To load the data, you must provide the two business time fields for any data record. This also applies if only one business time record in a set of records is missing.

- **Non-versioned entities:**
  When you use load procedures inside a flowgraph, the flowgraph must be procedure type.

Delete Procedures

You can delete data from FSDM tables in the following ways:

- **Logical delete procedure**:
  Performs a logical deletion of data. For this, you must specify the fields that are to be deleted from the FSDM tables. These serve as input for the deletion, which deletes the records and moves them to the FSDM history table.
- **Physical delete procedure/erase procedure**:
  Performs a physical deletion of data in both the current and history tables. For this, you must specify the key fields of the data records to be deleted.
- **Deletion by specifying a partial key**:
  By default, logical delete and physical delete/erase procedures generated by interface generator expects full primary key of the table to be passed. To delete records based on partially specified primary keys, please refer to Deploying the Generated Write or Read Interface to SAP HANA XS Advanced Runtime [page 77].

Read-only Procedures

Read-only procedures are used when the load/delete operation is expected to be executed as batch or transaction. The procedure does not contain any DML statements. It gives only outputs, which can be mapped to the FSDM tables, as these procedures do not directly update the FSDM tables.

When you are using read-only procedures inside a flowgraph, the type of flowgraph must be transactional.

Procedures for FSDM List Entities

List entities are entities where a set of records are always versioned together such as entries of a payment schedule. These are marked as a list in the CDM using an extended attribute.

The procedures process the incoming data records by identifying the lists using a subordinate key. They are available as load, delete and erase procedures together with their read-only counterparts.

Procedures Specific to Non-versioned FSDM Entities

There are procedures that are generated specifically for FSDM entities that are not versioned. They do not update existing records, but insert and update separately.

- Insert procedure: Inserts new records. Ignores any corrections to existing records
- Update Procedure: Updates only existing records but ignores any new records

The system generates the read-only counterparts of these procedures.

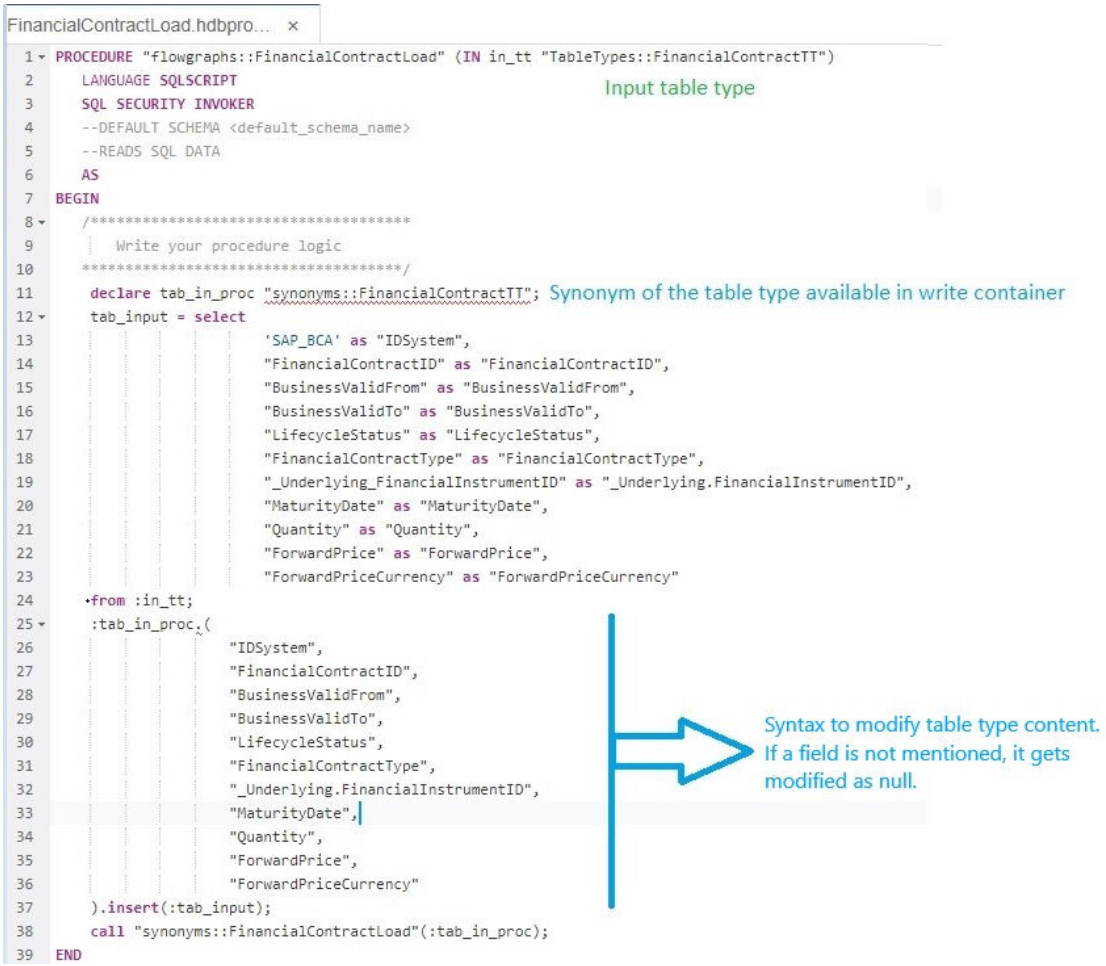
Consuming Load and Read-Only Procedures

To consume load and read-only procedures in SAP HANA Smart Data Integration (SDI), follow the steps:

1. Create a table type with the fields that are relevant for loading into an FSDM entity.

```
FinancialContractTT.hdbtablety... ×
1  TYPE "TableTypes::FinancialContractTT" AS TABLE (
2  "FinancialContractID" NVARCHAR (128),
3  "BusinessValidFrom" DATE ,
4  "BusinessValidTo" DATE ,
5  "LifecycleStatus" NVARCHAR (40),
6  "FinancialContractType" NVARCHAR (40),
7  "MaturityDate" DATE ,
8  "Quantity" DECIMAL (34,6),
9  "ForwardPrice" DECIMAL (34,6),
10 "ForwardPriceCurrency" NVARCHAR (3),
11 "_Underlying_FinancialInstrumentID" NVARCHAR(128)
12 )
```

2. Create a synonym for the table type of the entity into which you want to insert data. The table types are already exposed in the relevant roles in the write interface. The naming convention of the table type is `sap.fsdm.tabletypes::<FSDMEntityName>TT`.
3. Create a procedure:
    1. Enter a parameter with a table type as created in step 1.
    2. Declare a variable of the synonym table type mentioned in step 2.
    3. Use the syntax to modify the table type. For more information, see Modifying the Content of Table Variables.




```
PROCEDURE "flowgraphs::FinancialContractLoad" (IN in_tt "TableTypes::FinancialContractTT")
   LANGUAGE SQLSCRIPT
   SQL SECURITY INVOKER
   --DEFAULT SCHEMA <default_schema_name>
   --READS SQL DATA
   AS
BEGIN
   /*************************************
      Write your procedure logic
   *************************************/
    declare tab_in_proc "synonyms::FinancialContractTT";
    tab_input = select
                    'SAP_BCA' as "IDSystem",
                    "FinancialContractID" as "FinancialContractID",
                    "BusinessValidFrom" as "BusinessValidFrom",
                    "BusinessValidTo" as "BusinessValidTo",
                    "LifecycleStatus" as "LifecycleStatus",
                    "FinancialContractType" as "FinancialContractType",
                    "_Underlying_FinancialInstrumentID" as "_Underlying.FinancialInstrumentID",
                    "MaturityDate" as "MaturityDate",
                    "Quantity" as "Quantity",
                    "ForwardPrice" as "ForwardPrice",
                    "ForwardPriceCurrency" as "ForwardPriceCurrency"
    from :in_tt;
    :tab_in_proc.(
                "IDSystem",
                "FinancialContractID",
                "BusinessValidFrom",
                "BusinessValidTo",
                "LifecycleStatus",
                "FinancialContractType",
                "_Underlying.FinancialInstrumentID",
                "MaturityDate",
                "Quantity",
                "ForwardPrice",
                "ForwardPriceCurrency"
    ).insert(:tab_input);
    call "synonyms::FinancialContractLoad"(:tab_in_proc);
END
```

4. Consume this procedure in an SDI flowgraph.

Reporting

The interface generator generates views that handle versioning along with the required roles for cross-container access to these views. There is one view of the current data and a union view that includes historical data. Technically, the generator produces SQL views with input parameters for bi-temporal versioning. These views can be consumed by SAP HANA calculation views for data analysis or reporting.

4.3 Partitioning

The partitioning feature of SAP HANA splits column-store tables horizontally into disjunctive subtables or partitions. In this way, large tables can be broken down into smaller, more manageable parts.

Partitioning is relevant for database tables with more than a few hundred million records. In such cases, tables should be partitioned, ideally with a maximum of 300 million records per partition. There is a theoretical limitation of about two billion records per partition (or unpartitioned tables).

For a full discussion on the advantages of partitioning and the different partitioning options available, see the SAP HANA Administration Guide. We recommend that you familiarize yourself with the following topics:

- Table Partitioning
- ALTER TABLE Statement (Data Definition) → `<partition_clauses>`

Partitioning Using SQL Compared with Partitioning Using HDI Extensions

The data model is usually deployed identically on different SAP HANA systems in the system landscape, for example, development, test, and productive systems. By contrast, partitioning can and needs to be different in different SAP HANA systems depending on the data in these systems. Even within a development system, data modelers may work with a local unpartitioned version of the data model while the same data model exists in the same SAP HANA database with partitioning for testing.

To choose a suitable partitioning scheme, see Designing Partitions.

Partitioning information can be provided as extensions to the data model and be deployed using HDI.

- **CDS Extensions**: You can provide partitioning information as CDS extensions to the data model and deploy it through HDI. Since partitioning is likely to be different in different systems and must be adapted as more data is loaded, we recommend that you use the SQL-based approach.
- **SQL**: You can partition using SQL and SQL-based tools based on a deployed data model. You need to define an HDI extension using the key word `partition by keeping existing layout;` for this table. Extend the relevant entities with the following CDS command and then proceed with the partitioning using SQL:

  Sample Code

  ```
  extend entity fc with technical configuration {
    PARTITION BY KEEPING EXISTING LAYOUT;
  }
  ```

Single-Level Partitioning

It may be useful to start with single-level partitioning using RANGE partitioning on a time dimension (such as a suitable date column). This means that certain partitions with old data have only infrequent updates and that these partitions may not have to be considered in queries.

Sample Code

```
PARTITION BY RANGE ( YEAR("BusinessValidTo"))
(PARTITION '2012'<= VALUES < '2015', PARTITION '2015' <= VALUES < '2019',
PARTITION OTHERS);
```

Two-Level Partitioning

You can extend the single-level partitioning scheme to two levels using parts of the semantic key to divide the partitions described above. The date-based partitioning approach would be the second level in this case.

1. **HANA Native Hash-Range**
   If you use a hash on the first level, SAP HANA ensures that the data is distributed according to the defined hash key. You only need to specify the number of partitions. This is a simple approach where part of the data analysis is left to the database.

   Sample Code

   ```
   PARTITION BY HASH("FinancialContractID") PARTITIONS 5,
   RANGE ( YEAR("BusinessValidTo"))
   (PARTITION '2012'<= VALUES < '2015', PARTITION '2015' <= VALUES < '2019',
   PARTITION OTHERS);
   ```

2. **Range-Range on Semantic Keys**
   This has similar benefits to the SAP HANA native hash-range but allows for a more granular management of data. For example, if you want to keep data that is distributed over two tables in a parent child relationship and use the same partitioning key, you can analyze the larger child table and then partition the data in the parent table so that the data in the child table is evenly distributed between partitions. Using the result of such an analysis, you can precisely define the partitions as needed. In this way it is easier to adapt to changing data patterns in dependent tables, but it requires more work doing data analysis.

   Sample Code

   ```
   PARTITION BY RANGE("FinancialContractID")
   (PARTITION 'A' <= VALUES < 'F',PARTITION 'F' <= VALUES <= 'Z' , PARTITION
   OTHERS ),
                  RANGE( year("BusinessValidTo") )
                  (PARTITION '2012' <= VALUES < '2015',PARTITION '2015' <=
   VALUES < '2019' , PARTITION OTHERS );
   ```

3. **Synthetic Partitioning Approach**
   This is a **logical hash range** based on the SAP HANA range-range.
   To use this approach, the data model contains the `PARTITION_KEY` key, which manages the distribution of the data in the database. This key contains a number derived from a hash algorithm. This approach can combine the advantages of the SAP HANA native hash-range approach while allowing more control over data management. It has higher costs in terms of memory and needs to be more carefully managed. This approach has particular strengths in combination with **SAP S/4HANA for financial products subledger**. For more information, see SAP Note 2770223.

   Sample Code

   ```
       PARTITION BY RANGE("PARTITION_KEY")
       (PARTITION '1' <= VALUES < '357913942',PARTITION '357913942' <= VALUES
   < '715827884' , PARTITION OTHERS ),
                  RANGE( year("BusinessValidFrom") )
       (PARTITION '2012' <= VALUES < '2015',PARTITION '2015' <= VALUES <
   '2019' , PARTITION OTHERS );
   ```

Current Tables and History Tables

The current table and history table of a system-versioned table are independent. You can choose different partitioning schemes for each table.

5 Extensibility Information

To adapt the unified data model delivered with SAP Financial Services Data Management according to your business needs, you have different options. For example you can add:

- Data fields, as described in Adding Data Fields [page 113]
- Entities, as described in Adding Entities [page 114]
- Relationships, as described in Adding Relationships [page 123]
- Code list values, as described in Adding Code List Values [page 124]

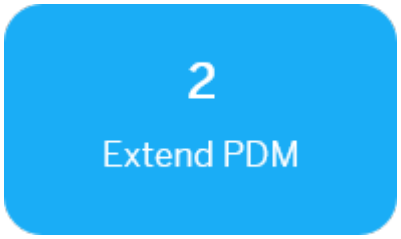
Examples

The extensibility documentation is based on a coherent example that shows, how you can extend the data model with new fields, new entities, or new relationships. A use case for this example is that some boat insurers provide hull insurance that cover physical damage of a boat and may include machinery, sails, furniture, outboard motors and tenders, for example. Tenders are not always covered automatically. Specific details about a tender such as size or engine manufacturer are required by some insurers. You can, therefore, add this details of a ship to the SAP Financial Services Data Management data model with extending the entity `ship`.

1. Additional data fields:
   This example shows, how you can extend the entity `ship` by adding your own attributes `C_EngineManufacturer` and `C_Length`, as some insurers require these information.
2. Additional entities:
   This example shows how you can add your own entity `C_Tender` with new attributes `C_ID` and `C_Capacity`, to cover lifeboats or day boats that belong to a ship.
3. Additional relationships:
   This example is based on the second example and shows, how you can create a relationship between the `Ship` and your new entity `C_Tender`.

5.1 Extensibility Steps

To extend the data model, please follow these steps:

	Extend the conceptual data model in SAP PowerDesigner to ensure that you keep lineage between the conceptual data model (CDM), the physical data model (PDM), and the SAP HANA CDS model that is deployed.
	Regenerate the physical data model in SAP PowerDesigner so that the new fields and entities appear there.
	Export the new CDS definition files from SAP PowerDesigner and deploy them on SAP HANA.

Before You Start

Before you start to extend the data model, please consider the following restrictions:

Do not change the original data model delivered by SAP.	To ensure that the original SAP FSDM data model remains unchanged, you create your own conceptual data model in SAP PowerDesigner first and then create shortcuts to the required SAP Financial Services Data Management entities in your model. Since these shortcut entities are read-only, the original data model remains unchanged. When SAP releases new updates, the shortcut entities are updated automatically.
Do not delete any fields in SAP PowerDesigner.	Fields that are not needed must be left empty.

Adhere to the following naming conventions:	• If you are an **SAP customer**, you must assign the prefix `C_` to new entities or fields. • If you are an **SAP partner**, you must assign the prefix `P_<Partner_ID>_` to new entities or fields. You can remove leading zeros from your ID. The partner ID is issued by SAP and assigned to everybody registered at https://partneredge.sap.com/.
Create your own entity with the relevant fields from the standard entity and then add the new primary key.	You are not allowed to add a primary key attribute to the standard FSDM entities delivered by SAP, as this would change the semantic meaning of the FSDM standard entity. Do not add the following in the CDM: • A primary key attribute in the extension of a standard FSDM entity • A dependent relationship that would end up as a primary key attribute in the standard FSDM PDM entity
Make sure that the SAP Financial Services Data Management data model is deployed.	To deploy the SAP Financial Services Data Management data model, see Deploying SAP Financial Services Data Management [page 75].
Make sure you have activated *Show Hidden Files* in SAP Web IDE for SAP HANA.	Choose *View* > *Show Hidden Files*.

5.1.1 Configure SAP PowerDesigner

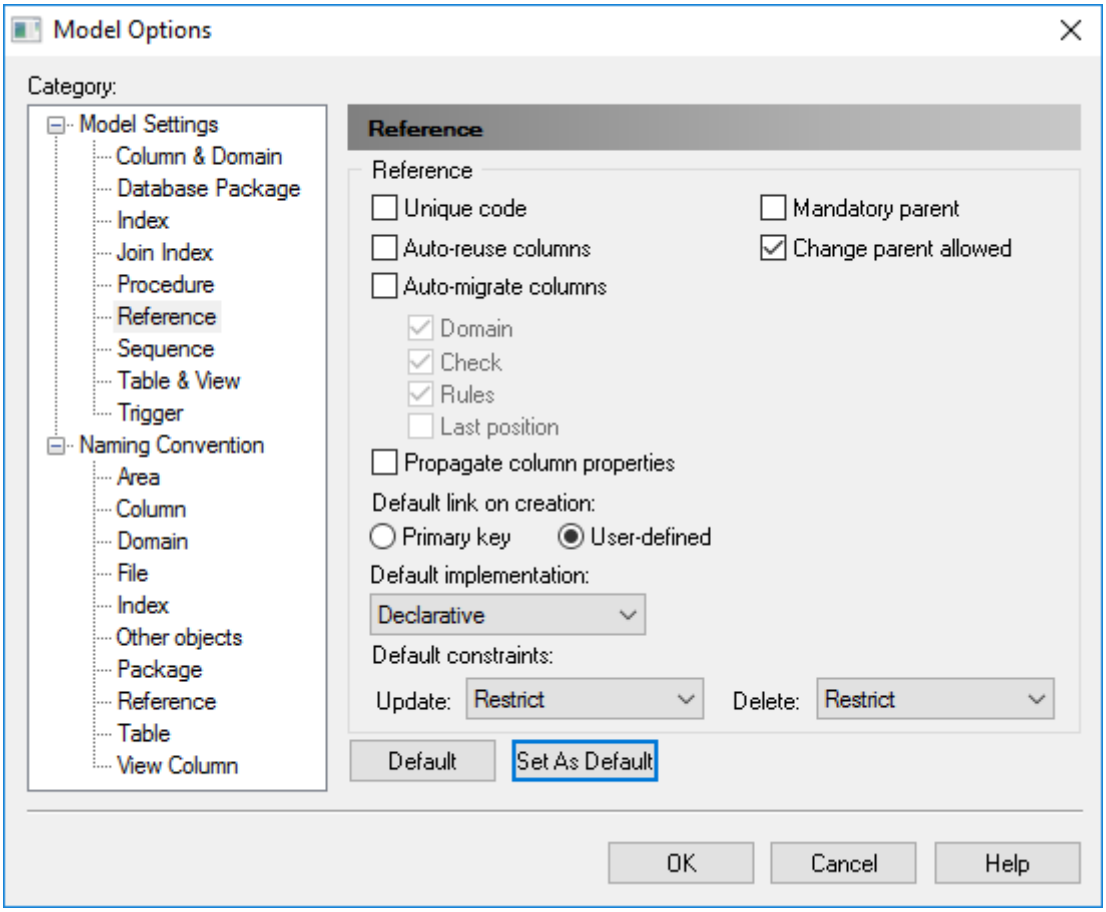
Context

Before you start to extend the data model, we strongly recommend that you make the following settings in SAP PowerDesigner, to make sure that you do not accidentally change the original SAP Financial Services Data Management data model.

Procedure

1. Set shortcuts to read-only.
   a. In the SAP PowerDesigner menu bar, choose *Tools* > *General Options...* and choose the *Dialog* category.
   b. Under *Shortcut property sheets*, set both buttons to *Shortcut*.
   c. Click *OK*.
2. Set references to user-defined.

a. Open the PDM.
b. In the menu bar, choose *Tools* > *Model Options* > *Model Settings* > *Reference*.
c. Change the settings as shown below:



d. We recommend that you set this as the default to make sure that these settings are also valid for further enhancements.
e. Click *OK*.

5.1.2 Extend the CDM

This step describes how to extend the conceptual data model (CDM) in SAP PowerDesigner.

Prerequisites

Extending the data model in SAP PowerDesigner ensures that you keep lineage between the conceptual data model (CDM), the physical data model (PDM), and the SAP HANA CDS files that are deployed on the database.

Before you extend the CDM, take the restrictions listed under Extensibility Steps [page 101] into account, and make sure that the following prerequisites are met:

Configure SAP PowerDesigner [page 102]

Procedure

1. Open SAP PowerDesigner and import the SAP Financial Services Data Management project file by choosing *File* > *Open*.
2. Double-click the CDM (green color) and PDM (below the CDM) in the *Object Browser*. The models open in the *Canvas* to the right, a tabbed view that displays your open diagrams.
3. Create your own project:
    1. Choose *File* > *New Project...* in the menu bar. A new window opens.
    2. Give the project a name of your choice and confirm.

        > **Example**
        >
        > New project: *FSDM_Extensibility*

    3. Create your own CDM: Select your project and choose *File* > *New Model...*. A new window opens.
    4. Select *Model types* on the left and double-click *Conceptual Data Model*. The new CDM now appears under your project. You can give it a name of your choice.

        > **Example**
        >
        > New CDM: *Customer_CDM*

    5. Right-click your CDM and choose *Model Options...*.
    6. Under *Model Settings*, find the *Notation* dropdown menu and select *Entity/Relationship*. We recommend that you set this as the default.
    7. Under *Naming Convention*, go to the *Code* tab page, select the character case *Mixed CASe* and confirm. We recommend that you set this as the default.
4. Create shortcuts for the entities that you want to extend in the SAP Financial Services Data Management project on your CDM:
    1. If not already open, right-click the CDM *FSDM* and choose *Open*.
    2. In the *Object Browser*, search for and select an entity that you want to extend.

        > **Example**
        >
        > FSDM entity for shortcut: *Ship*

    3. Drag the entity into the area of your new CDM (*Canvas*). This creates a shortcut.
       The shortcut symbol in the bottom left corner of each shortcut entity indicates that the shortcut has been created.
5. Create a new entity and add your attributes to it:
    1. In the *Toolbox* (on the right) under *Conceptual Diagram*, click *CDM Entity*, then drop a new entity to the canvas next to your shortcut.
    2. Double-click the new entity. The *Entity Properties* window opens.
       Under *General*, give it a name of your choice.

        > **Example**
        >
        > New entity: *ShipExtension*

        Under *Attributes*, add new attributes.

Example

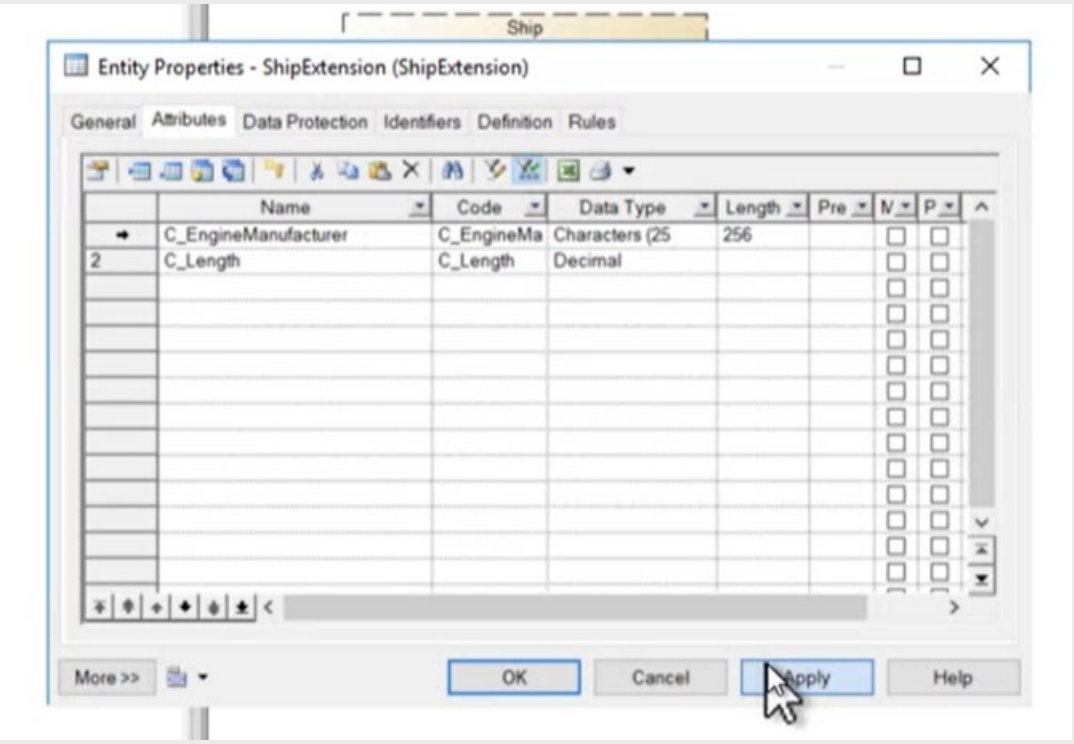
New attributes: *C_EngineManufacturer* and *C_Length*

Define the data type for both attributes.

Example

New data types for both attributes:

- *C_EngineManufacturer:* Characters (256)
- *C_Length:* Decimal



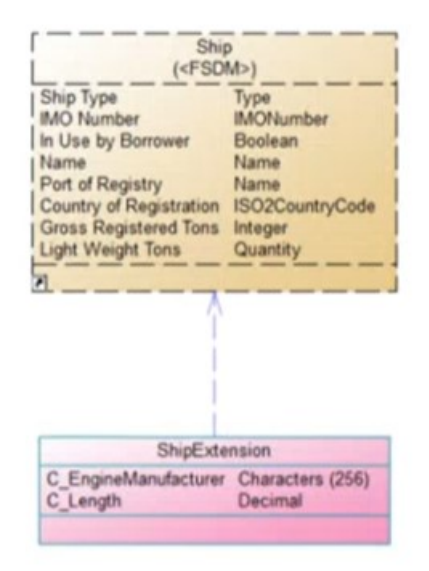
3. Optional: To distinguish the shortcut entity from your extension entity, change the color of your entity: Select your extension entity (*ShipExtension*), right-click and choose *Format...*. On the *Fill* tab page, choose a color and confirm.
4. Connect your entity (*ShipExtension*) with the existing shortcut entity (*Ship*) through a traceability link using the arrow icon in the *Toolbox*. The arrow has no technical function but helps you to make the connection graphically visible.

Note

For other use cases, you can also build relationships instead of tracability links. For more information, see Adding Relationships [page 123].

Example

The diagram for this example now looks like the following:



6. Save your changes.

Next Steps

To ensure lineage between the CDM, PDM, and the SAP HANA CDS model that is deployed, proceed with Extend the PDM [page 106].

5.1.3 Extend the PDM

This step describes how to extend the physical data model (PDM) after you have extended the conceptual data model (CDM) in SAP PowerDesigner.

Prerequisites

Extending the data model in SAP PowerDesigner ensures that you keep lineage between the conceptual data model (CDM), the physical data model (PDM), and the SAP HANA CDS files that are deployed on the database.

Before you extend the PDM, take the restrictions listed under Extensibility Steps [page 101] into account, and make sure that the following prerequisites are met:

- Configure SAP PowerDesigner [page 102]
- Extend the CDM [page 103]

Procedure

1. Generate a PDM of your extension entity:
    1. Choose F4 and deactivate the checks that you do not want to be executed. We recommend that you deactivate all checks.
    2. Select your CDM and go to *Tools* > *Generate Physical Data Model...* in the menu bar. A new window opens.
    3. Select *Generate new Physical Data Model* and give it a name of your choice.

        > **Example**
        >
        > New physical data model:
        > - Name: *Customer_PDM*
        > - Code: *CUSTOMER_PDM*

    4. Select *SAP HANA 2.0 CDS HDI* as DBMS.
    5. Go to the *Selection* tab page and select all your extension entities.

        > **Example**
        >
        > Extension entity: *ShipExtension*

        Deselect all the entities that you copied as shortcuts from FSDM. Click *OK*.
    6. A confirmation dialog may appear to inform you that there is no associated generated model to the FSDM CDM. Select *Yes* to continue.
    7. Optional: Change the color of the generated entity to visibly mark it as your own.
2. Connect your entity to other entities using CDS extensions:

    > **Example**
    >
    > Connect the new entity *ShipExtension* to the existing entity *PhysicalAsset*.

    1. Right-click your new PDM (*Customer_PDM*) and choose *Physical Diagram*.
    2. Link the extension entities to the PDM entities:
       Double-click the PDM *FSDM* and search under the folder *CDS Entities* for the entity that you want to extend.

        > **Example**
        >
        > Search for the entity *PhysicalAsset*.

        > **Note**
        >
        > If you are not sure which entities your extension extends in the PDM, you can use the SAP PowerDesigner mapping editor (*Tools* > *Mapping Editor...*) in the standard SAP Financial Services Data Management project. Open the PDM to check which CDM entity is linked to which PDM entity.

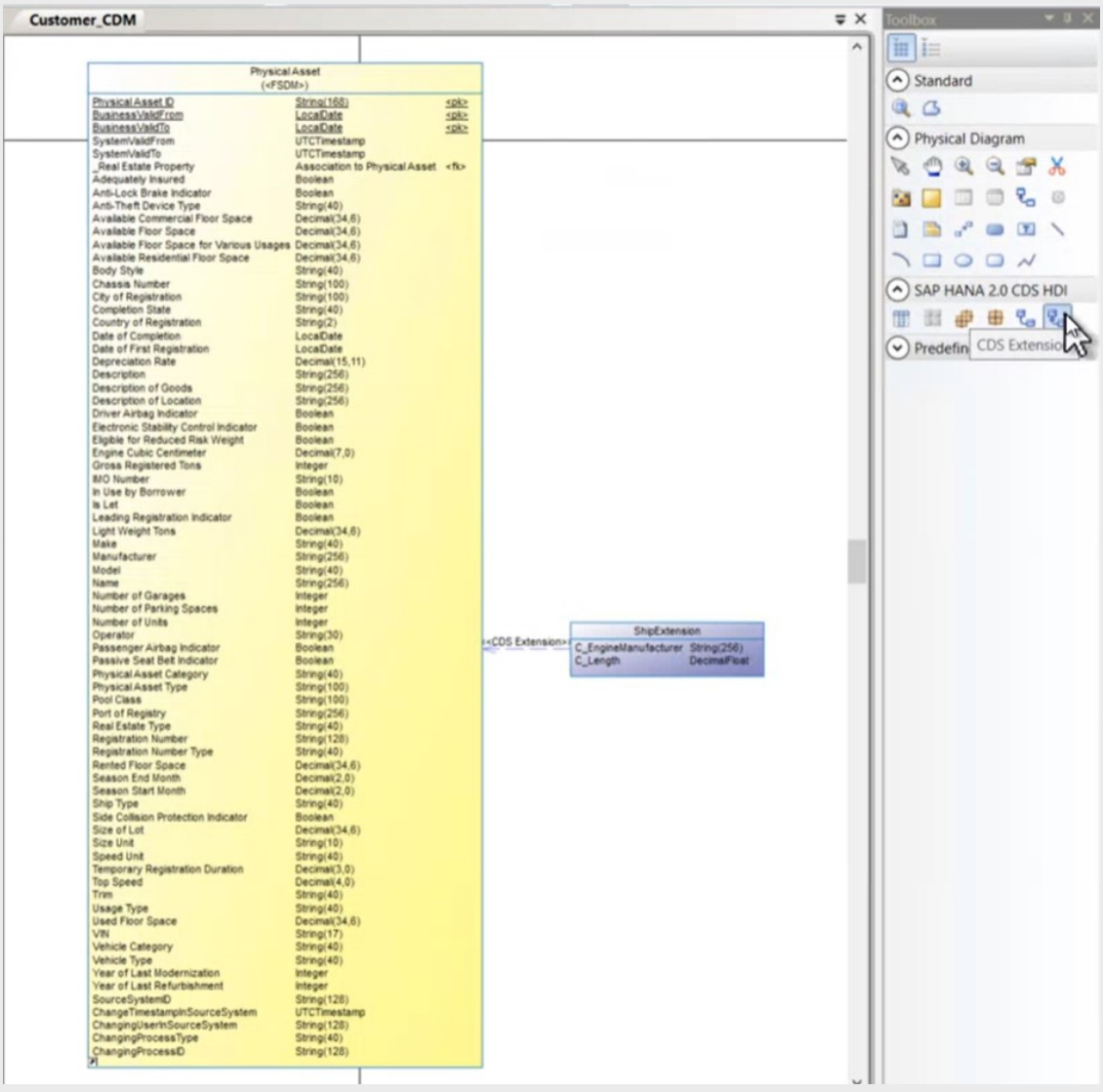
    3. Drag and drop the entity in your PDM area. This creates a shortcut.

4. Extend the existing entity with your new entity by using the CDS extension link.

> ### Example
>
> Extend the shortcut *PhysicalAsset* entity with your new *ShipExtension* entity.

3. Check your final data model.

> ### Example
>
> The data model for this example now looks like the following:



4. Change the namespace:
   Right-click your PDM, choose *Properties* and enter `customer.extension` as the namespace.
5. Export the generated CDS extension files:
   1. Open the FSDM PDM in your PowerDesigner workspace. Keep it open during the extraction of CDS extension files.
   2. Choose F4 and deactivate the checks you do not want to be executed. We recommend that you deactivate all checks.

3. Select your PDM and go to *Database* > *Export CDS Objects* in the menu bar.
4. Select your extension entity on both the *CDS Entities* and *Physical Data Models* tab. The system generates two files.

   > **Example**
   >
   > Extension entity: *ShipExtension* with generated files:
   > - *.hdnamespace*
   > - *ShipExtension.hdbcds*

5. Click *OK*.
6. Check the generated CDS extension files.
7. Save the CDS extension files as a zip file.

Next Steps

To ensure lineage between the CDM, PDM, and the SAP HANA CDS model that is deployed, proceed with Deploy CDS Files [page 109].

5.1.4 Deploy CDS Files

This step describes how to deploy the CDS files on SAP HANA after you have extended the conceptual data model (CDM), and the physical data model (PDM) in SAP PowerDesigner.

Prerequisites

Extending the data model in SAP PowerDesigner ensures that you keep lineage between the conceptual data model (CDM), the physical data model (PDM), and the SAP HANA CDS files that are deployed on the database.

Before you deploy the CDS files on SAP HANA, take the restrictions listed under Extensibility Steps [page 101] into account, and make sure that the following prerequisites are met:

1. Configure SAP PowerDesigner [page 102]
2. Extend the CDM [page 103]
3. Extend the PDM [page 106]
4. The SAP Financial Services Data Management data model is deployed as described in Deploying SAP Financial Services Data Management [page 75].

Procedure

1. Launch SAP Web IDE for SAP HANA.

   Go to *Tools* > *Preferences* > *Default Editors* and choose the *Code Editor* as the default editor.

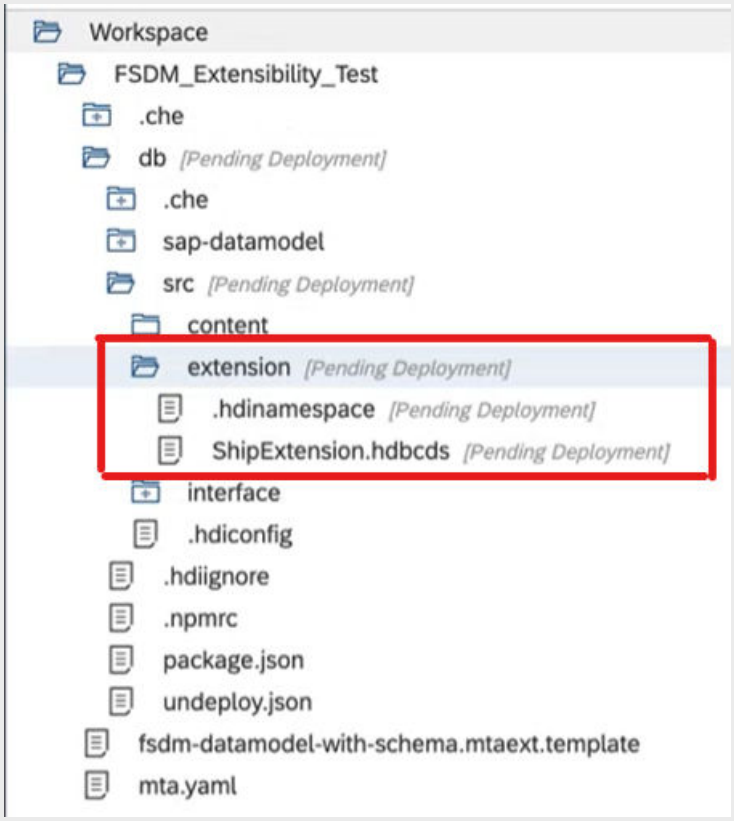
2. Go to the development area of SAP WebIDE for SAP HANA and open the folder *db* > *src* > *extension*. If this folder does not already exist, create it.

3. Import the CDS extension files that you generated as part of the step Extend the PDM [page 106]:

   Right-click the extension folder and choose *Import* > *File or Project*. Select the zip file that includes your CDS files.

   > **Note**
   >
   > Make sure that your Web IDE shows hidden files to see all files. If not, choose *View* > *Show Hidden Files*.

   In the field *Import to*, make sure that the import path ends with …`/db/src/extension`.

   > **Example**
   >
   > Imported files:
   >
   > 
   >

4. Make sure that you add the following content in all imported `hdbcds` files in your extension folder:

   ```
   in package "customer"."extension";
   ```

   If not generated, add in addition:

   ```
   using "sap"."fsdm"::"PhysicalAsset_Historical";
   ```

From the generated file `extend entity` delete the following:

```
sap.fsdm::
```

Example

Added/deleted content for *ShipExtension.hdbcds*:

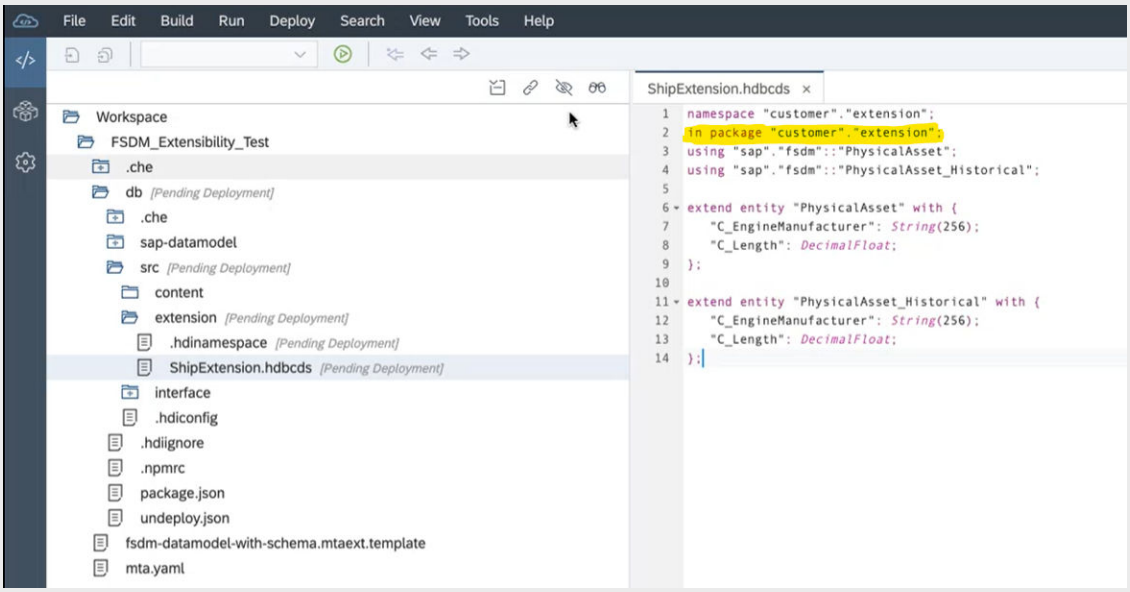
*ShipExtension.hdbcds ×

```
1  namespace "customer"."extension";
2  in package "customer"."extension";
3
4  using "sap"."fsdm"::"PhysicalAsset";
5  using "sap"."fsdm"::"PhysicalAsset_Historical";
6
7  extend entity "PhysicalAsset" with {
8     "C_EngineManufacturer": String(256);
9     "C_Length": DecimalFloat;
10 };
11
12 extend entity "sap.fsdm::PhysicalAsset_Historical" with {
13    "C_EngineManufacturer": String(256);
14    "C_Length": DecimalFloat;
15 };
```

As a result, the files include the package declaration in the beginning.

Example

Added content in extension folder *ShipExtension.hdbcds*:

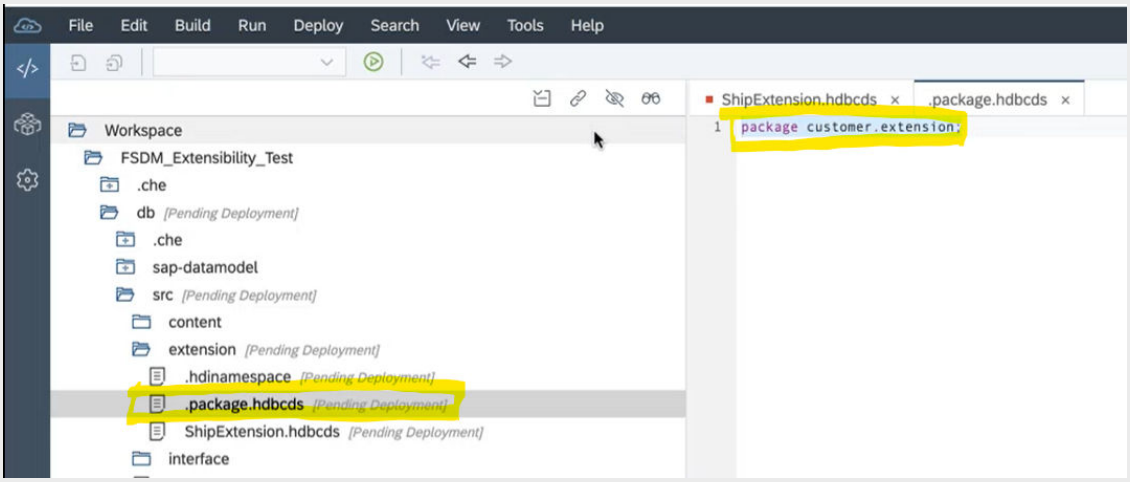


5. Manually add the file `.package.hdbcds` to the folder *extension* including the following content:

```
package customer.extension;
```

Example

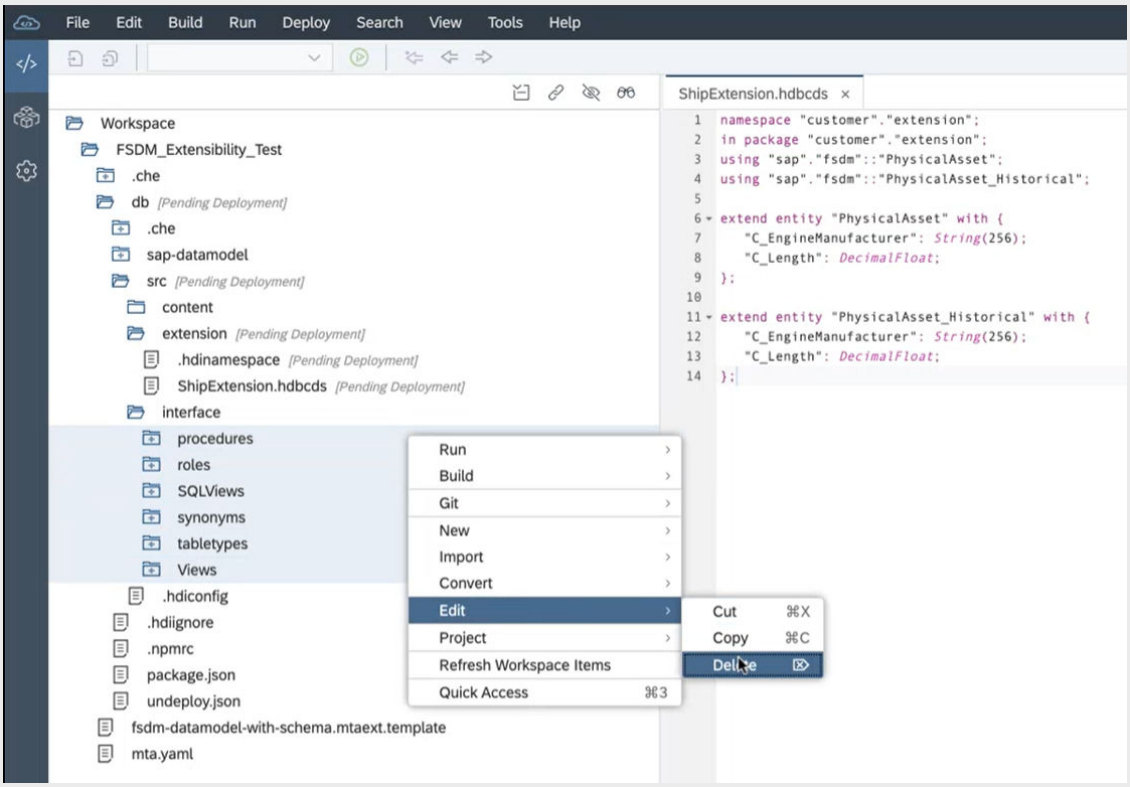
Folder *.package.hdbcds*:



6. Deploy the CDS extension files: Right-click `db/` and choose *Build/Build*.
7. Before you import the generated interfaces, under *db* > *src* > *interface* delete all existing entries but the header.

Example

Delete files in folder *interface*:



8. Regenerate the interfaces as described in Deploying the Generated Write or Read Interface to SAP HANA XS Advanced Runtime [page 77].
9. Import the zip file under *db* > *src* > *interface*. Test your extension again with test data.
10. Deploy the CDS extension files again: Right-click `db/` and choose *Build/Build*.
11. Go to Database Explorer and find the SAP Financial Services Data Management data model that has already been deployed.
    Test the upgrade of the data model on SAP HANA on a container with test data. You can do this either in SAP Web IDE using a build on the db folder, or on the command line using `xs deploy` on test data.

    ### Example

    Check that the entity *PhysicalAsset* contains the attributes *C_EngineManufacturer* and *C_EnginePower*.
12. Commit the new files to your Git repository.

5.2 Adding Data Fields

This example shows how you can extend the data model with new data fields, illustrated by two new attributes.

A use case for this example is that some boat insurers provide hull insurance that cover physical damage of a boat and may include machinery, sails, furniture, outboard motors and tenders, for example. Tenders are not

always covered automatically. Specific details about a tender such as size or engine manufacturer are required by some insurers. You can, therefore, add this details of a ship to the SAP Financial Services Data Management data model with extending the entity `ship`.

The names of entities and attributes used in this section are examples used to illustrate different extensibility scenarios.

This example shows, how you can extend the entity *ship* by adding your own attributes, since some insurers require this information.

Overview of Sample Objects

Object	Example
New project	`FSDM_Extensibility`
New CDM	`Customer_CDM`
FSDM entity used for shortcut	`Ship`
New entity	`ShipExtension`
New data types (attributes)	`C_EngineManufacturer` and `C_Length`
New PDM	`Customer_PDM`
FSDM entity that you extend	`PhysicalAsset`
Generated files	`.hdbnamespace` and `ShipExtension.hdbcds`

Procedure

To add these attributes, proceed as described in the linked steps below. This example is included in this detailed step-by-step description:

1. Extend the CDM [page 103]
2. Extend the PDM [page 106]
3. Deploy CDS Files [page 109]

5.3 Adding Entities

This example shows how you can add a new entity to the data model.

A use case for this example is that some boat insurers provide hull insurance that cover physical damage of a boat and may include machinery, sails, furniture, outboard motors and tenders, for example. Tenders are not always covered automatically. Specific details about a tender such as size or engine manufacturer are required by some insurers. You can, therefore, add this details of a ship to the SAP Financial Services Data Management data model with extending the entity `ship`.

The names of entities and attributes used in this section are examples used to illustrate different extensibility scenarios.

This example shows how you can add your own attributes to the *Ship* entity , since some insurers require this information.

Overview of Sample Objects

Object	Example
New project	`FSDM_Extensibility`
New CDM	`C_Tender`
FSDM entity used for shortcut	`Ship`
New entity	`ShipExtension`
New data types (attributes)	`C_ID` and `C_Capacity`
Relationship	1:n relationship `C_Ship Has Tender`
New PDM	`Customer_PDM`
History Table Name	`C_Tender_Historical`
FSDM entity that you extend	`PhysicalAsset`
Generated files	`.hdbnamespace` and `ShipExtension.hdbcds`

Procedure

To add these attributes, proceed as described in the linked steps below. Replace the object names with those listed in the table above:

1. Extend the CDM [page 103] with the following deviation:
   Instead of step 5d:
   Build a relationship between the shortcut *Ship* entity and the new *C_Tender* entity using the corresponding icon in the **Toolbox** (**1 : n** relationship).

Example

The example below shows the diagram for the new *C_Tender* entity as well as the *Ship_Extension* entity from the first example (Adding Data Fields [page 113]). Both examples are independent to each other, but can be combined as you like.

This diagram also shows the graphical representation of the different connections to the entity shortcut `Ship`, since `C_Tender` is connected through a relationship, while `ShipExtension` is connected through a traceability link. For more information about relationships, see Adding Relationships [page 123].

2. Extend the PDM [page 106] with the following deviation:
   1. In step 1e select:

      Example

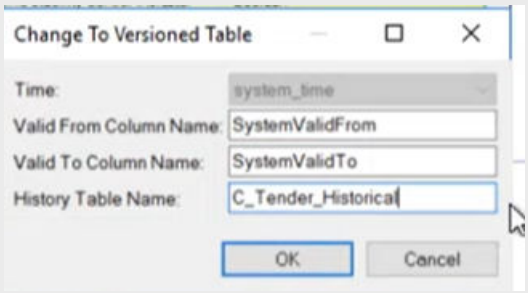
      Extension entity *C_Tender*

   2. After step 1, change the generated PDM of your extension entity to a versioned table:
      Right-click the new table in the diagram and select *Change to Versioned Table*. In the dialog, review the proposed names for the VALIDFROM and VALIDTO columns to **`SystemValidFrom`** and **`SystemValidTo`** and for the historical table to **`<YourTableName>_Historical`** and click OK. SAP Powerdesigner adds the columns for versioning to the current table. On the *Columns* tab, change the data type from TIMESTAMP to **`UTCTimestamp`** and mark the colums as mandatory.

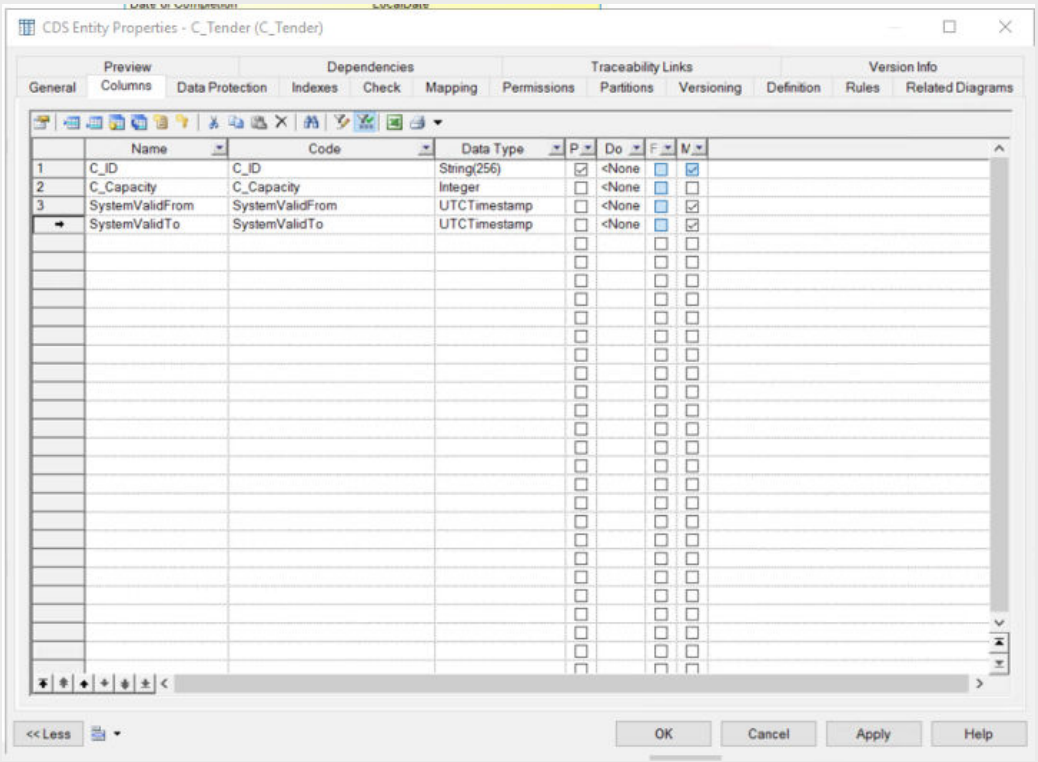
      Example

      Right-click your new table *C_Tender* and select *Change to Versioned Table*

      - Valid From Column Name: *SystemValidFrom*
      - Valid To Column Name: *SystemValidTo*
      - History Table Name: *C_Tender_Historical*

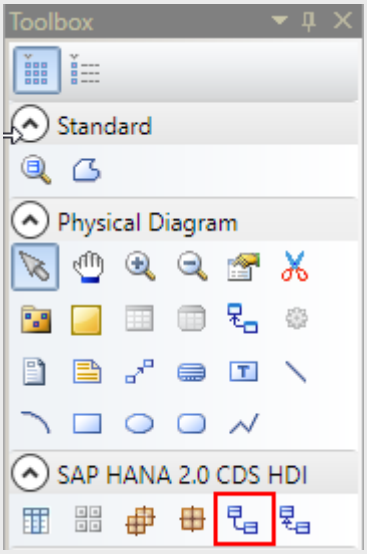
- Then update the data types and mark them as mandatory:



3. Instead of step 2 d:
   Build an association between your new entity and the shortcut:

Example

Build an association from `C_Tender` to `PhysicalAsset` as shown below.

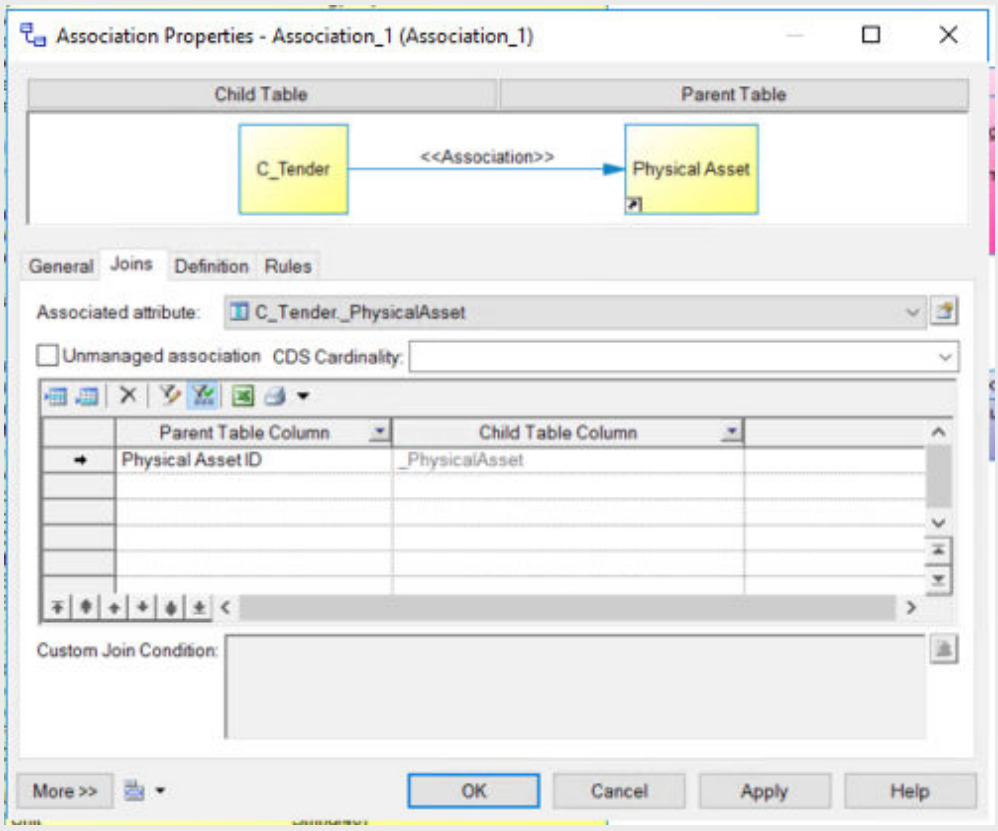


Note

You must use unmanaged associations in SAP Financial Services Data Management (for more information about associations, see CDS Associations). Therefore, you need to manually create the foreign key fields in the referencing table (not applicable for versioning fields):

- Double-click *C_Tender*. Add the field *_PhysicalAsset*. You do not need a data type for this field, since this is used for the associations to be created in the following steps. Click *OK*.
- Double-click the association arrow. A new window opens. Go to the *Joins* tab page and if not already done, mark the checkbox *Unmanaged association*.
- In the *Associated attribute* dropdown, select *C_Tender._PhysicalAsset*.
- In the *Parent Table Column*, select *Physical Asset ID* and in *Child Table Column*, select *_PhysicalAsset*.
- Unmark the checkbox *Unmanaged association* and click *OK*.

Example

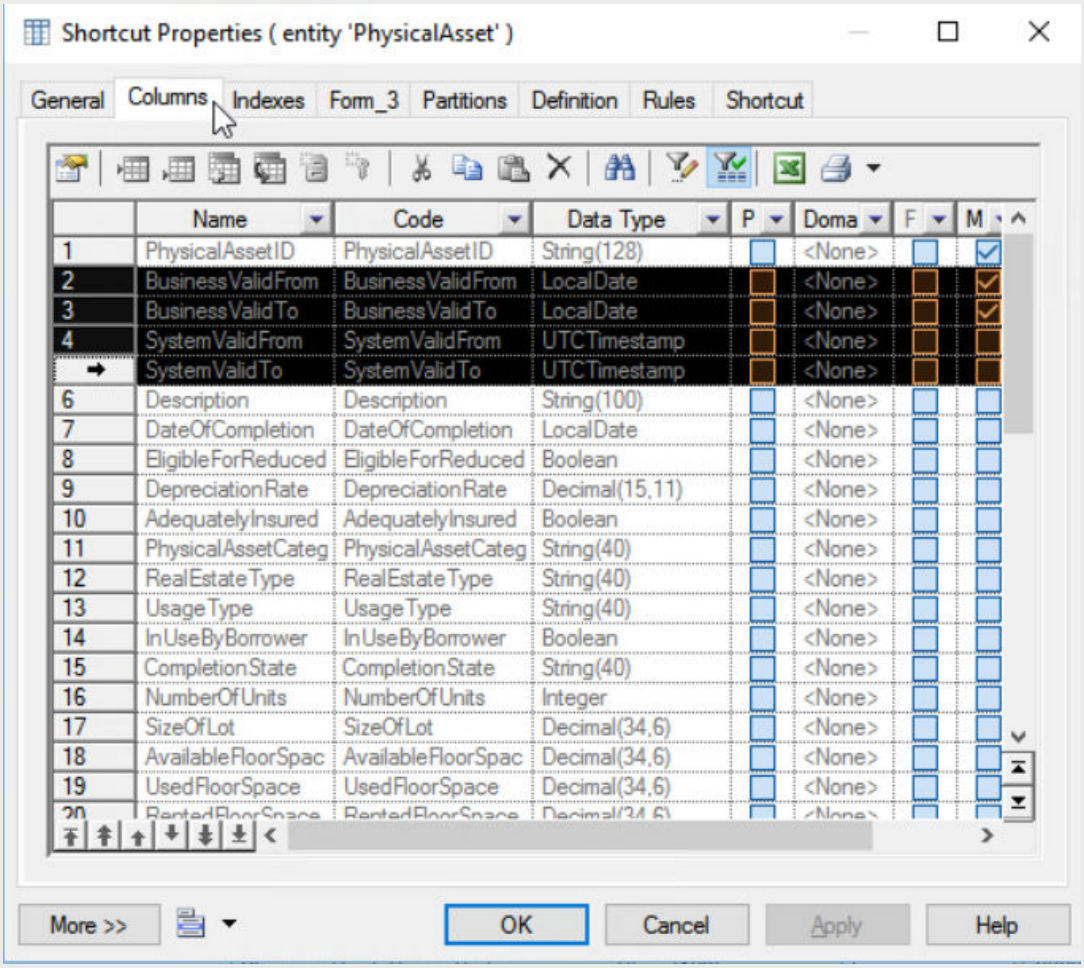


- Remove other parent table/child table suggestions that were generated by SAP PowerDesigner.

4. Edit the associated key field of the entity `C_Tender` manually:
   Double-click the entity and go to the **Columns** tab page. Add business and system versioning attributes, define the data type, and set business versioning as the primary keys. You can copy the versioning fields from the `PhysicalAsset` table. The first entry is the primary key of the entity. The versioning fields follow the primary key.

Example

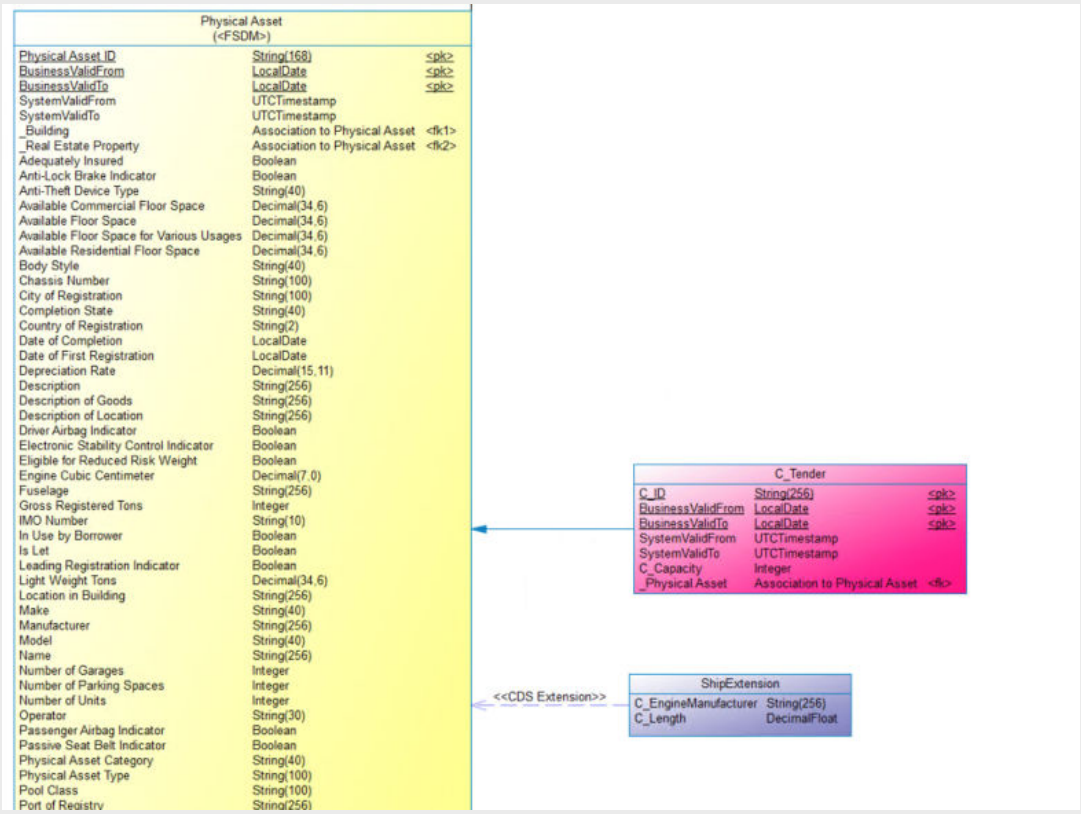
How to copy the versioning fields:



Attribute Name	Attribute Type	Is Primary Key	Mandatory
BusinessValidFrom	LocalDate	X	X
BusinessValidTo	LocalDate	X	X
SystemValidFrom	UTCTimestamp		X
SystemValidTo	UTCTimestamp		X

Example

The final PDM in this example looks like this:

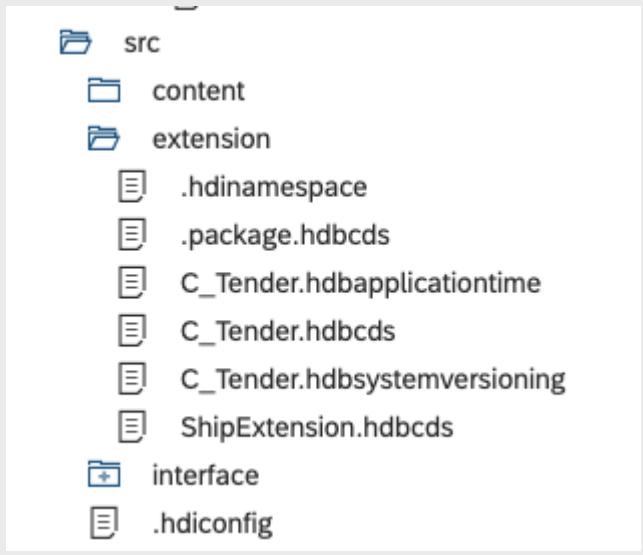


5. Proceed with step 4 and 5 as described in Extend the PDM [page 106] to change the name space, and export the generated CDS extension files.

Example

The system generates the following files:

- *.hdnamespace*
- *C_Tender.hdbcds*
- *C_Tender.hdbsystemversioning*
- *ShipExtension.hdbcds*



3. Deploy CDS Files [page 109] with the following deviation:
   Instead of step 5, proceed as follows:
   1. If not already available, add a new file `C_Tender.hdbsystemversioning` with the following content:

      ```
      SYSTEM VERSIONING "customer.extension::C_Tender" ("SystemValidFrom",
      "SystemValidTo")
      HISTORY TABLE "customer.extension::C_Tender_Historical" NOT VALIDATED
      ```

   2. Add a new file `C_Tender.hdbapplicationtime` with the following content:

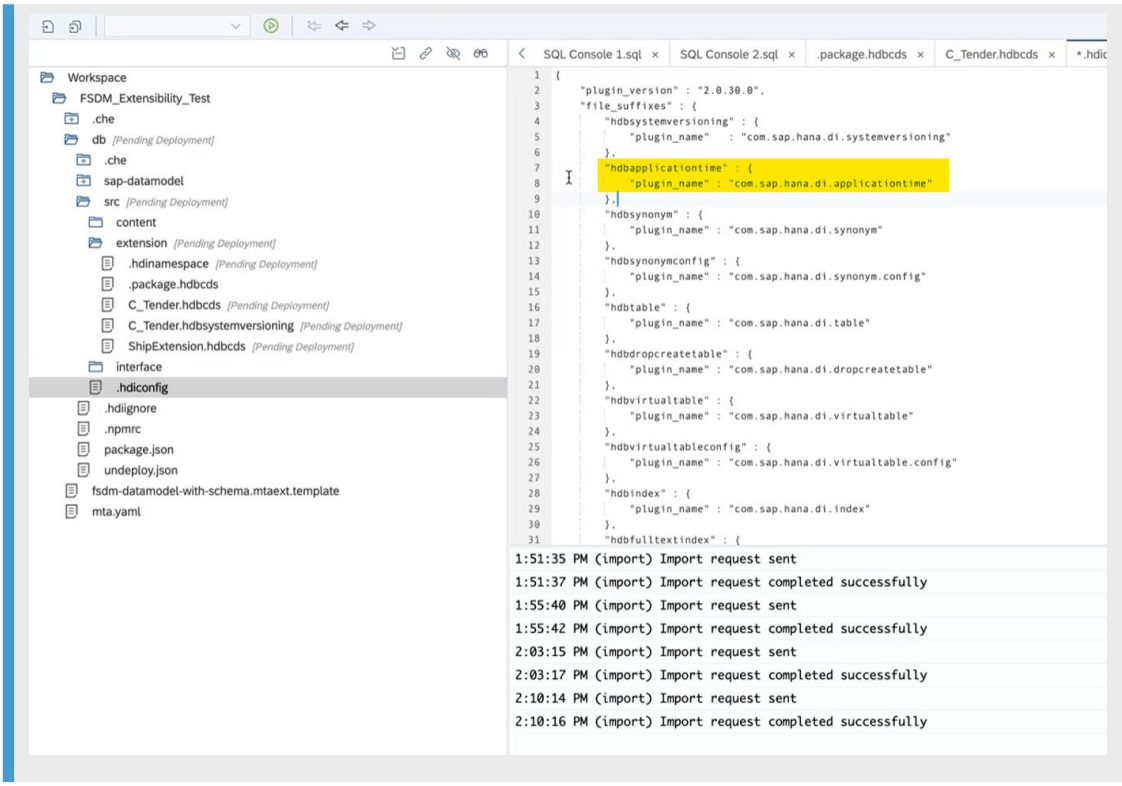
      ```
      APPLICATION TIME "customer.extension::C_Tender" ("BusinessValidFrom",
      "BusinessValidTo")
      ```

   3. In folder `src` add in existing file `.hdiconfig` the following content:

      ```
      "hdbapplicationtime" : { "plugin_name" :
      "com.sap.hana.di.applicationtime" },
      ```

      ### Example

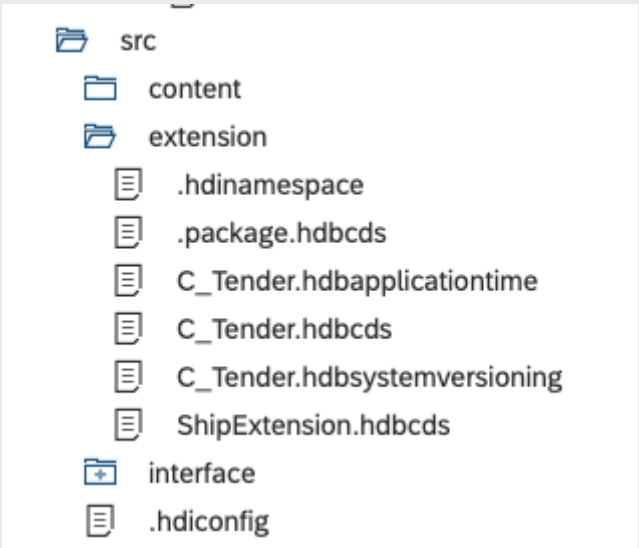
      This looks as follows:



Then follow the description of Deploy CDS Files [page 109] again.

Example

As a result, the generated files look as follows:



5.4 Adding Relationships

This example shows how you can add relationships to the data model.

You have multiple options:

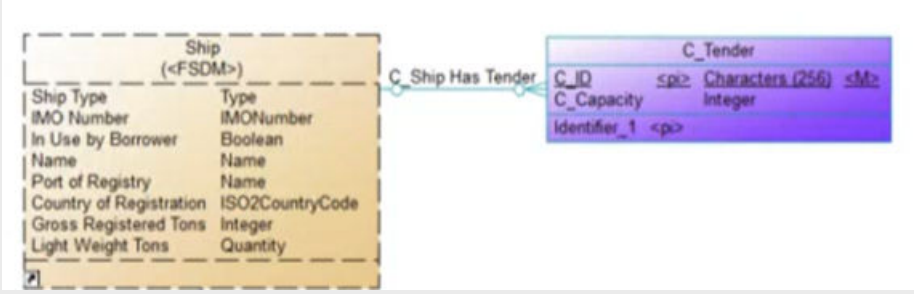
- You can add relationships between existing entities delivered with the data model from SAP Financial Services Data Management.
- You can create relationships between entities that you have created yourself.
- You can add relationships between entities delivered with the data model from SAP Financial Services Data Management and entities that you have created yourself.

You can combine these options with multiple dimensions as follows:

- **1 to n** relationship
  - 1 FSDM : n CUSTOMER
    In the conceptual data model (CDM) in SAP PowerDesigner, you can add a **1 to n** relationship between one FSDM entity and **n** customer entities. To build this relationship, proceed as described in Adding Entities [page 114], since this relationship is included there.

    ### Example

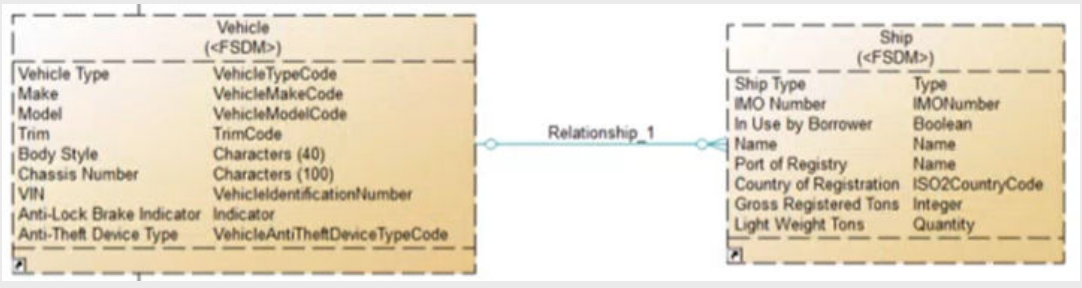
    The following diagram shows a **1 to n** relationship between the shortcut entity `Ship` and your own entity `C_Tender`:

  - 1 FSDM : n FSDM
    In the CDM in SAP PowerDesigner, you can add a **1 to n** relationship between one FSDM entity and n FSDM entities. To build this relationship, you have to create a shortcut for each FSDM entity and build a relationship between them, as described above.

    ### Example

    The following diagram shows a **1 to n** relationship between the shortcut entity `Vehicle` and the shortcut entity `Ship`:

    - 1 CUSTOMER : n FSDM
      In the CDM in SAP PowerDesigner, you can add a **1 to n** relationship between one customer entity and N FSDM entities. To build this relationship, proceed the same way as described above.
- **n to m** relationship
    - FSDM : FSDM
      You can add an **n to m** relationship between two FSDM entities.
    - FSDM : CUSTOMER
      You can add an **n to m** relationship between an FSDM and customer entities.
- **1 to 1** relationship
    - You can add a **1 to 1** relationship between different PDM entities in the FSDM data model.
    - You can add a **1 to 1** relationship between a PDM entity and another instance of itself in the FSDM data model.

More Information

For more information about relationships in the data model, see the General Information in the user guide.

5.5 Adding Code List Values

This section shows how you can extend code lists for tables shipped with the data model of SAP Financial Services Data Management.

The SAP Financial Services Data Management (FSDM) data model contains code lists for certain tables. When you deploy the data model, the table content is loaded into the tables also. You can find the source for these code lists in folder `db/sap-datamodel/<version>/@sap-fsdm/cds-datamodel/src/data/CodeList (replace <version> with the current version number)`.

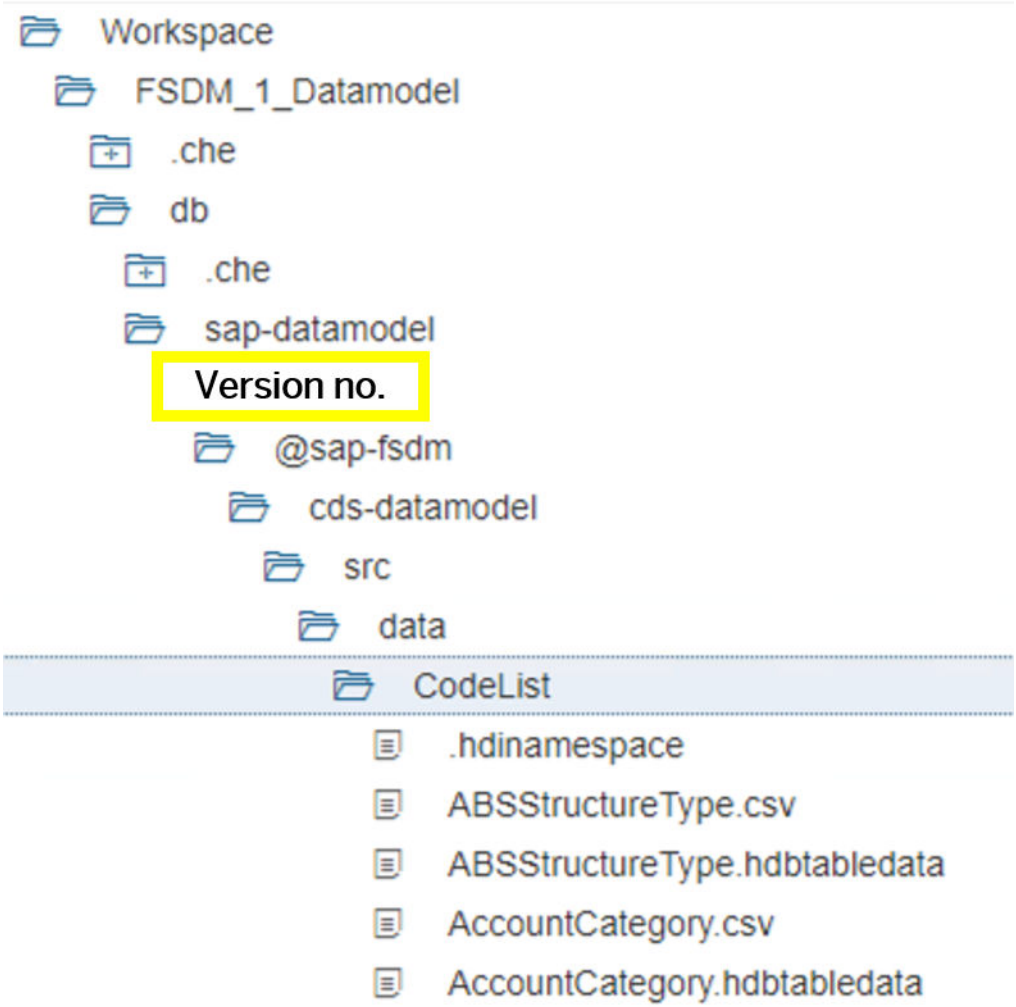
> **Note**
>
> Although the code list values are delivered in the FSDM data model, they are examples only and are subject to change. Definitions of code list values are available in SAP Note 2771618.

When you extend SAP code lists, we recommend that you use the prefix `C_` (standing for customer) or `P_` (standing for partner) for your own table entries. This ensures that during a technical upgrade, there are no inconsistencies between new entries by SAP and your own entries.

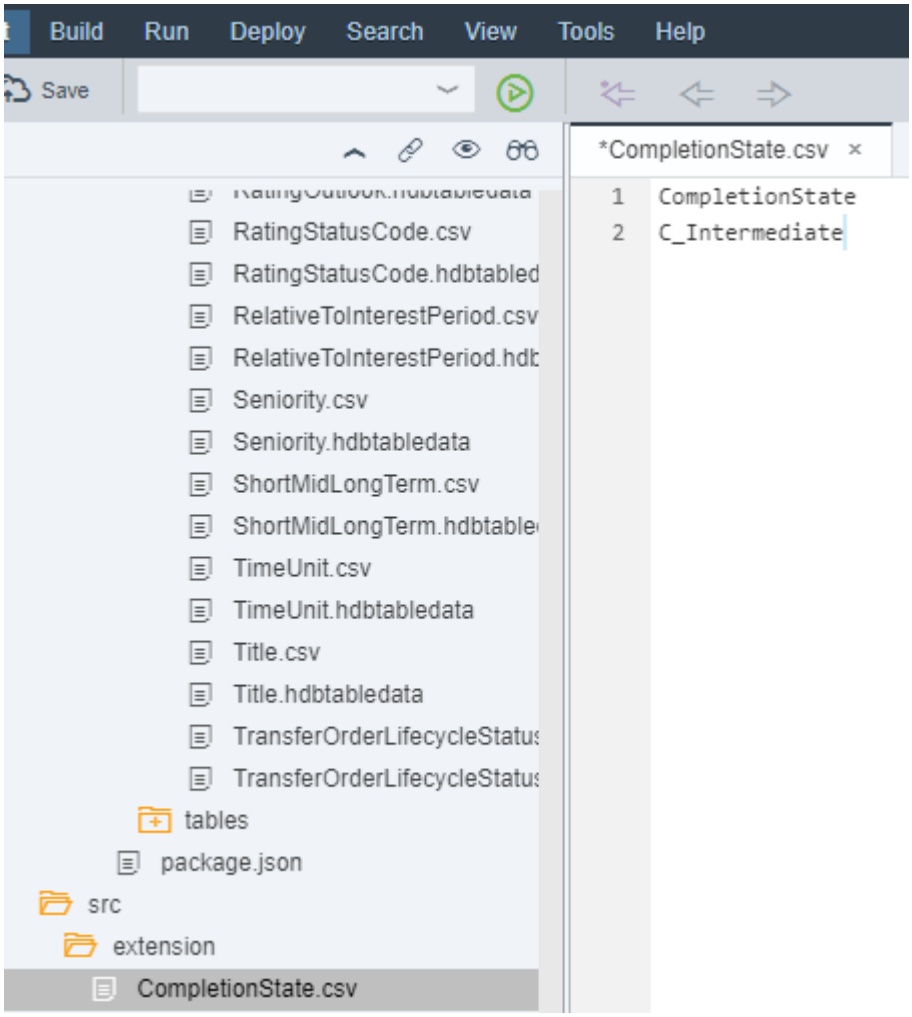
Procedure

To create your own table entries for code lists, perform the following steps:

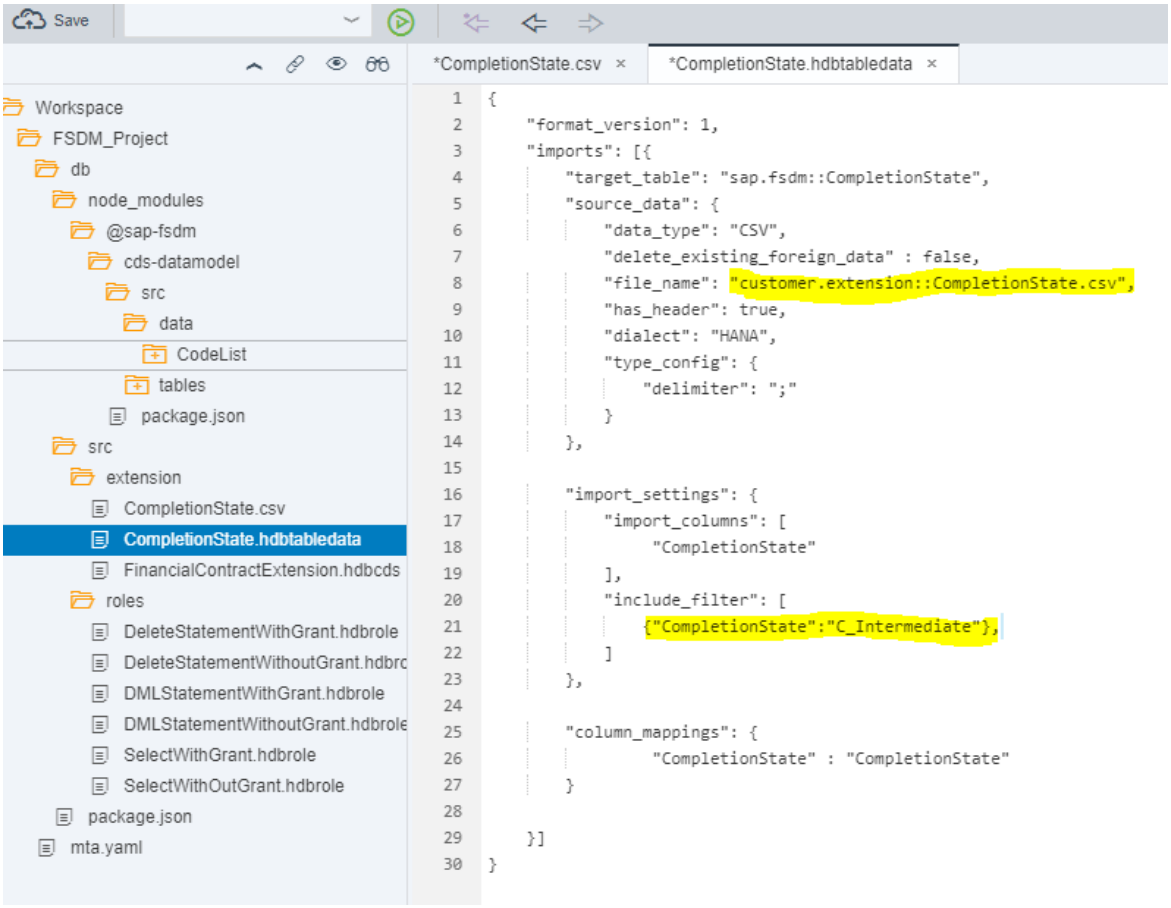
1. Launch SAP Web IDE and check out the project that you created from the FSDM template.
2. In the development area, choose a code list (such as `CompletionState`) in folder `db/sap-datamodel/1.9.0/@sap-fsdm/cds-datamodel/src/data/CodeList`.



3. Copy the corresponding `csv` and `hdbtabledata` files and paste them into your `db/src/extension` folder.
   Open the `csv` file with Code Editor and delete all existing entries but the header. Add an entry using the customer namespace starting with `C_` (for example `C_Intermediate`).



4. Open the `hdbtabledata` file with Code Editor and replace the filters section with any keys that you added in step 3. Adjust the namespace of the `csv` file in the `hdbtabledata` file under `file_name` to the namespace used in your extension folder, for example `customer.extension`.
   For more information about table data, see Table Data (.hdbtabledata).



If you do not have a `hdinamespace` file in your extension folder, create one with your own namespace:

```
{
    "name": "customer.extension",
    "subfolder": "append"
}
```

5. Build on db level.
6. Go to the Database Explorer and check whether the new entry exists in the table.
7. Commit the new files to your git repository.

6 Document History

Document history for the administrator's guide for SAP Financial Services Data Management

Document Version	Date	Description
1.23	12-July 2024	Update for version 1.23
1.22	08-March 2024	Update for version 1.22 Update of the recommended software versions listed under Installation and Upgrade Information [page 31]
1.21	17-November-2023	Update for version 1.21
1.20	14-July-2023	Update for version 1.20 New step 8 that describes how to generate additional SQL views added in Deploying the Generated Write or Read Interface to SAP HANA XS Advanced Runtime [page 77].
1.19	17-March-2023	Update for version 1.19 Update of the recommended software versions listed under Installation and Upgrade Information [page 31]
1.18	10-November-2022	Minor update for version 1.18
1.17	29-July-2022	Update for version 1.17 • Step 8 of Deploying the Generated Write or Read Interface to SAP HANA XS Advanced Runtime [page 77] contains a description of how to generate interfaces for entities for which SAP HANA system versioning is disabled.
1.16	08-April-2022	Update for version 1.16
1.15	15-December-2021	Update for version 1.15
1.14	06-August-2021	Update for version 1.14
1.13	28-May-2021	Update for version 1.13 The section Extensibility Information [page 100] has been reworked entirely. Changes regarding the

Document Version	Date	Description
		handling of historical data in the SAP PowerDesigner are included.
1.12	26-February-2021	Update for version 1.12 Additional steps how to install and set up an SAP PowerDesigner repository for your SAP Financial Services Data Management data model have been added to the section Installing or Upgrading SAP PowerDesigner [page 39].
1.11	13-November-2020	Update for version 1.11 • The Getting Started [page 4] section has been reworked. • Additional information and recommentations have been added to the Security Information [page 17] section.

Important Disclaimers and Legal Information

Hyperlinks

Some links are classified by an icon and/or a mouseover text. These links provide additional information.
About the icons:

- Links with the icon : You are entering a Web site that is not hosted by SAP. By using such links, you agree (unless expressly stated otherwise in your agreements with SAP) to this:
  - The content of the linked-to site is not SAP documentation. You may not infer any product claims against SAP based on this information.
  - SAP does not agree or disagree with the content on the linked-to site, nor does SAP warrant the availability and correctness. SAP shall not be liable for any damages caused by the use of such content unless damages have been caused by SAP's gross negligence or willful misconduct.
- Links with the icon : You are leaving the documentation for that particular SAP product or service and are entering an SAP-hosted Web site. By using such links, you agree that (unless expressly stated otherwise in your agreements with SAP) you may not infer any product claims against SAP based on this information.

Videos Hosted on External Platforms

Some videos may point to third-party video hosting platforms. SAP cannot guarantee the future availability of videos stored on these platforms. Furthermore, any advertisements or other content hosted on these platforms (for example, suggested videos or by navigating to other videos hosted on the same site), are not within the control or responsibility of SAP.

Beta and Other Experimental Features

Experimental features are not part of the officially delivered scope that SAP guarantees for future releases. This means that experimental features may be changed by SAP at any time for any reason without notice. Experimental features are not for productive use. You may not demonstrate, test, examine, evaluate or otherwise use the experimental features in a live operating environment or with data that has not been sufficiently backed up.
The purpose of experimental features is to get feedback early on, allowing customers and partners to influence the future product accordingly. By providing your feedback (e.g. in the SAP Community), you accept that intellectual property rights of the contributions or derivative works shall remain the exclusive property of SAP.

Example Code

Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

Bias-Free Language

SAP supports a culture of diversity and inclusion. Whenever possible, we use unbiased language in our documentation to refer to people of all cultures, ethnicities, genders, and abilities.

www.sap.com/contactsap

THE BEST RUN SAP