



Documentation Produit

EBX5 Version 5.6.1 Fix M

EBX5 includes:

- software developed by the Apache Software Foundation - <http://www.apache.org/>
- software developed by the JDOM Project - <http://www.jdom.org/>
- software developed by the JSON Project - <http://www.json.org/>
- the Gagawa HTML Generator - <http://code.google.com/p/gagawa/>
- the Jericho HTML Parser - <http://jericho.htmlparser.net/docs/index.html>
- the H2 Database Engine - <http://www.h2database.com/>
- Font Awesome by Dave Gandy - <http://fontawesome.io/>

Table des matières

Guide utilisateur

Introduction

1. Notions clés.....	9
2. Interface utilisateur EBX5.....	11
3. Glossaire.....	17

Modèles de données

4. Introduction aux modèles de données.....	28
---	----

Implémentation des modèles de données

5. Création du modèle de données.....	31
6. Configuration du modèle de données.....	33
7. Modélisation de la structure des données.....	35
8. Propriétés des éléments du modèle de données.....	41
9. Contrôles sur les éléments du modèle de données.....	47
10. Actions sur les modèles de données existants.....	53

Publication et gestion de versions des modèles de données

11. Publication du modèle de données.....	55
12. Gestion des versions de modèles de données embarqués.....	57

Espaces de données

13. Introduction aux espaces de données.....	62
14. Création d'un espace de données.....	65
15. Actions sur les espaces de données existants.....	67
16. Images.....	75

Jeux de données

17. Introduction aux jeux de données.....	80
18. Création du jeu de données.....	83
19. Visualisation des données.....	85
20. Edition des données.....	93
21. Actions sur les jeux de données existants.....	95
22. Héritage entre jeux de données.....	99

Modèles de workflow

23. Introduction aux modèles de workflow.....	104
24. Modélisation du workflow.....	109
25. Configuration du modèle de workflow.....	117
26. Publication d'un modèle de workflow.....	125

Workflows de données

27. Introduction aux workflows de données.....	128
28. Utilisation de l'interface utilisateur de la section Workflow de données.....	131
29. Bons de travail.....	135

Gestion de workflows de données

30. Lancement et monitoring de workflows de données.....	139
31. Administration de workflows de données.....	141

Services de données

32. Introduction aux services de données.....	146
33. Génération de WSDL pour services de données.....	149

Manuel de référence (en anglais)

Intégration

34. Built-in UI services.....	155
-------------------------------	-----

Services de données

35. Introduction.....	169
36. WSDL generation.....	175
37. Operations.....	183

Services d'import et d'export

38. XML import and export.....	209
39. CSV import and export.....	215
40. Supported XPath syntax.....	221

Guide utilisateur

Introduction

CHAPITRE 1

Notions clés

Ce chapitre contient les sections suivantes :

1. [Concepts et outils associés](#)

1.1 Concepts et outils associés

Le Master Data Management (MDM) est un moyen de modéliser, gérer et gouverner les données partagées. Quand des données sont partagées par plusieurs systèmes informatiques, ainsi que des équipes professionnelles différentes, l'existence d'une seule version gouvernée des données de référence est essentielle.

Avec EBX5, les utilisateurs métier et les informaticiens peuvent collaborer sur une solution unifiée, afin de concevoir des modèles de données et gérer le contenu des données de référence.

EBX5 est un logiciel de gestion des données de référence qui permet de modéliser tout type de données de référence et d'y appliquer une gouvernance grâce à des outils avancés comme le workflow collaboratif, le contrôle de l'édition de données, la gestion hiérarchique des données, le contrôle de version, et la sécurité.

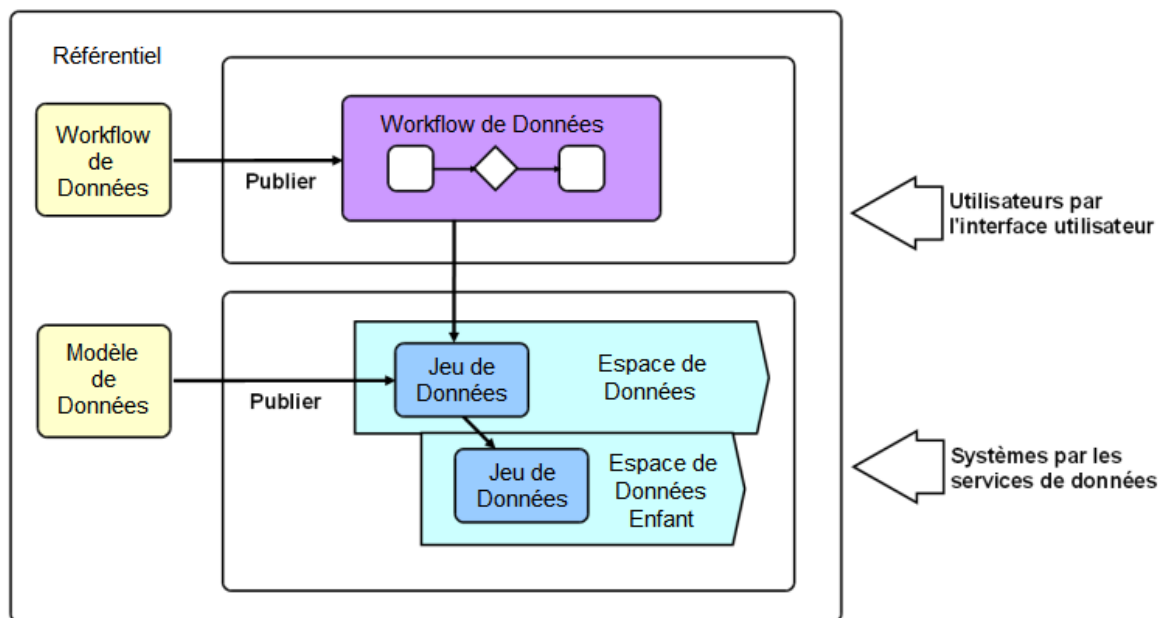
Un projet MDM qui utilise EBX5 commence par la création d'un *modèle de données*. Celui-ci définit les tables, les champs, les liens et les règles métier permettant de décrire les données de référence. De bons exemples sont les catalogues de produits, les hiérarchies financières, les listes de fournisseurs ou simplement les tables de référence.

Ce modèle de données peut ensuite être publié en tant que *jeu de données*, stockant le contenu des données de référence. Les jeux de données sont organisés dans des *espaces de données*. Un espace de données est un conteneur qui permet d'isoler toutes les mises à jour qui sont effectuées. Cela permet de travailler sur plusieurs versions parallèles des données.

Les *workflows* sont indispensables pour effectuer les processus de modification et d'approbation sur les données. Un workflow permet de modéliser un processus étape par étape, comprenant la participation de plusieurs utilisateurs humains et automatisés.

Les *modèles de workflow* définissent les tâches à effectuer, ainsi que les participants associés à chaque tâche. Dès qu'un modèle de workflow est publié, il peut être exécuté en tant que *workflow de données*. Les workflows de données permettent d'envoyer aux utilisateurs des notifications concernant les événements pertinents et les tâches à accomplir, le tout dans un contexte collaboratif.

Les *services de données* aident à intégrer EBX5 à des systèmes tiers ("middleware"), en leur permettant d'accéder aux données, ou de gérer des espaces de données et des workflows.



Voir aussi

[Modèle de données](#) [p 18]

[Jeu de données](#) [p 19]

[Espace de données](#) [p 21]

[Modèle de workflow](#) [p 22]

[Workflow de données](#) [p 23]

[Service de données](#) [p 24]

CHAPITRE 2

Interface utilisateur EBX5

Ce chapitre contient les sections suivantes :

1. [Présentation](#)
2. [Perspective avancée](#)
3. [Perspectives](#)
4. [Fonctionnalités de l'interface utilisateur](#)
5. [Où trouver de l'aide sur EBX5](#)

2.1 Présentation

La mise en page générale des espaces de travail sur EBX5 est entièrement personnalisable par un administrateur. Si des perspectives personnalisées existent, elles peuvent être sélectionnées via le menu déroulant 'Perspective' qui permet de passer d'une vue à l'autre.

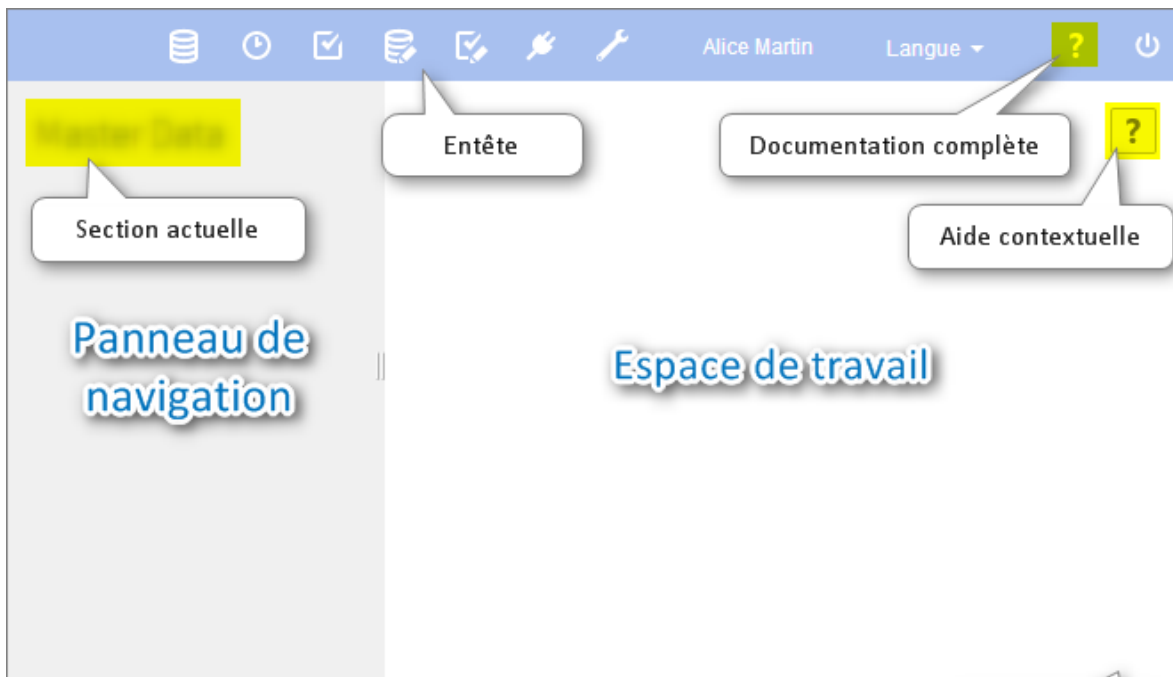
La perspective avancée est accessible par défaut.

2.2 Perspective avancée

Par défaut, la perspective avancée d'EBX5 est accessible à tous les utilisateurs. Cependant, son accès peut être restreint à certains profils uniquement. Cette vue est divisée en plusieurs zones principales, référencées dans la documentation sous les termes suivants:

- **En-tête** : le nom de l'utilisateur actuel s'affiche dans cette zone, ainsi que le menu de sélection de la langue (lorsque plusieurs sont disponibles), le menu de sélection des perspectives (lorsque plusieurs sont disponibles), un lien vers la documentation et un bouton pour fermer la session en cours.
- **Barre de menu** : cette zone comprend toutes les fonctionnalités accessibles à l'utilisateur actuel et lui permet de naviguer entre elles.
- **Panneau de navigation** : cette zone résume visuellement les diverses possibilités de navigation. Par exemple : sélectionner une table dans un jeu de données, ou un bon de travail dans un workflow.
- **Espace de travail** : zone de travail principale dépendant du contexte. Par exemple, la table sélectionnée dans le panneau de navigation s'affiche dans l'espace de travail, ou bien un bon de travail en cours s'y exécute.

Les sections fonctionnelles suivantes sont affichées dans l'interface selon les permissions de l'utilisateur actuel : *Données*, *Espace de données*, *Modélisation*, *Workflow de données*, *Services de données*, et *Administration*.

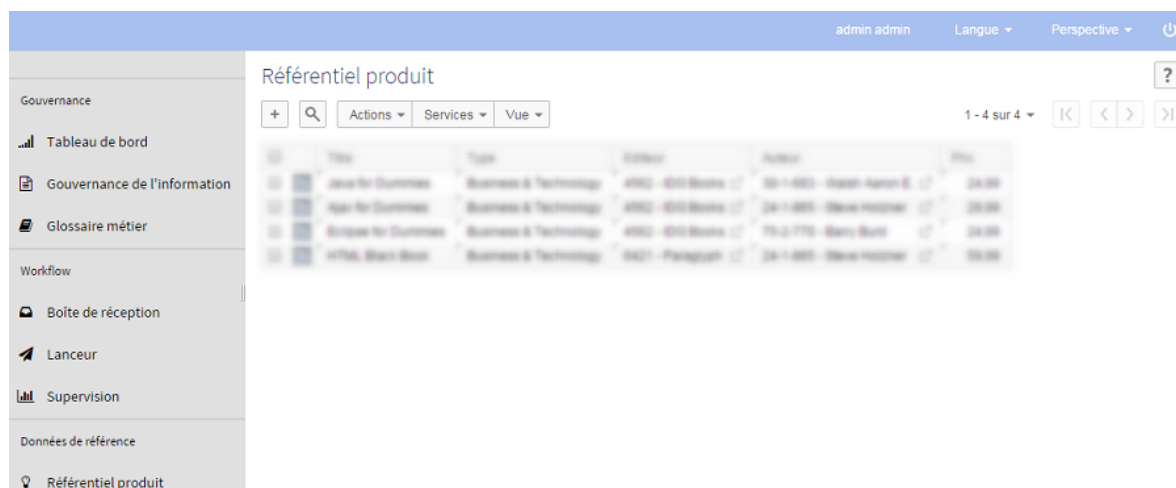


2.3 Perspectives

Les perspectives dans EBX5 sont des vues configurables avec une audience définie. Les perspectives permettent aux utilisateurs métier de bénéficier d'une interface simplifiée. Une perspective peut être affectée à un ou plusieurs profils. Cette vue est divisée en plusieurs zones principales, référencées dans la documentation sous les termes suivants:

- **En-tête** : le nom de l'utilisateur actuel s'affiche dans cette zone, ainsi que le menu de sélection de la langue (lorsque plusieurs sont disponibles), le menu de sélection des perspectives (lorsque plusieurs sont disponibles) et un bouton pour fermer la session en cours.
- **Panneau de navigation** : cette zone affiche le menu hiérarchique tel qu'il a été configuré par l'administrateur de perspectives. Ce panneau peut être développé ou réduit et permet d'accéder aux entités et services correspondant à l'activité de l'utilisateur.
- **Espace de travail** : zone de travail principale dépendant du contexte.

Exemple de présentation d'un menu hiérarchique :



2.4 Fonctionnalités de l'interface utilisateur

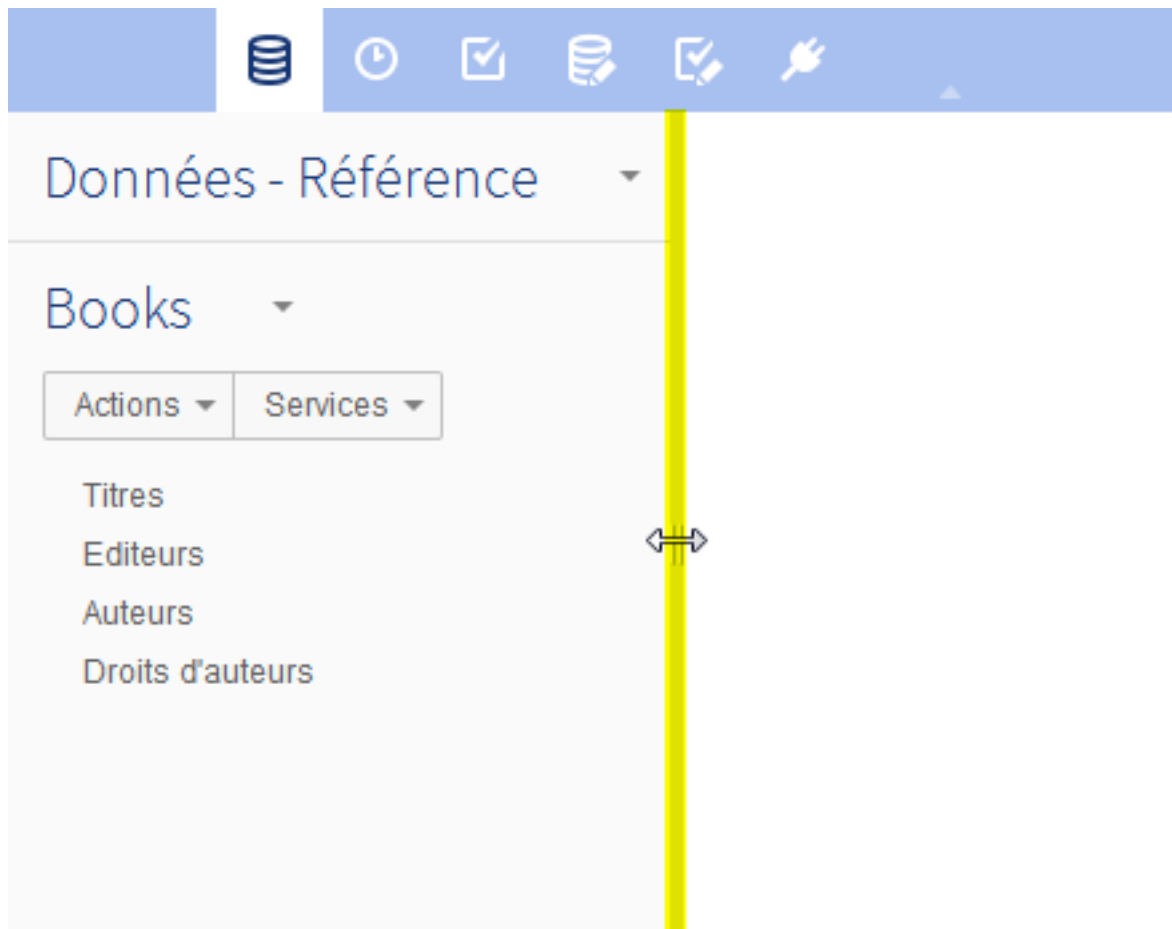
Cacher l'en-tête

L'en-tête de l'interface utilisateur peut être caché en le survolant, puis en cliquant sur le bouton en forme de flèche.



Réinitialiser la largeur du panneau de navigation

Si la largeur du panneau de navigation a été modifiée, elle peut être réinitialisée en double-cliquant sur la bordure.



2.5 Où trouver de l'aide sur EBX5

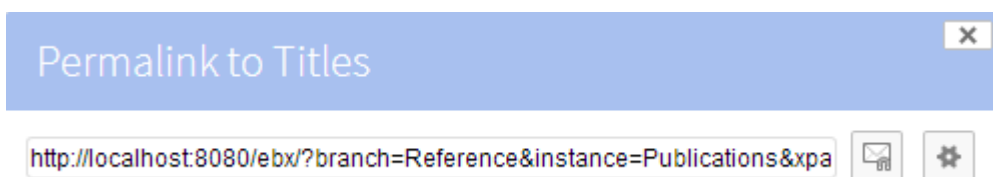
En plus de la documentation complète du produit, l'aide est accessible de plusieurs façons dans l'interface :

Aide contextuelle

Lorsque la souris survole un élément pour lequel une aide spécifique existe, un point d'interrogation apparaît. En cliquant sur le point d'interrogation, un panneau de documentation affiche les informations associées.



Un bouton dans le coin supérieur gauche du panneau permet de récupérer un permalien vers l'élément. Ce bouton n'est pas disponible pour tous les types d'élément.



CHAPITRE 3

Glossaire

Ce chapitre contient les sections suivantes :

1. [Gouvernance](#)
2. [Modélisation de données](#)
3. [Gestion des données](#)
4. [Cycle de vie des données](#)
5. [Historique](#)
6. [Modélisation de workflow](#)
7. [Workflows de données](#)
8. [Services de données](#)
9. [Transverse](#)

3.1 Gouvernance

référentiel

Entité de stockage côté serveur contenant toutes les données gérées par EBX5. Le référentiel est organisé en espaces de données.

Voir [espace de données](#) [p 21].

profil

Terme générique désignant soit un utilisateur, soit un rôle. Les profils sont utilisés pour définir des règles de permission et des workflows de données.

Voir [utilisateur](#) [p 17], [rôle](#) [p 18].

utilisateur

Entité créée dans le référentiel afin de permettre à des personnes physiques ou des systèmes externes de s'authentifier et d'accéder à EBX5. Un utilisateur peut avoir un ou plusieurs rôles et posséder diverses informations de compte telles que nom, prénom, login, e-mail, etc.

rôle

Classification d'utilisateur utilisée pour les règles de permission et les workflows de données. Chaque utilisateur peut appartenir à plusieurs rôles.

Dès qu'un profil de type rôle est configuré dans EBX5, le comportement résultant de cette configuration s'applique à tous les utilisateurs membres de ce rôle. Par exemple, dans un modèle de workflow, un rôle peut être configuré pour le(s) destinataire(s) d'un bon de travail.

3.2 Modélisation de données

Section de la documentation [Modèles de données](#) [p 28]

modèle de données


Définition structurée des données à gérer dans le référentiel EBX5. Un modèle de données décrit la structure des données en termes d'organisation, de type et de relations sémantiques. Le but d'un modèle de données est de définir la structure et les caractéristiques d'un jeu de données, qui est une instance d'un modèle de données contenant les données gérées par le référentiel.

Voir [jeu de données](#) [p 20].

Concept apparenté : [Modèles de données](#) [p 28].

champ

Élément de base du modèle de données défini par un nom et un type de données simple. Un champ peut être directement défini à la racine du modèle de données, ou en tant que colonne d'une table. Il est possible d'assigner des contraintes de base sur la valeur du champ, par exemple sur sa longueur, ainsi que des règles de validation plus complexes impliquant des calculs. La valeur du champ peut être automatiquement calculée à l'aide du mécanisme d'héritage de données, ou de règles de calcul. Un champ peut être défini comme étant une liste agrégée en spécifiant une cardinalité maximale supérieure à 1. Chaque élément de la liste ainsi définie sera du même type que le champ initial. Les champs peuvent être regroupés pour faciliter l'organisation du modèle de données.

Par défaut, les champs sont représentés par l'icône .

Voir [enregistrement](#) [p 19], [groupe](#) [p 19], [table \(modèle de données\)](#) [p 19], [règle de validation](#) [p 19], [héritage](#) [p 20].

Concept apparenté : [Propriétés des éléments de structure](#) [p 41], [Contrôles sur les champs de données](#) [p 47].

clé primaire

Champ ou composition de plusieurs champs identifiant de manière unique un enregistrement dans une table.

Les clés primaires sont représentées par l'icône .

clé étrangère

Champ ou composition de plusieurs champs référençant un enregistrement d'une autre table, via sa clé primaire.

Les clés étrangères sont représentées par l'icône .

Voir [clé primaire](#) [p 18].

table (modèle de données)

Élément du modèle de données composé de champs et/ou de groupes. Une table doit au moins être composée d'un champ défini comme étant une clé primaire. Une table peut être utilisée pour la création d'un type réutilisable, afin de créer d'autres éléments basés sur la structure de cette table.

Les tables sont représentées par l'icône .

Voir [enregistrement](#) [p 19], [clé primaire](#) [p 18], [type réutilisable](#) [p 19].

groupe

Entité de classification utilisée pour organiser les données du modèle. Un groupe peut contenir des champs, d'autres groupes, et des tables. Si un groupe contient des tables, alors celui-ci ne pourra pas être inclus dans une autre table. Un groupe peut être utilisé pour la création d'un type réutilisable afin de créer d'autres éléments basés sur la structure de ce groupe.

Les groupes sont représentés par l'icône .

Voir [type réutilisable](#) [p 19].

type réutilisable

Définition d'un type simple ou complexe qui peut être partagée entre différents éléments d'un modèle.

règle de validation

Association d'une ou plusieurs règles de contrôle définies sur un champ ou une table. Toute donnée saisie ne respectant pas ces contrôles sera déclarée invalide, selon la sévérité associée à la règle de validation.

assistant de modélisation de données (DMA)

L'interface utilisateur inclut un outil d'aide à la modélisation des données. Cet outil permet de définir la structure d'un modèle, de créer et éditer ses éléments, puis de configurer et publier le modèle.

Voir [Modèles de données](#) [p 28].

Gestion des données

Section de la documentation [Jeux de données](#) [p 80]

enregistrement

Ensemble de données identifié de manière unique par une clé primaire. Un enregistrement correspond à une ligne dans une table. Chaque enregistrement respecte la structure de données définie dans le

modèle de données associé. C'est ce modèle de données qui indique les types et les cardinalités des champs qui composent l'enregistrement.

Les enregistrements sont représentés par l'icône .

Voir [table \(jeu de données\)](#) [p 20], [clé primaire](#) [p 18].

table (jeu de données)

Ensemble d'enregistrements (lignes) de même structure contenant des données. Chaque enregistrement est identifié de manière unique par sa clé primaire.

Les tables sont représentées par l'icône .

Voir [enregistrement](#) [p 19], [clé primaire](#) [p 18].

jeu de données

Instance d'un modèle de données qui contient les données. La structure et le comportement d'un jeu de données sont basés sur les définitions fournies par le modèle de données qu'il implémente. En fonction de son modèle de données, un jeu de données peut contenir des données sous la forme de tables, groupes et champs.

Voir [table \(jeu de données\)](#) [p 20], [champ](#) [p 18], [groupe](#) [p 19], [vue personnalisée](#) [p 20].


Les jeux de données sont représentés par l'icône .

Concept apparenté : [Jeux de données](#) [p 80].

héritage

Mécanisme par lequel une donnée d'une entité peut être valorisée par défaut à partir d'une autre entité. Dans EBX5, deux types d'héritage sont possibles : le premier entre deux jeux de données, le deuxième entre deux champs.

Lorsqu'il est activé, l'héritage entre jeux de données permet à un jeu de données enfant d'obtenir comme valeur par défaut les données du jeu de données parent. Il est possible de surcharger dans les jeux de données enfants les valeurs héritées du parent. Par défaut, l'héritage est désactivé. Il peut être activé lors de la définition du modèle de données.

L'héritage depuis le jeu de données parent est représenté par l'icône .

L'héritage de champ fonctionne de manière similaire, mais s'applique entre champs d'un même jeu de données. Le champ hérité prend pour valeur par défaut la valeur du champ source.

Les champs hérités sont représentés par l'icône .

vue personnalisée

Configuration d'affichage applicable sur une table. Une vue peut être créée pour un utilisateur ou un rôle. La vue personnalisée permet de sélectionner sous quel mode seront affichés les enregistrements : mode hiérarchique ou tabulaire ; ainsi que de définir des critères de filtrage et de tri.

Voir [vue hiérarchique](#) [p 21].

Concept apparenté : [Vues](#) [p 87].

vue hiérarchique

Vue personnalisée qui affiche les données d'une table sous forme d'arbre. Une vue hiérarchique peut être utile pour montrer les relations entre les données du modèle. Lors de la création d'une vue hiérarchique, une dimension doit être sélectionnée pour déterminer les relations à exploiter. Dans une vue hiérarchique, il est possible de naviguer à travers des relations récursives, ainsi qu'entre plusieurs tables en utilisant des clés étrangères.

Voir [vue personnalisée](#) [p 20].


Concept apparenté : [Hiérarchies](#) [p 88].

3.4 Cycle de vie des données

Section de la documentation [Espaces de données](#) [p 62]

espace de données

Contient les jeux de données. L'espace de données est utilisé pour isoler différentes versions de jeux de données ou pour les organiser. Des espaces de données enfants peuvent être créés à partir d'un espace de données. Un espace de données enfant est initialisé dans le même état que son parent au moment de sa création. Ultérieurement, l'enfant pourra être fusionné avec son parent. A tout moment, une comparaison avec d'autres espaces de données est possible.

Les espaces de données sont représentés par l'icône .

Voir [héritage](#) [p 20], [référentiel](#) [p 17], [fusion](#) [p 21].

Concept apparenté : [Data spaces](#) [p 62].

espace de données de référence

Ancêtre commun des espaces de données du référentiel. N'ayant pas de parent, cet espace de données ne peut pas être fusionné.

Voir [espace de données](#) [p 21], [fusion](#) [p 21], [référentiel](#) [p 17].

fusion

Intégration, dans l'espace de données parent, des changements réalisés dans un espace de données enfant depuis sa création. L'espace de données enfant est fermé après une fusion réalisée avec succès. Pour effectuer cette fusion, un passage en revue des différences entre les deux espaces de données est requis afin de résoudre les éventuels conflits. En effet, des conflits peuvent survenir en cas de modification des mêmes données tant sur l'enfant que le parent. Une décision doit être prise pour chacun de ces conflits, afin de déterminer quelle modification doit prendre le pas sur l'autre.

Concept apparenté : [Fusion](#) [p 70].

image

Copie statique d'un espace de données qui capture son état et tout son contenu à un moment donné, afin d'être utilisée comme référence. Une image peut être consultée, exportée, et comparée à d'autres espaces de données, mais jamais modifiée directement.

Les images sont représentées par l'icône .

Concept apparenté : [Image](#) [p 75]

3.5 Historique

historisation

Mécanisme qui peut être activé au niveau d'une table afin de suivre les modifications dans le référentiel. Deux vues d'historique sont disponibles quand l'historisation est activée : la vue historique de table et la vue historique des transactions. Dans toutes les vues d'historique, les fonctionnalités classiques des tables telles que l'export, la comparaison et les filtres restent disponibles.

L'activation de l'historique nécessite la configuration d'un profil d'historisation. L'historisation des tables n'est pas activée par défaut.

Voir [vue historique de table](#) [p 22], [vue historique des transactions](#) [p 22], [profil d'historisation](#) [p 22].

profil d'historisation

Ensemble de préférences spécifiant d'une part les espaces de données dont les modifications doivent être enregistrées dans l'historique de la table et, d'autre part, si les transactions doivent échouer quand l'historisation n'est pas disponible.

Voir [profil d'historisation](#) [p 22].

vue historique de table

Vue contenant la trace de toutes les modifications effectuées sur une table donnée, notamment les créations, mises à jour et suppressions. Chaque entrée présente les informations transactionnelles telles que : la date et l'heure, l'utilisateur ayant effectué l'action, ainsi que l'état des données à l'issue de la transaction. Ces informations peuvent aussi être consultées au niveau d'un enregistrement ou d'un jeu de données.

vue historique des transactions

Vue présentant les données techniques et d'authentification des transactions au niveau du référentiel ou d'un espace de données. Etant donné qu'une transaction peut effectuer de multiples opérations/actions et peut affecter plusieurs tables dans un ou plusieurs jeux de données, cette vue montre toutes les opérations qui ont été effectuées dans le périmètre en question pour chaque transaction.

3.6 Modélisation de workflow

Section de la documentation [Modèles de workflow](#) [p 104]

modèle de workflow


Définition de la succession d'opérations à effectuer sur les données.

Un modèle de workflow de données décrit la totalité du parcours que doivent suivre les données pour être traitées, que ce soit en termes d'états ou d'actions associées à effectuer par des utilisateurs et des tâches automatiques.

Concept apparenté : [Modèles de workflow](#) [p 104].

tâche automatique

Tâche de workflow de données effectuée par une procédure automatique, sans intervention humaine. Les tâches automatiques les plus communes sont la création d'espace de données, la fusion d'espace de données et la création d'image.

Les tâches automatiques sont représentées par l'icône .

Voir [modèle de workflow](#) [p 22].

tâche utilisateur

Tâche de workflow composée d'un ou plusieurs bons de travail réalisés en parallèle par des utilisateurs (intervention humaine).

Les bons de travail sont proposés ou assignés aux utilisateurs, en fonction du modèle de workflow de données. L'avancement du workflow de données positionné sur une tâche utilisateur dépend de la satisfaction du critère de fin de tâche défini dans le modèle de workflow de données.

Les tâches utilisateur sont représentées par l'icône .

Voir [modèle de workflow](#) [p 22].

condition de workflow

Etape de décision dans le workflow de données.

Une condition de workflow de données décrit le critère utilisé pour déterminer quelle sera la prochaine étape à exécuter.

Les conditions de workflow sont représentées par l'icône .

appel à des sous-workflows

Etape qui met le workflow de données courant en attente et qui lance un ou plusieurs autres workflows de données. Si une telle étape lance plusieurs sous-workflows, les sous-workflows sont exécutés en parallèle.

tâche d'attente

Etape d'un workflow de données qui met en pause le workflow en cours en attendant un événement donné. Lorsque l'événement est reçu, le workflow est réveillé et va automatiquement à l'étape suivante.

contexte des données

Ensemble de données qui peuvent être partagées entre les étapes pendant toute la durée de vie d'un workflow afin de garantir la communication entre les étapes.

3.7 Workflows de données

Section de la documentation [Workflows de données](#) [p 128]

publication de workflow

Version particulière d'un modèle de workflow de données qui est mise à disposition des utilisateurs ayant les permissions nécessaires pour exécuter des workflows.

workflow de données

Instance particulière d'un modèle de workflow qui exécute les étapes définies dans le modèle de workflow de données (les tâches utilisateur, les tâches automatiques et les conditions).

Voir [modèle de workflow](#) [p 22].

Concept apparenté : [Workflows de données](#) [p 128].

corbeille

Liste des workflows de données publiés affichés en fonction des permissions de l'utilisateur. Les utilisateurs qui ont la permission de lancer des workflows peuvent le faire à partir de leur corbeille. Tous les bons de travail nécessitant une action de l'utilisateur sont affichés sous la publication de workflow associée dans la corbeille. De plus, si l'utilisateur est administrateur de workflows de données, il a la possibilité de voir leur état dans sa corbeille, et ainsi d'intervenir, si nécessaire, sur les workflows qu'il supervise.

Voir [workflow de données](#) [p 24]

bon de travail

Action unitaire d'une tâche utilisateur qui doit être réalisée par un utilisateur.

Les bons de travail alloués sont représentés par l'icône .

Voir [tâche utilisateur](#) [p 23].

3.8 Services de données

Section de la documentation [Services de données](#) [p 146]

service de données

EBX5 partage les données de référence conformément à l' [Architecture Orientée Service](#) en utilisant la technologie XML web service. Tous les services de données sont générés directement à partir des modèles ou services built-in. Ils peuvent être utilisés pour accéder à une partie des fonctionnalités disponibles via l'interface utilisateur.

Les services de données d'EBX5 proposent :

- Un générateur WSDL pour les modèles de données ainsi que pour les services built-in. Le fichier WSDL peut-être produit indifféremment au travers de l'interface utilisateur ou du connecteur HTTP(S) pour un logiciel d'intégration ou une application cliente. Les opérations de services sont invoquées par des messages XML communiqués au point d'entrée d'EBX5.
- Un connecteur SOAP ou point d'entrée pour les messages SOAP permet aux systèmes externes d'interagir avec le contenu du référentiel. Ce connecteur répond aux demandes issues des WSDL produits par EBX5. Après authentification, il accepte les messages XML et les exécute selon les permissions de l'utilisateur authentifié.

lignage

Mécanisme de mise en place de profils de droits d'accès pour des services de données. Les profils de droits d'accès ainsi définis sont utilisés pour accéder aux données via des interfaces WSDL.

Concept apparenté : [Générer un WSDL pour un lignage](#) [p 151].

3.9 Transverse

noeud

Un noeud est un élément d'une arborescence ou d'un graphe. Dans EBX5, 'Noeud' peut faire référence à différents concepts selon le contexte d'utilisation :

- Dans le contexte du [modèle de workflow](#) [p 22], un noeud est une étape du workflow ou une condition.
- Dans le contexte du [modèle de données](#) [p 18], un noeud est un groupe, une table ou un champ.
- Dans le contexte des [hiérarchies](#) [p 21], un noeud représente la valeur d'une dimension.
- Dans un arbre d'[héritage de jeux de données](#) [p 20], un noeud est un jeu de données.
- Dans un [jeu de données](#) [p 19], un noeud est le noeud du modèle de données évalué dans le contexte du jeu de données ou de l'enregistrement.

Modèles de données

CHAPITRE 4

Introduction aux modèles de données

Ce chapitre contient les sections suivantes :

1. [Présentation](#)
2. [Utilisation de l'interface utilisateur de la section Modèles de données](#)

4.1 Présentation

La fonction d'un modèle de données

La première étape de toute gouvernance de données dans EBX5 est le développement d'un modèle de données. Le but d'un modèle de données est de définir la structure des données gérées dans le référentiel en termes d'organisation, de types de données, et de relations sémantiques. Une fois que le modèle de données a été défini et publié, il devient possible de créer des jeux de données à partir de celui-ci.

Afin de définir un modèle de données dans le référentiel, créez d'abord un nouveau modèle de données, puis définissez sa structure et les propriétés de ses éléments (tables, champs et groupes). Le modèle de données ainsi défini doit être publié pour devenir disponible. Les utilisateurs pourront créer des jeux de données à partir de cette publication qui contiendront les données gérées par le référentiel EBX5.

Concepts de base utilisés dans la modélisation des données

Une compréhension des termes suivants est nécessaire pour commencer la création de modèles de données :

- [champ](#) [p 18]
- [clé primaire](#) [p 18]
- [clé étrangère](#) [p 19]
- [table](#) [p 19]
- [groupe](#) [p 19]
- [type réutilisable](#) [p 19]
- [règle de validation](#) [p 19]

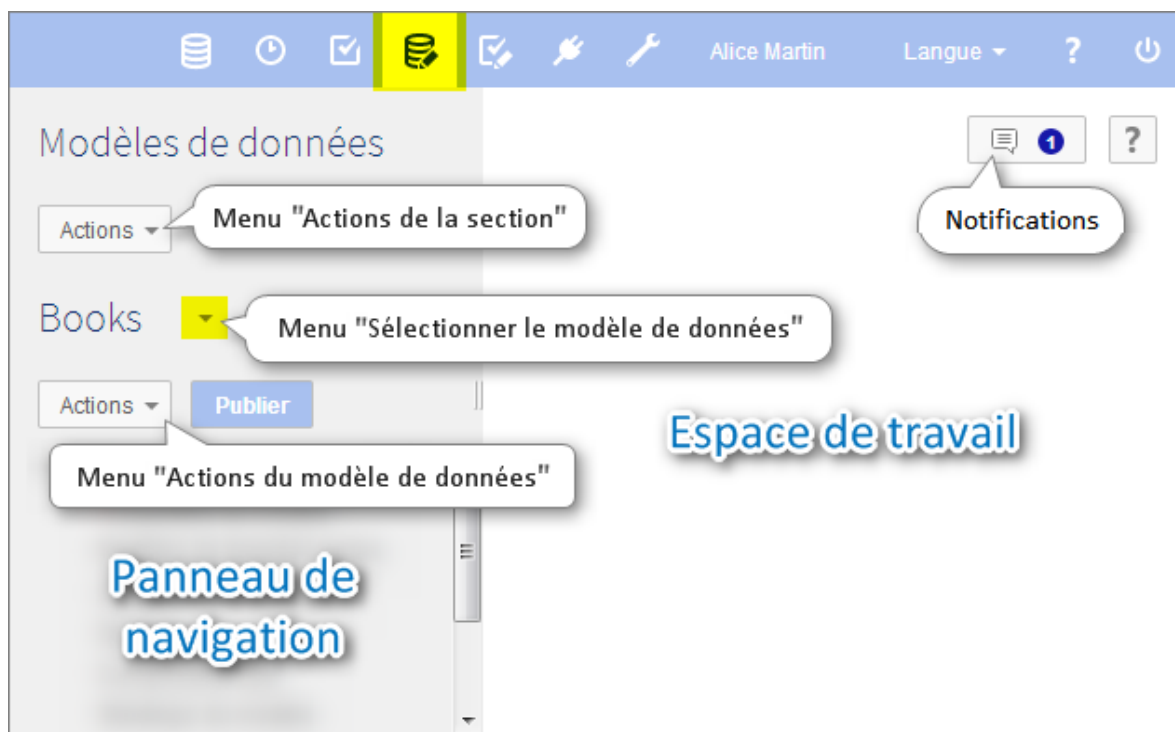
4.2 Utilisation de l'interface utilisateur de la section Modèles de données

Navigation dans le Data Model Assistant

Les modèles de données peuvent être importés, édités, et publiés dans la section **Modèles de données**. Le Data Model Assistant de EBX5 facilite le développement des modèles de données.

Note

Seuls les utilisateurs autorisés peuvent accéder à cet écran via la 'Perspective avancée'.



Le panneau de navigation est organisé selon les sections suivantes :

Configuration	La configuration technique du modèle de données.
Modèle de données inclus	Les modèles de données inclus dans le modèle courant. Les types de données définis dans les modèles inclus peuvent être réutilisés dans le modèle de données courant.
Structure de données	Structure du modèle de données. Définit les relations entre les éléments du modèle de données et permet d'accéder à la définition de chaque élément.
Types de données simples	Types simples réutilisables définis dans le modèle de données courant.
Types de données complexes	Types complexes réutilisables définis dans le modèle de données courant.
Types de données simples inclus	Types simples réutilisables définis dans un modèle de données inclus dans le modèle courant.
Types de données complexes inclus	Types complexes réutilisables définis dans un modèle de données inclus dans le modèle courant.

Voir aussi

[Modélisation de la structure des données](#) [p 35]

[Configuration du modèle de données](#) [p 33]

[Types réutilisables](#) [p 37]

Icônes des éléments du modèle de données



[champ](#) [p 18]



[clé primaire](#) [p 18]



[clé étrangère](#) [p 19]



[table](#) [p 19]



[groupe](#) [p 19]

Concepts apparentés

[Espaces de données](#) [p 62]

[Jeux de données](#) [p 80]

CHAPITRE 5

Création du modèle de données

Ce chapitre contient les sections suivantes :

1. [Création d'un modèle de données](#)

5.1 Création d'un modèle de données

Pour créer un modèle de données, cliquez sur le bouton **Créer** dans le sélecteur, puis suivez les instructions de l'assistant de création de modèle de données.

CHAPITRE 6

Configuration du modèle de données

Ce chapitre contient les sections suivantes :

1. [Informations associées au modèle de données](#)
2. [Permissions](#)
3. [Propriétés du modèle de données](#)
4. [Modèles de données inclus](#)

6.1 Informations associées au modèle de données

Pour visualiser et éditer les informations concernant le propriétaire et la documentation du modèle de données, sélectionnez "Informations" dans le menu ["Actions" du modèle de données](#) [p 29] dans le panneau de navigation.

Note

Seuls les utilisateurs autorisés peuvent accéder à cet écran via la 'Perspective avancée'.

Nom unique	Le nom unique du modèle de données. Ce nom ne peut pas être modifié après la création du modèle.
Propriétaire	Spécifie le propriétaire du modèle de données, qui a le droit de modifier les informations du modèle et ses permissions.
Documentation localisée	Libellés et descriptions localisés pour le modèle de données.

6.2 Permissions

Pour définir les permissions d'accès au modèle de données, sélectionnez "Permissions" dans le menu ["Actions" du modèle de données](#) [p 29] dans le panneau de navigation.

La définition des permissions d'un modèle de données s'effectue de la même manière que pour les jeux de données. Les détails se trouvent dans la section [Permissions](#) [p 96].

6.3 Propriétés du modèle de données

Dans le panneau de navigation, sous Configuration > Propriétés du modèle, vous pouvez accéder aux propriétés techniques suivantes :

Héritage des données	Spécifie si l'héritage des données, entre jeux de données, est activé pour ce modèle de données. L'héritage des données est désactivé par défaut. Voir héritage [p 99] pour plus d'informations.
Désactiver les contrôles de l'auto-incrément	Indique si le contrôle de la valeur d'un champ auto-incrémenté par rapport à la valeur maximale trouvée dans la table en cours de mise à jour doit être désactivé.

6.4 Modèles de données inclus

Les types de données définis dans un autre modèle de données peuvent être utilisés dans le modèle de données courant en ajoutant une entrée pour l'autre modèle de données dans la table Configuration > Modèles de données inclus.

En accédant à l'enregistrement du modèle inclus depuis cette table, des informations techniques liées à ce modèle sont consultables sous l'onglet **Information**. Cet onglet contient aussi le rapport de validation du modèle de données inclus.

Seuls les modèles de données sans erreur de validation, qui ont été définis et publiés comme modèle "embarqué" ou dans un module, peuvent être inclus.

Les types de données doivent être uniques à la fois dans le modèle courant et dans tous les modèles inclus. Il est impossible d'inclure un modèle contenant des types de données déjà existant dans le modèle courant ou dans les autres modèles de données inclus.

CHAPITRE 7

Modélisation de la structure des données

Pour définir la structure du modèle de données, sélectionnez le modèle de données avec lequel vous voulez travailler dans le panneau de navigation.

La structure du modèle de données est accessible depuis le panneau de navigation dans la section "Structure de données". Cette section permet de visualiser et de définir la structure des champs, groupes, et tables du modèle de données.

Ce chapitre contient les sections suivantes :


1. [Actions et propriétés communes](#)
2. [Types réutilisables](#)
3. [Détails de la création des éléments du modèle de données](#)
4. [Modification des éléments existants](#)

7.1 Actions et propriétés communes

Créer des éléments

Les éléments suivants peuvent être ajoutés à un modèle de données :

- champs
- groupes
- tables
- clés primaires
- clés étrangères
- associations

Ajoutez un de ces éléments sous un élément existant en cliquant sur la flèche  située à la droite de l'élément existant, puis en sélectionnant une option de création parmi les options présentées dans le menu. Suivez ensuite l'assistant de création pour créer un élément.

Note


L'élément root est ajouté par défaut lors de la création d'un modèle de données. Cet élément représente la racine de la structure du modèle de données. S'il faut renommer cet élément, il peut être supprimé et recréé avec un nom différent.

Noms, libellés, descriptions, et informations

Le nom de l'élément à créer est obligatoire. Ce nom doit être unique au sein d'un même niveau dans la structure de données. En effet, sous un même groupe, deux éléments ne peuvent avoir le même nom. Une fois l'élément créé, son nom ne peut plus être modifié.


Il est possible de définir des libellés localisés qui seront affichés dans l'interface utilisateur au lieu du nom unique de l'élément. Il est aussi possible de définir une description localisée de l'élément. Contrairement au nom de l'élément, les libellés et descriptions sont modifiables après la création de l'élément. Selon la préférence de langue de chaque utilisateur, EBX5 affichera le libellé et la description localisés de l'élément.

Supprimer des éléments

Tous les éléments du modèle de données peuvent être supprimés de la structure de données en utilisant la flèche  située à la droite de l'élément à supprimer.

Si un groupe ou une table n'utilisant pas un type réutilisable est supprimé, la suppression est effectuée récursivement sur tous les éléments situés sous le groupe ou la table.

Dupliquer des éléments existants


Pour dupliquer un élément, cliquez sur la flèche  située à la droite de l'élément à dupliquer. Spécifiez le nom de l'élément dupliqué. Ce nom doit être unique au sein d'un même niveau dans la structure de données. Toutes les propriétés de l'élément source sont dupliquées.

L'élément dupliqué est rajouté dans le modèle de données au même niveau que l'élément d'origine, en dernière position. Lorsqu'un élément contenant d'autres éléments est dupliqué, tous les sous-éléments sont dupliqués avec leurs noms originaux.

Note

En cas de duplication d'un champ appartenant à une clé primaire, les propriétés du champ sont dupliquées mais le nouveau champ n'est pas ajouté à la clé primaire de la table parente.

Déplacer des éléments

Pour déplacer un élément, cliquez sur la flèche  située à la droite de l'élément à déplacer et sélectionnez "Déplacer". Sélectionnez ensuite la flèche qui correspond à l'élément *avant lequel* positionner l'élément actuel.

Note

Le déplacement d'un élément est uniquement possible au sein d'un même niveau dans la structure de données du modèle.


7.2 Types réutilisables

Les types réutilisables sont des éléments partagés qui, après leur création, peuvent être réutilisés par différents éléments du modèle de données.

Note


En modifiant la définition d'un type réutilisable, la structure de tous les éléments basés sur ce type réutilisable est aussi modifiée. L'arborescence "Structure de données" affiche, en lecture seule, la structure d'un groupe ou d'une table qui utilise un type réutilisable. Pour éditer la structure du type réutilisable associé, accédez au type dans la section "Types de données simples" ou "Types de données complexes".

Définition d'un type réutilisable

En utilisant le menu avec la flèche  des sections "Types de données simples" ou "Types de données complexes" dans le panneau de navigation, il est possible de créer des types simples et des types complexes réutilisables qui seront disponibles pour créer d'autres éléments avec la même structure et les mêmes propriétés. Il est également possible de convertir les tables et groupes existants en types

réutilisables en utilisant le menu avec la flèche  situé à côté de l'élément à convertir.

Il est possible de visualiser les éléments du modèle utilisant un type réutilisable, en éditant ce type et en cliquant sur le lien "Références vers ce type". Ce lien affiche une table listant tous les éléments utilisant ce type. Si le type n'est utilisé par aucun élément, il peut être supprimé en sélectionnant

"Supprimer type" en utilisant le menu avec la flèche  situé à droite du type à supprimer.

Utilisation d'un type réutilisable

Les structures et les propriétés de nouveaux éléments peuvent être définies par des types réutilisables en sélectionnant un type réutilisable à la création d'un élément. L'élément créé utilisera la structure et les propriétés du type réutilisable.

Inclusion des types de données définis dans d'autres modèles de données

Les types réutilisables peuvent aussi être partagés entre plusieurs modèles de données. En configurant l'inclusion d'un modèle de données externe, il est possible d'utiliser les types de données inclus pour créer des éléments dans la structure de données, de la même manière que pour les types réutilisables définis en local.

Note

Les types de données devant être uniques pour tous les types définis en local et inclus, il n'est pas possible de créer un type réutilisable portant le même nom qu'un type de données dans un modèle de données inclus. De la même manière, il n'est pas possible d'inclure un modèle de données externe qui définit un type de données portant le même nom qu'un type réutilisable défini en local ou dans un autre modèle de données inclus.

Les types de données inclus apparaissent dans les sections "Types de données simples inclus" et "Types de données complexes inclus" dans le panneau de navigation. Les détails de ces types réutilisables sont consultables, mais ils ne sont éditables que dans leurs modèles de données d'origine.

Voir [Modèles de données inclus](#) [p 34] pour plus d'informations.

7.3 Détails de la création des éléments du modèle de données

Création de champs

A la création d'un champ, un type de données doit être sélectionné. Il définira le type de données associé aux valeurs saisies dans un jeu de données basé sur ce modèle. Le type de données du champ ne peut pas être modifié après la création du champ.

Durant la création d'un champ, il est également possible de le désigner comme clé étrangère, champ obligatoire, ou comme clé primaire si le champ est créé sous une table.

Création de tables

Lors de la création d'une nouvelle table, un type réutilisable existant peut être utilisé pour définir la structure et les propriétés de cette nouvelle table. Voir [Types réutilisables](#) [p 37] pour plus d'informations.

Chaque table nécessite la désignation d'au moins un champ clé primaire, qui peut être créé comme un élément enfant de la table dans la section "Structure de données" du panneau de navigation.

Création de groupes

Lors de la création d'un groupe, il est possible d'utiliser un type réutilisable existant pour définir la structure et les propriétés du nouveau groupe. Voir [Types réutilisables](#) [p 37] pour plus d'informations.

Création de clés primaires

Pour chaque table, il est nécessaire de définir une clé primaire. Pour cela, ajoutez un nouvel élément enfant à partir du menu d'actions disponible sur la table dans la section "Structure de données" du panneau de navigation.

Il est aussi possible d'ajouter un champ existant à la définition de la clé primaire, sur l'onglet "Clé primaire" dans les "Propriétés avancées" de la table.

Création ou définition de clés étrangères

Les champs associés à une clé étrangère sont de type "Chaîne de caractères". Pour créer une clé étrangère sur une table, ajoutez un nouvel élément enfant à partir du menu d'actions disponible sur la table dans la section "Structure de données" du panneau de navigation. Il est également possible de définir directement les propriétés d'une clé étrangère en éditant un champ de type "Chaîne de caractères". Pour convertir un champ existant de type "Chaîne de caractères" en clé étrangère, activez la propriété "Contrainte de clé étrangère" dans les "Contrôles avancés" du champ et définissez les propriétés associées.

Il faut toujours spécifier la table référencée par une clé étrangère.

Création d'associations

Une association permet de définir un lien sémantique entre deux tables. Une association peut être définie en créant un élément dans une table de la section "Structure de données" du panneau de navigation en sélectionnant la propriété "association" dans le formulaire de création. Une association peut uniquement être créée dans une table et il n'est pas possible de convertir un champ existant en association.

Lors de la création d'une association, spécifiez son type. Pour cela, deux options sont disponibles :

- Relation inverse d'une *clé étrangère*. Dans ce cas, l'association est définie dans une *table source* et fait référence à une *table cible*. Ce type d'association est l'inverse d'une clé étrangère qui est définie dans la table cible et qui référence la table source. Définissez la clé étrangère qui référence la table contenant l'association. Pour cela, des assistants de saisie sont disponibles lors de la création de l'association.
- Utilisation d'une *table de liens*. Dans ce cas, l'association est définie dans une *table source* et fait référence à une *table cible* qui est inférée à partir d'une *table de liens*. Cette table de liens doit définir deux clés étrangères : une clé étrangère référençant la table source et une autre référençant la table cible. La clé primaire de la table de liens doit aussi être composée de champs auto-incrémentés et/ou de clés étrangères vers la table source ou cible de l'association. Définissez la table de liens et ces deux clés étrangères. Pour cela, des assistants de saisie sont disponibles à la création de l'association.

Pour ces deux types d'association, nous appelons *enregistrements associés* les enregistrements de la table cible sémantiquement liés aux enregistrements de la table source.

Une fois l'association créée, il est possible de définir d'autres propriétés :

- filtrer les enregistrements associés en définissant un filtre XPath. Il est uniquement possible d'utiliser les champs de la table source et de la table cible lors de la définition du filtre XPath. Il n'est donc pas possible d'utiliser les champs d'une table de liens dans un filtre XPath. Un assistant de saisie est disponible pour définir les champs à utiliser dans un filtre XPath.
- configurer une vue tabulaire pour définir les champs de la table cible devant être affichés dans les formulaires des jeux de données. Il n'est pas possible de configurer ou de modifier une vue tabulaire si la table cible n'est pas définie ou n'existe pas. Par défaut, tous les champs de la table cible qu'un utilisateur a le droit de voir seront affichés dans les jeux de données si la vue tabulaire n'est pas définie.
- définir comment doivent être présentés les enregistrements associés dans les formulaires des jeux de données. Indiquez si les enregistrements associés doivent être inclus dans le formulaire ou dans un onglet spécifique. Par défaut, les enregistrements associés seront inclus dans le formulaire au niveau de l'association dans le modèle de données.
- inclure / exclure les enregistrements associés dans les opérations de sélection de Data Service. Par défaut, les enregistrements ne sont pas inclus dans les opérations de sélection des Data Service.
- spécifier les nombres minimum et maximum d'enregistrements associés requis. Dans les jeux de données associés, un message de validation est ajouté lorsque les nombres minimum ou maximum d'enregistrements associés ne correspondent pas à ces critères. Par défaut, les nombres minimum et maximum d'enregistrements associés requis ne sont pas restreints.
- définir une contrainte de validation en utilisant un prédicat XPath pour restreindre les enregistrements associés. Il est uniquement possible d'utiliser les champs de la table source et de la table cible lors de la définition du prédicat XPath. Il n'est donc pas possible d'utiliser les champs d'une table de liens dans un prédicat XPath. Un assistant de saisie permet de sélectionner les champs à utiliser dans un prédicat XPath. Dans les jeux de données associés, un message de validation sera ajouté pour tout enregistrement associé ne vérifiant pas cette contrainte.

7.4 Modification des éléments existants

Suppression d'un champ de la clé primaire

Tout champ appartenant à la clé primaire peut être supprimé de la clé primaire d'une table sur l'onglet "Clé primaire" dans les "Propriétés avancées" de la table.

Voir [clé primaire](#) [p 18] dans le glossaire.

CHAPITRE 8

Propriétés des éléments du modèle de données

Après avoir créé un élément, il est possible de définir des propriétés supplémentaires pour compléter sa définition.

Voir aussi [Contrôles sur les éléments du modèle de données](#) [p 47]

Ce chapitre contient les sections suivantes :

1. [Propriétés basiques des éléments](#)
2. [Propriétés avancées des éléments](#)

8.1 Propriétés basiques des éléments

Propriétés basiques communes

Les propriétés basiques suivantes sont disponibles pour plusieurs types d'éléments :

Information	Informations supplémentaires non internationalisées associées à l'élément.
Nombre minimum de valeurs	<p>Nombre minimum de valeurs pour cet élément.</p> <p>Les clés primaires ne pouvant être multi-valuées, cette propriété doit être égale à "1" ou être "non définie".</p> <p>Pour les éléments de type "Noeud de sélection", le nombre minimum est automatiquement défini à "0".</p>
Nombre maximum de valeurs	<p>Nombre maximum de valeurs pour cet élément. Si cette propriété est supérieure à "1", l'élément est considéré comme multi-valué.</p> <p>Les clés primaires ne pouvant être multi-valuées, cette propriété doit être égale à "1" ou être "non définie".</p> <p>Pour une table, le nombre maximum d'éléments est défini à "unbounded" par défaut lors de sa création.</p> <p>Pour les éléments de type "Noeud de sélection", le nombre maximum d'éléments est défini à "0" par défaut lors de la définition des propriétés du noeud de sélection.</p>
Règles de validation	<p>Cette propriété est disponible pour les champs situés dans une table, sauf pour les champs de type Mot de passe, les types réutilisables, les champs des types complexes réutilisables et les noeuds de sélection. Cette propriété est utilisée pour définir des règles de validation riches et complexes à l'aide d'un éditeur de prédicats XPath 1.0.</p> <p>Une règle de validation peut être utile lorsque la validation de la valeur dépend de critères complexes ou des valeurs d'autres champs.</p> <p>En utilisant l'assistant associé, il est possible de définir des libellés localisés pour la règle de validation, ainsi qu'un message localisé avec la sévérité qui s'affichera si le critère n'est pas satisfait.</p>

Propriétés basiques des champs

Les propriétés basiques suivantes sont spécifiques aux champs simples :

Valeur par défaut	Définit une valeur par défaut pour ce champ. Cette valeur sera insérée automatiquement dans le champ de saisie du formulaire de création des nouveaux enregistrements. Le type de la valeur par défaut doit être compatible avec le type du champ courant.
Message d'erreur de conversion	Messages d'erreur internationalisés affichés aux utilisateurs lors de la saisie d'une valeur qui n'est pas compatible avec le type du champ courant.
Règle de calcul	<p>Cette propriété est disponible pour les champs des tables, excepté pour les types réutilisables. Définit une règle de calcul pour la valeur du champ avec l'aide d'un éditeur de prédicats XPath 1.0.</p> <p>Une règle de calcul peut être utile lorsqu'une valeur dépend d'autres valeurs dans le même enregistrement, mais qu'un calcul programmatique n'est pas nécessaire.</p> <p>Les limitations suivantes existent pour les règles de calcul :</p> <ul style="list-style-type: none">• Les règles de calcul s'utilisent uniquement avec les champs simples d'une table.• Les règles de calcul ne peuvent pas être définies sur des champs du type OResource ou Password.• Les règles de calcul ne peuvent pas être définies sur les noeuds de sélection ni les champs clés primaires.• Les règles de calcul ne peuvent pas être définies en accédant à l'élément depuis le rapport de validation.

8.2 Propriétés avancées des éléments

Propriétés avancées des champs

Les propriétés avancées suivantes sont spécifiques aux champs.

Auto-incrément

Cette propriété est disponible uniquement pour les champs de type "entier" contenus dans une table. Si elle est activée, la valeur du champ est calculée automatiquement lors de la création d'un nouvel

enregistrement. Peut être utile pour les clés primaires, car l'auto-incrément génère un identifiant unique pour chaque enregistrement. Deux attributs peuvent être spécifiés :

Valeur de démarrage	Valeur initiale de l'auto-incrément. Si aucune valeur n'est définie, la valeur par défaut est "1".
Pas de l'incrément	La valeur ajoutée à la valeur précédente de l'auto-incrément. Si aucune valeur n'est définie, la valeur par défaut est "1".
Désactiver les contrôles de l'auto-incrément	Indique s'il faut désactiver le contrôle de la valeur d'un champ auto-incrémenté par rapport à la valeur maximale trouvée dans la table en cours de mise à jour.

Champ hérité

Définit une relation entre le champ courant et un champ dans une autre table afin de chercher sa valeur automatiquement.

Enregistrement source	Une clé étrangère ou une séquence de clés étrangères, séparées par des espaces, permettant de trouver l'enregistrement dont hérite le champ courant. Si cette propriété n'est pas renseignée, alors l'élément courant sera utilisé comme source d'héritage de champ.
Élément source	Chemin XPath définissant l'élément à hériter. L'élément source doit être terminal, se trouver dans "Enregistrement source", et son type doit être le même que le type de ce champ. Cette propriété est obligatoire pour l'héritage de champ.

Voir [héritage](#) [p 20] dans le glossaire.

Propriétés avancées des tables

Les propriétés avancées suivantes sont spécifiques aux tables.

Table

Clé primaire	<p>Liste des champs composant la clé primaire de la table. Il est possible d'ajouter ou de supprimer des champs de la clé primaire.</p> <p>Chaque champ de la clé est identifié par un chemin XPath absolu qui débute sous la table.</p> <p>Si la clé est composite, la liste des champs doit être séparée par des espaces. Par exemple, <code>"/name /startDate"</code>.</p>
Présentation	<p>Spécifie la manière dont les enregistrements de cette table sont affichés dans l'interface utilisateur dans les jeux de données associés.</p>
Labellisation des enregistrements	<p>Définit les champs composant le libellé par défaut et les libellés localisés pour les enregistrements de la table.</p> <p>Attention : Les droits d'accès définis dans les jeux de données associés ne sont pas appliqués lors de l'affichage des libellés. Les champs habituellement cachés à cause de droits d'accès seront affichés dans les libellés des enregistrements.</p>
Présentation par défaut des groupes dans un formulaire	<p>Spécifie la politique d'affichage par défaut des groupes contenus dans cette table. Si cette propriété n'est pas définie, alors la politique par défaut sera utilisée pour afficher les groupes.</p> <p>Présentation activée des groupes : spécifie un mode de présentation autorisé en complément de "Développé" et "Réduit", qui sont toujours disponibles. Les liens doivent être activés sur la table afin de définir spécifiquement le mode de présentation des groupes en tant que liens.</p> <p>Présentation par défaut des groupes : spécifie la présentation par défaut des groupes contenus dans cette table. Si un groupe ne spécifie aucune vue par défaut, alors le mode de présentation par défaut défini par cette table sera utilisé. En fonction des performances du réseau et du navigateur, ajustez la manière d'afficher chaque groupe du formulaire. En ce qui concerne le poids de la page téléchargée, le mode "Lien" est léger, les modes "Développé" et "Réduit" sont plus lourds.</p> <p>Note : quand les onglets sont activés dans une table, tous les groupes qui utiliseraient les liens dans les autres cas sont convertis automatiquement en mode "Réduit". Cela permet d'éviter les complexités d'affichage qui se posent quand les liens et les onglets sont dans la même interface utilisateur.</p>

Présentation spécifique d'un formulaire	Définit une présentation spécifique pour personnaliser le formulaire d'un enregistrement dans les jeux de données associés.
Historisation	Spécifie si l'historisation est activée, et le niveau de garantie demandé. Les profils d'historisation peuvent être modifiés dans la section Administration > Historique et journaux .
Index	<p>Propriété permettant d'indexer certains champs de la table. L'indexation améliore le temps d'accès des requêtes portant sur ces champs. La combinaison des champs dans chaque index doit être unique.</p> <p>Nom de l'index: Nom unique pour cet index.</p> <p>Champs à indexer: Les champs à indexer, où chaque champ d'index est identifié par un chemin XPath absolu qui débute sous la table.</p>
Filtres spécifiques	Définit des filtres pour afficher uniquement les enregistrements correspondant à certains critères.
Actions	Indique les actions autorisées ou interdites dans le contexte d'un jeu de données. Toutes les actions sont autorisées par défaut, cependant des droits d'accès peuvent être définis dans les jeux de données associés pour restreindre ces actions.

Contraintes d'unicité

Indique les champs dont les valeurs doivent être uniques dans la table.

Concepts apparentés [Contrôles sur les éléments du modèle de données](#) [p 47]

CHAPITRE 9

Contrôles sur les éléments du modèle de données

Après la création d'un élément, complétez sa définition avec des contrôles supplémentaires.

Voir aussi [Propriétés des éléments du modèle de données](#) [p 41]

Ce chapitre contient les sections suivantes :

1. [Contrôles simples](#)
2. [Contrôles avancés](#)

9.1 Contrôles simples

Il est possible d'établir des contrôles simples sur le contenu d'un champ en définissant des contrôles statiques. Les contrôles disponibles dépendent du type du champ.

Longueur fixe	Nombre exact de caractères requis pour ce champ.
Longueur minimale	Nombre minimum de caractères requis pour ce champ.
Longueur maximale	Nombre maximum de caractères autorisés pour ce champ.
Pattern	Expression régulière à laquelle la valeur du champ doit être conforme. Il est interdit de définir un pattern à la fois sur un champ et sur son type de donnée.
Partie décimale	Nombre maximum de chiffres autorisés dans la partie décimale de la valeur d'un champ de type décimal.
Nombre maximum de chiffres	<p>Nombre maximum de chiffres composant la valeur d'un champ de type entier ou décimal.</p> <p>Pour les champs de type "Décimal", la longueur maximale par défaut est fixée à 15 chiffres. Cette valeur par défaut est définie pour éviter des problèmes de précision ou d'arrondi lors de la saisie de nombres dépassant les 15 chiffres. En effet, si une valeur dépassant les 15 chiffres était saisie dans un formulaire, la mauvaise valeur remplacerait l'originale lors de la soumission de ce formulaire.</p>
Énumération	Définit une liste de valeurs possibles pour ce champ. Si une énumération est définie à la fois sur ce champ et sur son type de données, alors une énumération représentant l'intersection entre ces deux énumérations sera utilisée dans les jeux de données associés au modèle de données.
Supérieur à [constante]	Définit la valeur minimale autorisée pour ce champ.
Inférieur à [constante]	Définit la valeur maximale autorisée pour ce champ.

9.2 Contrôles avancés

Il est possible d'établir des contrôles avancés sur le contenu d'un élément en définissant des contrôles dynamiques contextuels. Les contrôles disponibles pour un élément dépendent de sa nature (table, groupe, etc.) et de son type de données.

Contrainte de clé étrangère	
Table	Définit la table ciblée par la clé étrangère. Une clé étrangère référence une table dans le même jeu de données par défaut. Elle peut aussi référencer une table d'un autre jeu de données du même espace de données, ou un jeu de données d'un autre espace de données.
Mode	Emplacement de la table ciblée par la clé étrangère. "Défaut" : modèle courant. "Autre jeu de données" : jeu de données qui se trouve dans le même espace de données. "Autre espace de données" : jeu de données appartenant à un espace de données différent.
Table référencée	Expression XPath indiquant l'emplacement de la table. Par exemple, /racine/MaTable.
Jeu de données référencé	Obligatoire si la table est dans un autre jeu de données. Le nom unique du jeu de données contenant la table référencée.
Espace de données référencé	Obligatoire si la table est dans un autre espace de données. Nom unique de l'espace de données contenant la table référencée.
Libellé	Définit les champs composant un libellé par défaut et des libellés localisés pour présenter les enregistrements de la table cible. Attention : Les droits d'accès définis dans les jeux de données associés ne sont pas appliqués lors de l'affichage des libellés. Les champs habituellement cachés à cause de droits d'accès seront affichés dans les libellés des enregistrements.
Filtre	Définit un filtre de clé étrangère en utilisant une expression XPath.

Supérieur à [variable]	Définit un champ qui spécifie la valeur minimale autorisée pour ce champ.
Inférieur à [variable]	Définit un champ qui spécifie la valeur maximale autorisée pour ce champ.
Longueur fixe [dynamique]	Définit un champ qui spécifie le nombre exact de caractères requis pour ce champ.
Longueur minimale [dynamique]	Définit un champ qui spécifie le nombre minimum de caractères requis.
Longueur maximale [dynamique]	Définit un champ qui spécifie le nombre maximum de caractères autorisés.
Valeurs exclues	Définit une liste de valeurs non autorisées pour ce champ.
Plage exclue de valeurs	<p>Définit une plage de valeurs non autorisées pour ce champ.</p> <p>Valeur maximale exclue : La valeur maximale non autorisée pour ce champ.</p> <p>Valeur minimale exclue : La valeur minimale non autorisée pour ce champ.</p>
Énumération alimentée par un autre noeud	Spécifie un champ qui définit les valeurs autorisées pour cette énumération. Le champ spécifié doit être une liste ou doit définir une énumération.

Propriétés de validation

Chaque contrainte, excepté celles utilisant une classe Java, peut définir des messages de validation. Il est possible d'associer une sévérité à ces messages de validation et ils peuvent être localisés en utilisant les propriétés suivantes :

Validation	Spécifie le message de validation et la sévérité associée à la contrainte.
Sévérité	Spécifie la sévérité de la contrainte. Les valeurs possibles sont : "Erreur", "Avertissement", et "Information".
Gestion des erreurs	Spécifie le comportement de la contrainte en cas d'erreur de validation. Il est possible d'indiquer que la contrainte doit toujours être valide pour toute opération (mise à jour d'un jeu de données, création, mise à jour et suppression d'un enregistrement), ou lorsqu'un utilisateur soumet un formulaire. Dans ce cas, toutes les saisies ou opérations qui rendraient la contrainte invalide seront rejetées et les données seront non modifiées. Si cette propriété n'est pas spécifiée, alors la contrainte bloquera uniquement les erreurs lors de la soumission d'un formulaire, sauf dans le cas de modèles relationnels où les contraintes de clés étrangères bloquent par défaut toutes les erreurs. Cette propriété est uniquement disponible sur les contrôles statiques, valeurs exclues, plage exclue de valeurs et contraintes de clés étrangères. Le caractère bloquant pour toutes les opérations ne s'applique pas sur les filtres de clé étrangère. Une contrainte de clé étrangère n'est donc pas bloquante si un enregistrement référencé existe mais ne répond pas aux critères de filtrage. Il n'est pas possible de définir cette propriété sur les contraintes structurelles définies dans les modèles relationnels. Si cette propriété est définie sur les contraintes "Longueur fixe", "Longueur maximum", "Nombre total de chiffres" et "Partie décimale" dans des modèles relationnels, une erreur sera levée lors de la compilation du modèle. Le caractère bloquant d'une contrainte est ignoré lors de l'import d'archives. Toutes les contraintes bloquantes, sauf les contraintes dites structurelles, sont désactivées lors de l'import d'archives.
Message	Message à afficher lorsque la valeur de ce champ dans un jeu de données ne respecte pas cette contrainte. Ce message peut être localisé.

Concepts apparentés [Propriétés des éléments du modèle de données](#) [p 41]

CHAPITRE 10

Actions sur les modèles de données existants

Lorsque le modèle de données est créé, des actions sont disponibles dans le menu **"Actions" du modèle de données** [p 29] dans le panneau de navigation.

Ce chapitre contient les sections suivantes :

1. [Validation du modèle de données](#)
2. [Import et export de fichiers XML Schema Document \(XSD\)](#)
3. [Duplication d'un modèle de données](#)
4. [Suppression d'un modèle de données](#)

10.1 Validation du modèle de données

Il est possible de valider un modèle de données en sélectionnant **"Actions" du modèle de données** > **Valider** dans le panneau de navigation. Les éventuels messages issus de la validation du modèle de données sont présentés dans un rapport. Depuis le rapport de validation, cliquez sur le bouton **Revalider** pour mettre à jour ce rapport, ou cliquez sur le bouton **Réinitialiser le rapport de validation** pour supprimer tous les messages de validation actuellement associés au modèle de données afin de pouvoir relancer une validation complète.

10.2 Import et export de fichiers XML Schema Document (XSD)

EBX5 fournit des services intégrés pour importer et exporter des fichiers XML Schema Document (XSD). Les services d'import et d'export sont accessibles depuis le menu **"Actions" du modèle de données** [p 29] dans le panneau de navigation. Un import ou export est toujours effectué sur l'intégralité du modèle de données. Lors d'un import, la structure du modèle de données courant est entièrement remplacée par le contenu du document XML Schema importé. Lors d'un export, le modèle de données complet est exporté dans le document XML Schema cible.

Pour importer un fichier XSD, le fichier doit être valide et conforme aux règles de validation du référentiel EBX5. Si le document déclare des ressources situées dans un module, le module doit être déclaré aussi dans la configuration du modèle de données. Si le module n'a pas été déclaré, le fichier XSD ne pourra pas être importé. Voir [Propriétés du modèle de données](#) [p 34] pour plus d'informations sur la déclaration des modules.

Pour importer un modèle, sélectionnez *Importer XSD* dans le menu **Actions** situé dans le panneau de navigation du modèle de données.

Un document XML Schema (XSD) peut être importé à partir du système de fichier local. Pour cela, sélectionnez *Importer à partir d'un document local* :

- **Nom du document** : chemin du document XSD à importer dans le système de fichiers local.

Il est possible d'importer un document XML Schema (XSD) contenu dans le référentiel. Pour cela, sélectionnez *Importer à partir du référentiel de modèles* :

- **Modèle** : nom du modèle de données à importer.

Note

Les fichiers XSD à importer doivent être encodés en "UTF-8". Les fichiers XSD exportés sont toujours encodés en "UTF-8".

10.3 Duplication d'un modèle de données

Pour dupliquer un modèle de données, sélectionnez "Dupliquer" dans le menu **Actions** du modèle. Nommez le nouveau modèle de données. Ce nom doit être unique dans le référentiel.

10.4 Suppression d'un modèle de données

Pour supprimer un modèle de données, sélectionnez "Supprimer" dans le menu [Actions du modèle de données](#) [p 29]. Quand un modèle de données est supprimé, toutes les publications associées au modèle restent disponibles. Si un nouveau modèle de données est recréé avec le même nom que celui qui a été supprimé, le nouveau modèle de données sera réassocié avec toutes les publications existant dans le référentiel. Au moment de la publication, il sera possible de confirmer le remplacement d'une publication existante.

Note

Seul un administrateur peut supprimer les publications d'un modèle de données dans la section "Administration".

Voir [Publication des modèles de données](#) [p 55] pour plus d'informations sur le processus de publication.

CHAPITRE 11

Publication du modèle de données

Ce chapitre contient les sections suivantes :

1. [A propos des publications](#)
2. [Modes de publication](#)
3. [Mode de publication "Embarqué"](#)

11.1 A propos des publications

Chaque jeu de données dans le référentiel EBX5 basé sur un **modèle de données embarqué** est associé à une publication d'un modèle de données, et non directement au modèle de données défini dans le **Data Model Assistant**. Une publication est créée la première fois qu'un modèle de données est publié en utilisant le bouton **Publier** du panneau de navigation. Il est possible, à partir d'une publication, de créer des jeux de données dont la structure sera basée sur la structure du modèle de données publié.

Note

Le bouton **Publier** est uniquement affiché pour les utilisateurs qui ont le droit de publier le modèle de données. Voir [Permissions du modèle de données](#) [p 33] pour plus d'informations.

Les jeux de données étant basés sur des publications, les modifications faites sur le modèle de données n'impacteront pas les jeux de données existants. Les jeux de données seront impactés uniquement lors de la mise à jour d'une publication existante.

11.2 Modes de publication

Un modèle de données peut être publié en mode "Embarqué" ou "Dans un module". Le mode de publication "Embarqué" génère une publication gérée et persistée dans le référentiel EBX5 et possède des fonctionnalités spécifiques dédiées à la gestion de version. Le mode de publication "Dans un module" crée un fichier XML Schema Document (XSD) à l'intérieur d'un module qui n'est pas géré par le référentiel EBX5.

Selon la configuration du modèle de données, EBX5 détermine automatiquement le processus de publication à utiliser lorsque l'on clique sur le bouton **Publier** dans le panneau de navigation. Quand un modèle de données spécifie le mode de publication "Dans un module" ainsi qu'un fichier XSD à cibler, le processus de publication génère le fichier XSD dans le module défini dans la configuration.

11.3 Mode de publication "Embarqué"

La première fois qu'un modèle de données est publié en mode embarqué, une nouvelle publication avec le même nom que le modèle est automatiquement créée dans le référentiel. Si différentes publications ont été créées à partir du modèle, sélectionnez celle à mettre à jour.

Pendant le processus de publication, si le modèle de données a déjà été publié, il est possible de visualiser dans une interface de comparaison les différences structurelles introduites par la nouvelle publication.

Le processus de publication permet aussi de créer une image en lecture seule de l'état actuel du modèle de données. Cette image sera utile si un précédent état du modèle doit être restauré après plusieurs modifications et publications du modèle de données.

Note

Les images sont des archives statiques du modèle de données et ne doivent pas être confondues avec les *versions* du modèle de données, qui sont des éditions du modèle évoluant en parallèle. Voir [Gestion des versions du modèle de données embarqué](#) [p 57] pour plus d'informations sur les versions des modèles de données.

CHAPITRE 12

Gestion des versions de modèles de données

Ce chapitre contient les sections suivantes :

1. [A propos des versions](#)
2. [Accès aux versions](#)
3. [Actions sur les versions](#)
4. [Limitations connues sur la gestion de versions du modèle de données](#)

12.1 A propos des versions

Il est possible de créer des *versions* pour les modèles de données. Cela permet d'avoir différents états du modèle qui évoluent en parallèle. Les versions ne doivent pas être confondues avec les images des modèles de données, qui sont prises au moment de publication et sont sauvegardées uniquement pour consultation d'historique en lecture seule.

12.2 Accès aux versions

Pour voir les versions existantes de votre modèle de données, sélectionnez "Gérer les versions" dans le menu "[Actions](#)" **du modèle de données** [p 29].

Les versions existantes sont représentées dans une arborescence selon leurs relations parent-enfant. Chaque modèle de données a une version racine par défaut.

12.3 Actions sur les versions

Dans l'espace de travail, en utilisant le menu avec la flèche vers le bas ▼ à côté de chaque version, accédez aux actions suivantes :

Accéder à la version du modèle	Pour visualiser la version correspondante du modèle de données.
Créer une nouvelle version	Crée une nouvelle version basée sur le contenu de la version sélectionnée. La nouvelle version est créée comme enfant de la version sélectionnée, mais les contenus évoluent indépendamment.
Définir en tant que version par défaut	Définit la version par défaut sélectionnée lorsqu'un utilisateur accède au modèle de données.
Exporter une archive	Exporte le modèle de données sélectionné dans une archive contenant les données de la version, ses permissions et ses informations. L'archive est exportée vers le répertoire d'archives, accessible par les administrateurs du référentiel.
Importer une archive	Importe le contenu d'une archive dans la version sélectionnée. L'archive à importer doit contenir un modèle de données avec le même nom que le modèle associé à cette version.

Une version peut être supprimée en cliquant le bouton **X** situé à sa droite. Une version ne peut pas être supprimée si elle est liée à une publication ou si elle a des sous versions. La version racine du modèle de données ne peut pas être supprimée.

Deux versions du même modèle peuvent être comparées dans l'espace de travail en sélectionnant leur case à cocher, puis "**Actions**" **du modèle de données** > **Comparer les versions sélectionnées**. La vue de comparaison côte-à-côte affiche les différences structurelles entre les versions du modèle de données, avec la plus ancienne à gauche et la plus récente à droite.

12.4 Limitations connues sur la gestion de versions du modèle de données

- Il est impossible de fusionner deux versions d'un modèle de données.
- L'interface de comparaison n'affiche pas les mises à jour des champs, uniquement les ajouts et suppressions.
- Il n'est pas possible de gérer des versions de modèles de données publiés dans un module.
- Les ressources contenues dans un module et utilisées par un modèle de données embarqué ne sont pas versionnées lorsqu'une version est créée. En effet, seules les références des ressources sont sauvegardées, et les développeurs doivent s'assurer que les ressources référencées restent compatibles avec les différentes versions.

Espaces de données

CHAPITRE 13

Introduction aux espaces de données

Ce chapitre contient les sections suivantes :

1. [Présentation](#)
2. [Utilisation de l'interface utilisateur de la section Espace de données](#)

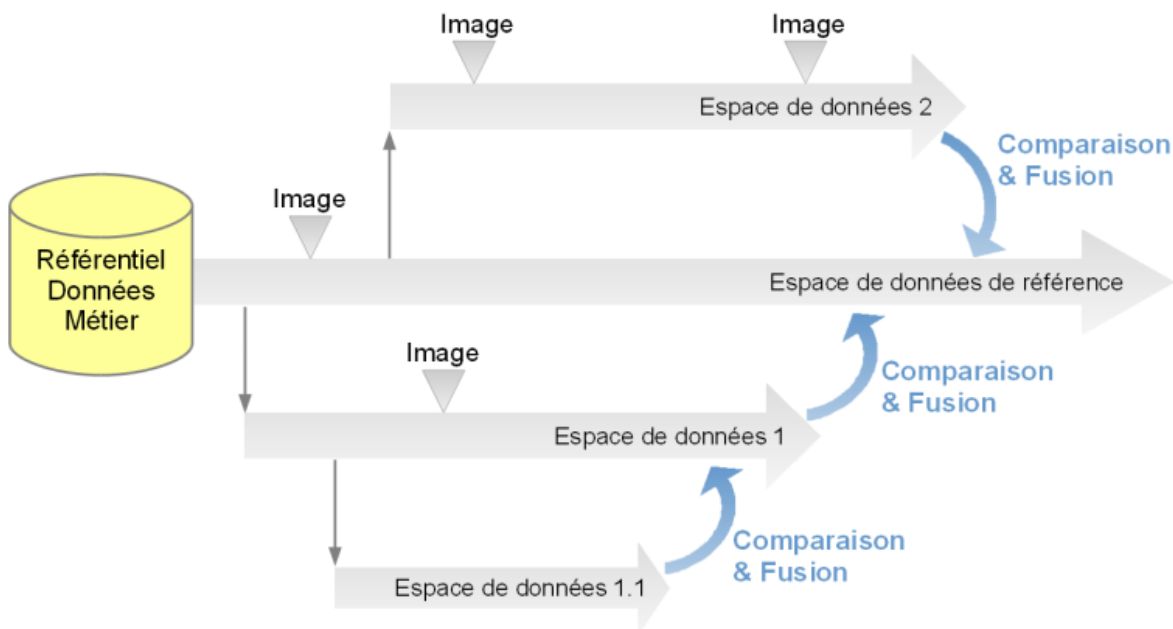
13.1 Présentation

Fonction d'un espace de données

Le cycle de vie des données est souvent complexe. Il est parfois nécessaire d'entretenir une version courante des données tout en travaillant sur des évolutions futures. De plus, il faut conserver une trace des états intermédiaires. EBX5 rend ces démarches possibles grâce aux espaces de données et aux images.

Un espace de données est un conteneur qui isole différentes versions de jeux de données et les organise. Des espaces de données enfants peuvent être créés à partir d'un espace de données. Un espace de données enfant est initialisé avec le même état que son parent au moment de sa création. Ainsi, des modifications peuvent être effectuées isolément dans l'espace de données enfant, sans impacter l'espace de données parent ni d'autres espaces de données. Lorsque les modifications dans l'espace de données enfant sont terminées, cet espace de données peut être fusionné avec son parent.

Une image est une copie statique d'un espace de données qui capture son état et tout son contenu à un moment donné. Les images peuvent être consultées, exportées, et comparées à d'autres espaces de données.



Concepts de base liés aux espaces de données

La compréhension des termes suivants est recommandée pour l'utilisation des espaces de données :

- [espace de données](#) [p 21]
- [image](#) [p 21]
- [jeu de données](#) [p 20]
- [fusion](#) [p 21]
- [espace de données de référence](#) [p 21]

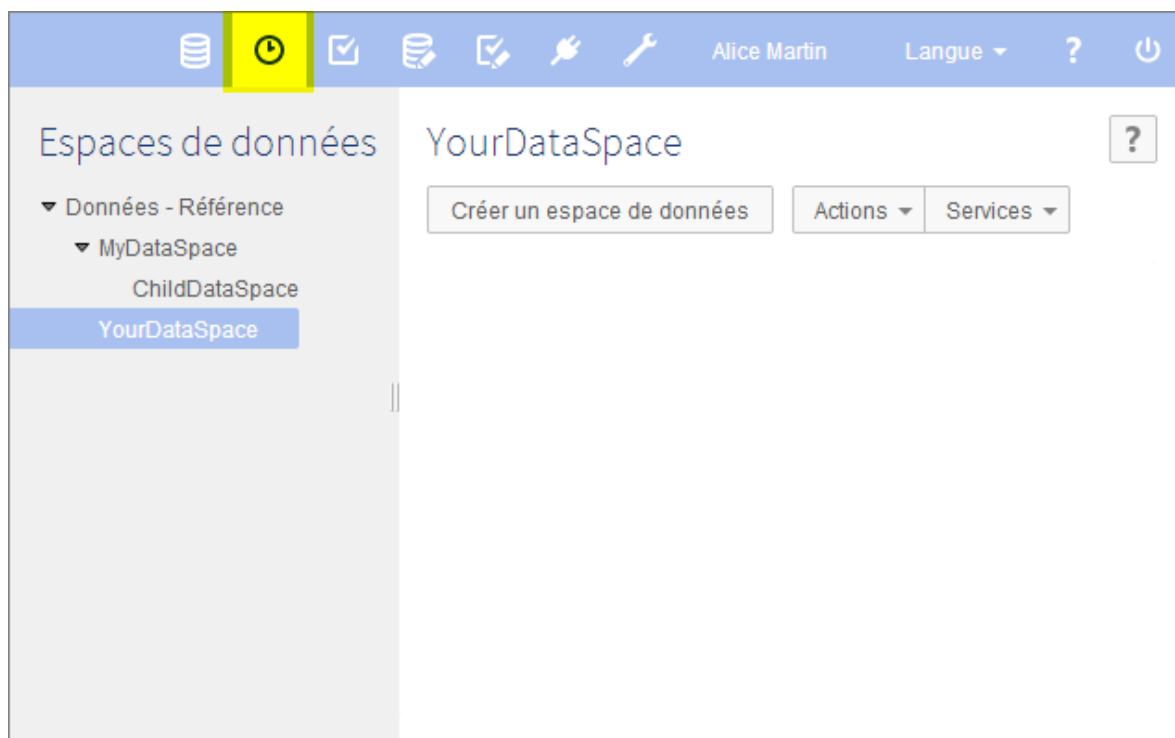
13.2 Utilisation de l'interface utilisateur de la section Espace de données

Les espaces de données sont créés, consultés et modifiés dans la section **Espaces de données**.

Note

Seuls les utilisateurs autorisés peuvent accéder à cet écran via la 'Perspective avancée'.

Le panneau de navigation affiche l'organisation hiérarchique des espaces de données existants, tandis que l'espace de travail affiche les informations concernant l'espace de données sélectionné et liste ses images.



Voir aussi

[Création d'un espace de données](#) [p 65]

[Images](#) [p 75]

Concepts apparentés [Jeux de données](#) [p 80]

CHAPITRE 14

Création d'un espace de données

Ce chapitre contient les sections suivantes :

1. [Création d'un espace de données](#)
2. [Propriétés](#)
3. [Mode relationnel](#)

14.1 Création d'un espace de données

Par défaut, les espaces de données dans EBX5 sont en *mode sémantique*. Ce mode supporte toutes les fonctionnalités permettant la gestion du cycle de vie des données.

Pour créer un nouvel espace de données en mode sémantique, sélectionnez un espace de données existant sur lequel il se basera, puis cliquez sur le bouton **Créer un espace de données** situé dans l'espace de travail.

Note

Seuls les utilisateurs autorisés peuvent accéder à cet écran via la 'Perspective avancée'.

L'espace de données nouvellement créé devient un espace de données enfant de celui qui était sélectionné. Son contenu est automatiquement initialisé avec le contenu de son parent au moment de la création. Une image initiale représentant cet état est créée.

Excepté l'espace de données de référence, qui est la racine de tous les espaces de données sémantiques du référentiel, les espaces de données sémantiques sont toujours l'enfant d'un autre espace de données.

Voir aussi [Mode relationnel](#) [p 66]

14.2 Propriétés

Les informations suivantes sont requises lors de la création d'un nouvel espace de données :

Identifiant	Identifiant unique de l'espace de données.
Propriétaire	Utilisateur possédant l'espace de données et qui est autorisé à en modifier les informations et les permissions. Le propriétaire d'un espace de données n'est pas obligatoirement son créateur.
Libellé	Libellé et description associés à l'espace de données en plusieurs langues.
Mode relationnel	Spécifie si l'espace de données est en mode relationnel. Cette option existe seulement lors de la création d'un espace de données à partir de l'espace de données de référence.

14.3 Mode relationnel

Les espaces de données en mode relationnel ne peuvent être créés qu'à partir de l'espace de données de référence. Ils sont limités en fonctionnalités par rapport aux espaces de données en mode sémantique. Par exemple, les espaces de données en mode relationnel n'ont pas d'images et ne peuvent pas avoir d'espaces de données enfants.

CHAPITRE 15

Actions sur les espaces de données existants

Ce chapitre contient les sections suivantes :

1. [Informations de l'espace de données](#)
2. [Permissions sur un espace de données](#)
3. [Fusion d'un espace de données](#)
4. [Comparaison d'un espace de données](#)
5. [Validation d'un espace de données](#)
6. [Archives d'espaces de données](#)
7. [Fermeture d'un espace de données](#)

15.1 Informations de l'espace de données

Certaines propriétés associées à un espace de données peuvent être modifiées en sélectionnant **Actions** > **Information** dans l'espace de travail de la section Espaces de Données.

Documentation	Libellé et description localisés associés à l'espace de données.
Politique de fusion	<p>La politique de fusion des espaces de données enfants ne s'applique qu'aux fusions initiées par un utilisateur. Elle ne s'applique pas aux fusions programmatiques, telles que celles initiées par les scripts de workflow.</p> <p>Les politiques possibles sont :</p> <ul style="list-style-type: none"> • Autoriser des erreurs de validation dans le résultat : c'est la politique par défaut. Un espace de données enfant peut être fusionné quelle que soit la validité du résultat de cette fusion. • Fusion pré-validante : un espace de données enfant ne peut être fusionné que si le résultat de la fusion est valide.
Propriétaire	L'utilisateur possédant l'espace de données et qui est autorisé à en modifier les informations et les permissions. Le propriétaire d'un espace de données n'est pas obligatoirement son créateur.
Tri des espaces enfants	Définit l'ordre d'affichage des espaces de données enfants dans l'arbre des espaces de données. Si la valeur n'est pas définie, l'ordre défini par le parent est pris en compte. La valeur par défaut est "par libellé".
Changement de propriétaire	Définit si le propriétaire de l'espace de données a le droit de modifier l'attribut "Propriétaire". Si la valeur est "Non habilité", seul l'administrateur a le droit d'effectuer cette modification.
Changement des permissions	Définit si le propriétaire de l'espace de données a le droit de modifier les permissions de l'espace de données. Si la valeur est "Non habilité", seul l'administrateur a le droit d'effectuer cette modification.

15.2 Permissions sur un espace de données

Permissions générales

Identifiant de l'espace de données	Indique l'espace de données auquel les permissions vont être appliquées.
Sélection du profil	Indique le profil concerné par la permission définie.
Restriction d'accès	Indique si la permission définie restreint les permissions affectées à un utilisateur donné par des politiques définies pour d'autres profils.
Permissions d'accès à l'espace de données	<p>Indique la permission globale d'accès à l'espace de données.</p> <p>Lecture seule</p> <ul style="list-style-type: none"> • Permet de visualiser l'espace de données et son image, et de voir les espaces de données enfants selon les droits s'appliquant à chacun d'eux. • Permet de visualiser le contenu de l'espace de données sans pouvoir le modifier, à condition que les droits s'appliquant au contenu en autorisent l'accès. <p>Ecriture</p> <ul style="list-style-type: none"> • Permet de visualiser l'espace de données et son image, et de voir les espaces de données enfants selon les droits s'appliquant à chacun d'eux. • Permet de modifier le contenu de l'espace de données, à condition que les droits s'appliquant au contenu en autorisent l'accès. <p>Non visible</p> <ul style="list-style-type: none"> • Ni l'espace de données, ni ses images ne peuvent être vus. • A partir d'un espace de données enfant, l'espace de données courant est visible sans toutefois pouvoir être sélectionné. • Le contenu de l'espace de données n'est pas accessible. • Aucune action ne peut être effectuée sur l'espace de données.

Actions possibles

Les utilisateurs peuvent être autorisés à effectuer les actions suivantes :

Créer un espace de données enfant	Indique si le profil peut créer un espace de données enfant.
Créer une image	Indique si le profil peut créer une image de l'espace de données.
Initier une fusion	Indique si le profil peut fusionner un espace de données avec son parent.
Exporter une archive	Indique si le profil peut exporter l'espace de données.
Importer une archive	Indique si le profil peut importer dans l'espace de données.
Fermer un espace de données	Indique si le profil peut fermer l'espace de données.
Fermer une image	Indique si le profil peut fermer les images de l'espace de données.
Droits sur les services	Spécifie les permissions d'accès aux services.
Permissions des espaces de données enfants à la création	Spécifie les permissions d'accès aux espaces de données enfants créés à partir de l'espace de données courant.

15.3 Fusion d'un espace de données

Quand le travail dans un espace de données est terminé, il est possible d'effectuer une fusion unidirectionnelle de l'espace de données vers son espace de données parent. Le processus de fusion est le suivant :

1. l'espace de données parent et l'espace de données enfant sont verrouillés pour tous les utilisateurs, excepté l'utilisateur qui a initié la fusion ainsi que les utilisateurs administrateurs. Les verrous sont maintenus pendant la durée de la fusion. Ainsi, les contenus des deux espaces de données peuvent être lus, mais ne peuvent pas être modifiés.

Note : en plus de s'appliquer aux modifications directes sur l'espace de données, cette restriction s'applique également aux autres fusions des autres espaces de données enfants. Ainsi, il est impossible de fusionner d'autres espaces de données enfants tant que la fusion en cours n'est pas terminée.

2. les changements qui ont été effectués dans l'espace de données depuis sa création sont intégrés dans l'espace de données parent ;
3. l'espace de données enfant est fermé ;
4. l'espace de données parent est déverrouillé.

Initiation d'une fusion

Suivez ces étapes pour initier la fusion d'un espace de données avec son espace de données parent :

1. sélectionnez l'espace de données à fusionner dans le panneau de navigation de la section Espaces de données ;
2. dans l'espace de travail, sélectionnez **Fusionner l'espace de données** dans le menu **Actions**.

Revue et acceptation des changements

Avant la fusion définitive, sélectionnez les changements survenus dans l'espace de données enfant (source) qui doivent être fusionnés dans l'espace de données parent (cible).

Pendant la fusion, un écran de comparaison récapitule tous les changements devant être revus. Deux colonnes de *listes de changements* sont affichées, prenant en compte les changements des comparaisons d'espaces de données suivantes :

- l'espace de données enfant par rapport à son image initiale ;
- l'espace de données parent par rapport à l'image initiale de l'espace de données enfant.

Par défaut, tous les changements détectés sont sélectionnés pour la fusion. Pour exclure un changement de la fusion, désélectionnez-le. Les changements relatifs aux différents éléments sont visibles en les sélectionnant dans le panneau de navigation.

Afin de détecter les conflits, la fusion implique l'espace de données courant, l'image initiale et l'espace de données parent. En effet, les données peuvent avoir été modifiées à la fois dans l'espace de données courant et dans son parent.

Le processus de fusion concerne aussi les droits d'accès aux tables. Il faut également passer en revue les modifications des permissions pour décider si elles seront incluses dans la fusion.

Lorsque vous avez choisi les changements à fusionner, vous devez cliquer sur le bouton **Marquer les différences comme revues** pour indiquer que vous avez vérifié les changements dans le périmètre actuel. Tous les changements doivent être revus pour effectuer la fusion.

Types de modifications

Le processus de fusion considère les opérations suivantes comme des modifications à évaluer :

- la création d'un enregistrement ou d'un jeu de données ;
- la modification de toute entité ;
- la suppression d'un enregistrement, d'un jeu de données, ou de la valeur d'un noeud ;
- la modification des permissions d'une table.

Types de conflits

Cette interface de revue affiche également les conflits détectés. Des conflits peuvent survenir quand une entité contient des modifications dans l'espace de données source et l'espace de données cible.

Les conflits sont catégorisés comme suit :

- conflit de création d'un enregistrement ou d'un jeu de données,
- conflit de modification de toute entité,
- conflit de suppression d'un enregistrement ou d'un jeu de données,
- tout autre conflit.

Finalisation d'une fusion

Lorsque tous les changements ont été vérifiés et que ceux à inclure dans le résultat de la fusion ont été choisis, cliquez sur le bouton **Fusionner** >> dans le panneau de navigation.

En fonction de la politique de fusion de l'espace de données parent, le processus de finalisation peut différer. Par défaut, une fusion peut être effectuée même si le résultat de la fusion contient des erreurs de validation. En revanche, l'administrateur de l'espace de données parent peut configurer la politique de fusion pour que les fusions de ses espaces de données enfants soient finalisées uniquement si le résultat ne contient pas d'erreur de validation.

Si la politique de fusion pré-validante est utilisée, un espace de données dédié est d'abord créé pour accueillir le contenu de la fusion. S'il est valide, cet espace de données est automatiquement fusionné avec l'espace de données parent.

Dans le cas contraire, si des erreurs de validation sont détectées dans l'espace de données dédié, seuls seront accessibles l'espace de données d'origine et l'espace de données dédié contenant le résultat de la fusion, nommé "[fusion] <nom de l'espace de données enfant>". Les options suivantes sont alors proposées dans le menu de l'espace de travail **Actions** > **Fusion en cours** :

- **Abandonner la fusion** : cette action abandonne la fusion en cours et restitue l'espace de données enfant dans son état précédant la fusion.
- **Poursuivre la fusion** : cette action permet de tenter de nouveau la fusion après avoir effectué les corrections nécessaires dans l'espace de données de fusion dédié.

Configuration de la politique de fusion d'un espace de données

En tant qu'administrateur d'un espace de données, il est possible, à travers l'interface utilisateur, d'interrompre la finalisation des fusions de ses espaces de données enfants si le résultat contient des erreurs de validation. Pour ce faire, cliquez sur **Actions** > **Informations** dans l'espace de travail de l'espace de données parent. Sur la page des informations de cet espace de données, positionnez la **Politique de fusion des enfants** à "Fusion pré-validante". Cette politique de fusion sera appliquée pour toutes les fusions des espaces de données enfants avec l'espace de données parent ainsi paramétré.

Note

Si la fusion est effectuée à travers un composant web, le comportement pour la politique de fusion est le même ; la politique définie par l'espace de données parent sera automatiquement utilisée lors de la fusion des modifications de l'espace de données enfant. Cependant, cette politique n'est pas appliquée pour les fusions programmatiques. Elle est donc ignorée pour les tâches automatiques des workflows de données.

Voir aussi [Politique de fusion](#) [p 72]

Abandon d'une fusion

Une fusion est effectuée dans le contexte d'une session utilisateur et doit être achevée en une seule opération. Si vous décidez de ne pas poursuivre la fusion après l'avoir initiée, cliquez sur le bouton **Annuler** afin d'abandonner l'opération.

Si vous naviguez vers une autre page pendant une fusion, celle-ci sera abandonnée mais les verrous sur l'espace de données parent et l'espace de données enfant seront maintenus. Il faudra les déverrouiller dans la section **Espaces de Données**.

Pour déverrouiller un espace de données, sélectionnez l'espace de données dans le panneau de navigation, et cliquez sur le bouton **Déverrouiller** dans l'espace de travail. Si vous effectuez le déverrouillage depuis l'espace de données enfant, les deux espaces de données seront déverrouillés. Si vous effectuez le déverrouillage depuis l'espace de données parent, lui seul sera déverrouillé, vous devrez donc aussi effectuer un déverrouillage de l'espace de données enfant.

15.4 Comparaison d'un espace de données

Il est possible de comparer le contenu d'un espace de données à celui d'un autre espace de données ou à une image dans le référentiel. Pour effectuer une comparaison, ouvrez l'espace de données dans le panneau de navigation, puis sélectionnez **Actions > Comparer** dans l'espace de travail. L'assistant permet de choisir un autre espace de données ou une image à comparer avec l'espace de données courant.

Pour une comparaison plus rapide, qui ignore les champs avec une valeur héritée ou calculée, sélectionnez le filtre "Valeurs persistantes seulement".

15.5 Validation d'un espace de données

Le contenu d'un espace de données peut être validé globalement en utilisant le service de validation au niveau de l'espace de données. Ce service est accessible en sélectionnant **Actions > Validation** dans l'espace de travail.

Note

Ce service est proposé uniquement si l'utilisateur a la permission de valider tous les jeux de données contenus dans l'espace de données.

15.6 Archives d'espaces de données

Export d'une archive

Le contenu d'un espace de données peut être exporté dans une archive en sélectionnant **Actions > Export** dans le panneau de navigation. L'archive est sauvegardée sur le système de fichiers du serveur, où seul un administrateur peut la récupérer.

Pour réaliser un export, les informations suivantes sont requises:

Nom du fichier archive	Nom de l'archive exportée.
Type d'export	<p>Obligatoire.</p> <p>Le type d'export par défaut est "Contenu complet de l'espace de données". Il permet d'exporter l'ensemble des données sélectionnées dans l'archive.</p> <p>Il peut être utile d'inclure uniquement les différences entre l'espace de données et son image initiale dans l'archive. Il existe deux types d'export qui incluent un delta : "Mises à jour avec leur contenu complet" et "Mises à jour seules". Le premier exporte l'état actuel de l'espace de données ainsi que le delta qui contient les différences entre l'état actuel de l'espace de données et l'image initiale. Le second exporte uniquement le delta qui contient les différences entre l'état actuel de l'espace de données et l'image initiale. Les deux options affichent une page de comparaison, sur laquelle il est possible de sélectionner les différences à inclure dans le delta. Les différences sont détectées au niveau table.</p>
Jeu de données à exporter	Jeux de données de cet espace de données à exporter. Pour chaque jeu de données, il est possible de spécifier si les données, les permissions et les informations doivent être exportées.

Importer une archive

Le contenu d'une archive peut être importé dans un espace de données en sélectionnant **Actions > Import**.

Si l'archive sélectionnée ne contient pas de delta, l'état actuel de l'espace de données sera remplacé par le contenu de l'archive.

Si l'archive sélectionnée se compose d'un contenu complet et d'un delta, vous pouvez choisir d'appliquer le delta afin de fusionner les différences incluses. Un écran de comparaison sera affiché, depuis lequel les différences à fusionner peuvent être sélectionnées.

Si l'archive sélectionnée contient uniquement un delta, il est possible de sélectionner les différences à fusionner dans un écran de comparaison.

15.7 Fermeture d'un espace de données

Si un espace de données n'est plus utilisé, il peut être fermé. Dès qu'il est fermé, l'espace de données n'apparaîtra plus dans la section **Espaces de Données** de l'interface utilisateur, et ne sera plus accessible.

Un administrateur peut rouvrir un espace de données fermé, tant que celui-ci n'a pas encore été purgé du référentiel.

Pour fermer un espace de données, sélectionnez **Actions > Fermer l'espace de données**.

CHAPITRE 16

Images

Ce chapitre contient les sections suivantes :

1. [Présentation des images](#)
2. [Création d'une image](#)
3. [Visualisation de contenu d'une image](#)
4. [Informations de l'image](#)
5. [Comparaison d'une image](#)
6. [Validation d'une image](#)
7. [Export d'une image](#)
8. [Fermeture d'une image](#)

16.1 Présentation des images

Une image est une copie en lecture seule d'un espace de données. Une image sert de référence de l'état et du contenu d'un espace de données à un instant donné.

Voir aussi [image](#) [p 21]

16.2 Création d'une image

Une image est créée en sélectionnant un espace de données dans le panneau de navigation, puis en cliquant sur le bouton 'Créer une image' sous le menu 'Actions'.

Les informations suivantes sont requises :

Identifiant	Identifiant unique de l'image. Le modèle suivant doit être respecté: [a-zA-Z0-9_:\-\\]*.
Libellé	Libellé et description localisés associés à l'image.

16.3 Visualisation de contenu d'une image

Pour visualiser le contenu d'une image, ouvrir l'image, puis sélectionner *Actions > Voir ou éditer les jeux de données* dans l'espace de travail.

16.4 Informations de l'image

Certaines propriétés associées à une image peuvent être modifiées en sélectionnant l'image, puis *Actions > Informations* dans l'espace de travail de la section *Espaces de données*.

Documentation	Libellé et description localisés associés à l'image.
Mode relationnel	Indique que les tables des jeux de données présents dans cet espace de données sont contenues dans une base de données relationnelle. Il ne sera pas possible de créer des images et des espaces de données enfants.
Propriétaire	L'utilisateur possédant l'image, et qui est autorisé à en modifier les informations et les permissions. Le propriétaire d'une image n'est pas obligatoirement son créateur.
Tri des espaces enfants	Définit l'ordre d'affichage des espaces de données enfants dans l'arbre des espaces de données. Si non défini, l'ordre défini par le parent est pris en compte. La valeur par défaut est 'par libellé'.
Changer le propriétaire	Définit si le propriétaire de l'image a le droit de modifier l'attribut "Propriétaire". Si la valeur est "Non habilité", seul l'administrateur a le droit d'effectuer cette modification.
Changer les permissions	Spécifie si un utilisateur qui est propriétaire de l'espace de données a le droit de modifier les permissions de cet espace de données. Si ce n'est pas le cas, seul un administrateur ou un 'super propriétaire' de l'espace de données a le droit de modifier les permissions.

16.5 Comparaison d'une image

Il est possible de comparer le contenu d'une image à celui d'une autre image ou d'un espace de données dans le référentiel. Pour effectuer une comparaison, ouvrir l'image, puis sélectionner *Actions > Comparer* dans l'espace de travail. L'assistant permet de choisir une autre image ou un espace de données à comparer avec l'image courante.

Pour une comparaison plus rapide, qui ignore les champs avec une valeur héritée ou calculée, sélectionner le filtre 'Valeurs persistantes seulement'.

16.6 Validation d'une image

Le contenu d'une image peut être validé globalement en utilisant le service de validation au niveau de l'image. Ce service est accessible en sélectionnant *Actions > Valider* dans l'espace de travail.

Note

Pour utiliser ce service, l'utilisateur doit avoir la permission de valider tous les jeux de données contenus dans l'image.

16.7 Export d'une image

Le contenu d'une image peut être exporté dans une archive en sélectionnant *Actions > Exporter* dans l'espace de travail. L'archive est sauvegardée sur le système de fichiers du serveur, où seul un administrateur peut la récupérer.

Pour réaliser un export, les informations suivantes sont requises :

Nom du fichier archive à créer	Le nom de l'archive exportée.
Type d'export	Le contenu complet de l'espace de données, Mises à jour avec leur contenu complet (*), Mises à jour seules (*). (*) : les mises à jour à exporter sont sélectionnées dans les pages suivantes.
Jeu de données (ou arbre de jeux de données) à exporter	Les jeux de données de cette image à exporter. Pour chaque jeu de données, il est possible de choisir si les données, les permissions et les informations doivent être exportées.

16.8 Fermeture d'une image

Si une image n'est plus utilisée, elle peut être fermée. Dès qu'elle est fermée, l'image n'apparaît plus dans la section 'Espaces de données' de l'interface utilisateur et n'est plus accessible.

Un administrateur peut rouvrir une image fermée, tant qu'elle n'a pas été purgée du référentiel.

Pour fermer une image, sélectionner *Actions > Fermer l'image*.

Jeux de données

CHAPITRE 17

Introduction aux jeux de données

Ce chapitre contient les sections suivantes :

1. [Présentation](#)
2. [Utilisation de l'interface utilisateur de la section Données](#)

17.1 Présentation

Fonction d'un jeu de données

Un jeu de données est un conteneur de données qui se base sur les définitions de structure fournies par le modèle de données qu'il implémente. Lors de la publication d'un modèle de données, il est possible de créer des jeux de données basés sur sa définition. Par la suite, si ce modèle est modifié et republié, tous ses jeux de données associés sont mis à jour automatiquement.

Dans un jeu de données, les valeurs de données sont consultables et modifiables. A l'aide des vues, il est possible d'afficher les tables d'une manière adaptée à la nature des données et du mode d'accès. Les recherches et les filtres peuvent aussi être utilisés pour restreindre l'affichage ou rechercher des données.

Des permissions peuvent aussi être affectées à différents rôles pour contrôler l'accès au niveau du jeu de données. Ainsi, en appliquant des permissions spécifiques, on peut permettre à certains utilisateurs d'afficher ou de modifier des données tout en les cachant à d'autres.

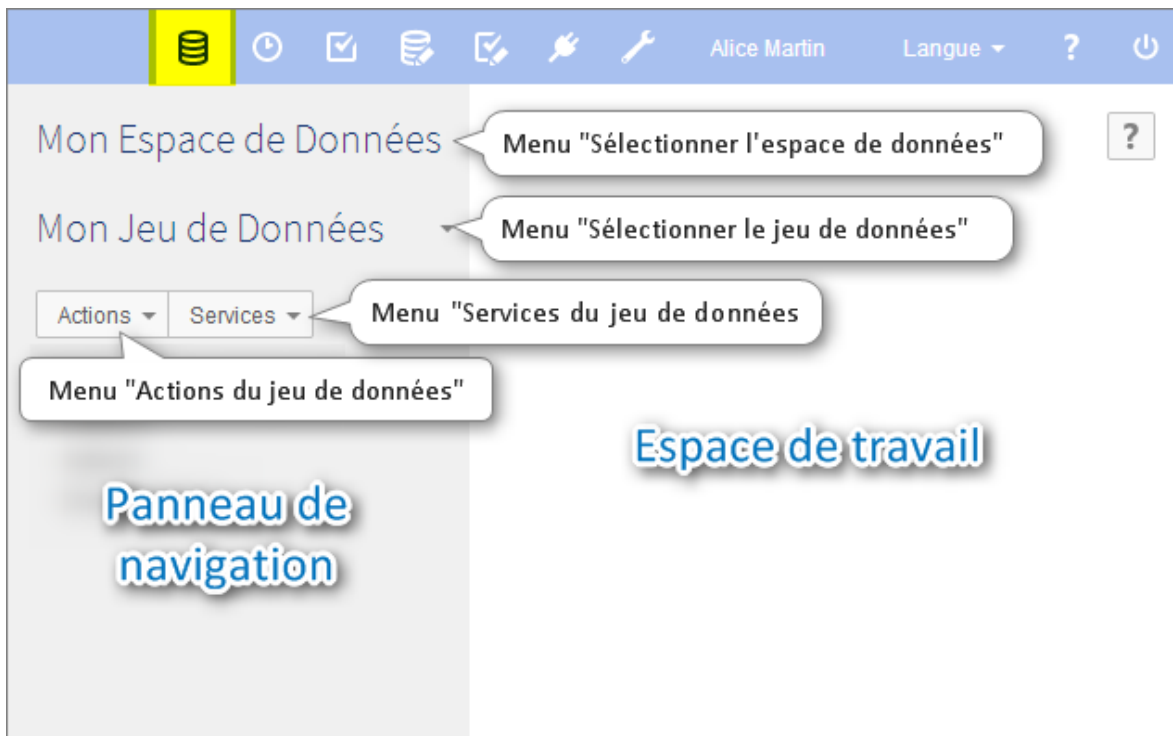
Concepts de base liés aux jeux de données

La compréhension des termes suivants est recommandée pour utiliser les jeux de données :


- [espace de données](#) [p 21]
- [jeu de données](#) [p 20]
- [enregistrement](#) [p 19]
- [champ](#) [p 18]
- [clé primaire](#) [p 18]
- [clé étrangère](#) [p 19]
- [table \(jeu de données\)](#) [p 20]
- [groupe](#) [p 19]

17.2 Utilisation de l'interface utilisateur de la section Données

Dans le cadre de l'utilisation de la [Perspective avancée](#) [p 11], ou d'une perspective spécifique, les jeux de données sont créés, consultés et modifiés dans la section 'Données'. Seuls les utilisateurs autorisés peuvent accéder à ces interfaces spécifiques.



Pour sélectionner ou créer un jeu de données, cliquer sur 'Sélectionner le jeu de données' dans le panneau de navigation. La structure de données du jeu de données s'affichera dans le panneau de navigation et les formulaires d'enregistrement ainsi que les vues de table s'afficheront dans l'espace de travail.

Lors de la visualisation d'une table du jeu de données dans l'espace de travail, le bouton  permet d'afficher les recherches et filtres disponibles pour restreindre l'affichage des enregistrements.

Les actions applicables au jeu de données sont disponibles dans les menus 'Actions' et 'Services' du panneau de navigation.

Voir aussi

[Création du jeu de données](#) [p 83]

[Recherche et filtrage des données](#) [p 86]

[Actions sur les enregistrements dans l'interface utilisateur](#) [p 93]

[Héritage](#) [p 20]

Concepts apparentés

[Modèle de données](#) [p 28]

[*Espace de données*](#) [p 62]

CHAPITRE 18

Création du jeu de données

Ce chapitre contient les sections suivantes :

1. [Création d'un jeu de données racine](#)
2. [Création d'un jeu de données enfant utilisant l'héritage](#)

18.1 Création d'un jeu de données racine

Afin de créer un jeu de données racine, qui n'hérite pas d'un jeu de données parent, il faut sélectionner le menu '[Sélectionner le jeu de données](#) [p 81]' dans le panneau de navigation, cliquer sur le bouton 'Créer un jeu de données' dans la fenêtre contextuelle, et suivre l'assistant.

Note

Seuls les utilisateurs autorisés peuvent accéder à cet écran via la 'Perspective avancée' ou via une perspective spécifique.

L'assistant vous permet de sélectionner un des trois types de modèles de données, sur lequel le nouveau jeu de données sera basé : packagé, embarqué ou externe.

- Un *modèle de données packagé* est un modèle de données défini à l'intérieur d'un module (application web).
- Un *modèle de données embarqué* est un modèle de données créé et publié en utilisant l'assistant de modèles de données. Il est entièrement géré à l'intérieur du référentiel EBX5. Un modèle de données doit avoir été préalablement publié pour que cette fonctionnalité soit disponible.
- Un *modèle de données externe* est un modèle de données référencé au moyen d'une URI.


Après avoir choisi le modèle de données sur lequel le nouveau jeu de données sera basé, vous devez fournir un nom unique, sans espaces ni caractères spéciaux. Facultativement, vous pouvez donner des libellés localisés pour le jeu de données, qui seront affichés aux utilisateurs dans l'interface en fonction de leurs préférences de langue.

Attention

Le contenu des tables n'est pas copié lors de la duplication d'un jeu de données.

18.2 Création d'un jeu de données enfant utilisant l'héritage

Le mécanisme d'héritage permet d'utiliser les relations parent-enfant, grâce auxquelles les jeux de données enfants héritent par défaut des valeurs de leurs jeux de données ancêtres. Pour pouvoir créer des jeux de données enfants depuis un jeu de données parent, il faut activer l'héritage entre jeux de données dans le modèle de données associé.

Pour créer un jeu de données enfant, cliquer sur [Créer ou sélectionner un jeu de données](#) [p 81] dans le panneau de navigation, puis sur le bouton  correspondant au jeu de données parent.

Le jeu de données enfant sera basé sur le même modèle de données que celui de son parent, et donc les seules informations à spécifier sont son nom unique, et éventuellement les libellés localisés.

Voir aussi [Héritage entre jeux de données](#) [p 99]

CHAPITRE 19

Visualisation des données

Ce chapitre contient les sections suivantes :

1. [Menu 'Vue'](#)
2. [Tri des données](#)
3. [Recherche et filtrage des données](#)
4. [Vues](#)
5. [Gestion des vues](#)

19.1 Menu 'Vue'

Le menu déroulant 'Vue' permet d'accéder facilement à toutes les fonctionnalités majeures de visualisation.

Les vues sont gérées dans le sous-menu dédié '[Gestion de vues](#)' [p 91].

Les vues peuvent également être regroupées. L'administrateur doit avoir défini des groupes au préalable dans la table 'Groupes de vues' de la section 'Configuration des vues'. L'utilisateur peut ainsi définir une vue comme appartenant à un groupe dans le champ 'Groupe de la vue' lors de la création ou de la modification d'une vue. Voir [Description d'une vue](#) [p 88] pour plus d'informations.

19.2 Tri des données

Les critères de tri contrôlent l'ordre dans lequel les enregistrements sont présentés.

Par défaut, les enregistrements sont ordonnés par clé primaire ascendante.


Afin de définir des critères de tri spécifiques, sélectionnez *Vue > Critères de tri* dans l'espace de travail.

Chaque critère de tri est défini par un nom de colonne et un ordre de tri (ascendant ou descendant). Utiliser les flèches 'Déplacer à gauche/droite' pour ajouter ou enlever un critère dans la table 'Trié'. Une fois un critère sélectionné, il est possible de choisir son ordre de tri en cliquant sur le bouton 'ASC' ou 'DESC' à droite.

Pour changer la priorité d'un critère de tri, sélectionnez-le dans la liste, puis cliquez sur les flèches vers le haut ou vers le bas pour le déplacer.

Pour rétablir la vue sans critère de tri spécifique, sélectionner *Vue > Rétablir*.

19.3 Recherche et filtrage des données

Des outils spécifiques permettent la recherche d'enregistrements dans une table. On peut y accéder par l'icône , qui affiche les panneaux des filtres.

Chaque panneau de filtre (ou de recherche) propose une case à cocher sur sa barre de titre : cocher cette case applique le filtre ; la décocher le désapplique.

Note

Appliquer une vue réinitialisera et retirera tous les filtres actuellement appliqués.

Recherche

En mode simple, la 'Recherche' permet d'ajouter des critères contextualisés sur un ou plusieurs champs. Lors de l'ajout d'un critère, les opérateurs proposés seront en adéquation avec le type du champ correspondant.

En activant le mode avancé, il est possible de créer des sous-blocs contenant des critères, afin de créer des opérations logiques plus complexes dans l'élaboration du filtre.

Note

En mode avancé, les critères dont l'opérateur est "correspond à" ou "correspond à (respecter la casse)" suivent la syntaxe standard des expressions régulières de Java.

Voir aussi [Regex pattern](#)

Recherche textuelle

La recherche textuelle est utilisée pour une recherche de texte brut sur un ou plusieurs champs. Cette recherche ne prend pas en compte le type du champ.

- Si le texte entré contient un ou plusieurs mots sans caractère de remplacement (* ou ?), les champs trouvés seront ceux contenant tous ces mots. Les mots entre guillemets ("aa bb" par exemple) sont considérés comme un seul mot.
- Les caractères usuels de remplacement sont proposés : l'étoile * (tout texte) ou le point d'interrogation ? (tout caractère). Pour des raisons de performance, leur utilisation est restreinte à un seul par recherche.
- Les caractères de remplacement peuvent être considérés comme de simple caractères en les échappant à l'aide du caractère '\', par exemple, '*'.

Exemples :

- aa bb : le champ contient "aa" et "bb".
- aa "bb cc" : le champ contient "aa" et "bb cc".
- aa* : le champ commence par "aa".
- *bb : le champ se termine par "bb".
- aa*bb : le champ commence par "aa" et se termine par "bb".
- aa? : le champ commence par "aa" et a une taille de 3 caractères.
- ?bb : le champ se termine par "bb" et a une taille de 3 caractères.
- aa?bb : le champ commence par "aa" et se termine par "bb" et a une taille de 5 caractères.
- aa*bb : le champ contient "aa*bb" tel quel.

Sur les tables de grande taille, il est recommandé de ne sélectionner qu'un seul champ de la table ; si le champ n'est pas du type "chaîne de caractères", il est conseillé d'entrer un texte au bon format, par exemple :

- booléen : Oui, Non
- date : 01/01/2000
- nombre : 100000 ou 100 000
- champ énuméré : Rouge, Bleu...

L'option *Sensible à la casse* oblige la recherche à tenir compte de la casse, en distinguant les majuscules des minuscules.

Recherche sur validation

La recherche par validation permet de voir les enregistrements en fonction de leur statut dans la dernière validation effectuée. Il est possible de voir les enregistrements avec les niveaux 'Erreur', 'Avertissements' et 'Informations'.

Note

Cette recherche s'applique seulement aux enregistrements de tables qui ont déjà été validés ; pour ce faire, sélectionner *Actions > Valider* au niveau de la table dans l'espace de travail, ou au niveau du jeu de données dans le panneau de navigation.

Recherches spécifiques sur tables

Pour chaque table, le modèle peut spécifier des filtres additionnels pour la recherche.

19.4 Vues

Une vue est créée en sélectionnant *Vue > Créer une nouvelle vue* dans l'espace de travail. Pour appliquer une vue, la sélectionner dans *Vue > nom de la vue*.

Deux modes avancés de visualisation sont disponibles à la création d'une nouvelle vue :

- 'Vue tabulaire simple' : une vue tabulaire qui permet de trier et filtrer les enregistrements affichés ;
- 'Vue hiérarchique' : une arborescence qui lie les données de différentes tables en utilisant leurs relations.

Description d'une vue

Lors de la création ou de la modification d'une vue, la première page permet de définir les informations générales concernant la vue.

Documentation	Libellé et description localisés associés à la vue.
Propriétaire	Nom du propriétaire de la vue, qui peut à ce titre la gérer et la modifier.
Autres profils autorisés	Détail des profils ayant le droit d'utiliser la vue.
Mode de vue	Vue tabulaire simple ou vue hiérarchique.
Groupe de la vue	Groupe d'appartenance de cette vue (le cas échéant).

Vue tabulaire simple

Les vues tabulaires simples offrent la possibilité de définir des critères pour filtrer les enregistrements et de sélectionner les colonnes à afficher.

Colonnes affichées	Spécifie, à l'aide de flèches, les colonnes de la table à afficher dans la vue.
Colonnes triées	Spécifie l'ordre d'affichage des colonnes et indique si les enregistrements de chaque colonne sont triés par ordre croissant ou décroissant. Voir Tri des données [p 85].
Filtre	Définit les critères utilisés pour filtrer les enregistrements.

Vues hiérarchiques

Une hiérarchie est une arborescence permettant de présenter les relations existant entre les tables. Elle peut être structurée sur plusieurs niveaux, appelés niveaux de dimension. En outre, il est possible de définir des filtres sur des niveaux afin de filtrer les enregistrements.

Dimension d'une hiérarchie

Une dimension définit une dépendance dans la hiérarchie. Par exemple, une dimension pourrait être précisée pour afficher des produits par catégorie. Plusieurs dimensions peuvent être définies pour une vue hiérarchique.

Options de configuration d'une vue hiérarchique

Ce formulaire permet de configurer les options d'une vue hiérarchique.

Afficher les enregistrements dans une nouvelle fenêtre	Si "oui", une nouvelle fenêtre sera ouverte pour afficher l'enregistrement. Sinon, il sera affiché dans une nouvelle page de la fenêtre courante.
Hiérarchie élaguée	Si "oui", les noeuds de la hiérarchie qui n'ont pas d'enfants et qui n'appartiennent pas à la table cible ne seront pas affichés.
Afficher les orphelins	Si "oui", les noeuds de la hiérarchie qui n'ont pas de parent seront affichés.
Afficher le noeud racine	Si 'Non', le noeud racine de la hiérarchie sera caché de la vue.
Libellé du noeud racine	Libellé localisé du noeud racine de la hiérarchie.

Libellés

Pour chaque niveau de dimension faisant référence à une autre table, il est possible de définir les libellés localisés pour les noeuds correspondants dans la hiérarchie. Utiliser l'assistant pour sélectionner les champs utilisés dans les définitions des libellés.

Filtre

L'éditeur de critères permet de définir un filtre d'enregistrement pour la vue.

Champ d'ordonnement

Afin de pouvoir déplacer les noeuds dans la vue hiérarchique, il faut désigner un champ d'ordonnement éligible. Celui-ci est défini dans la table sur laquelle est appliquée la vue hiérarchique. Un champ d'ordonnement doit avoir le type de données 'Entier' et doit avoir la vue par défaut 'Caché' dans les propriétés avancées du modèle de données.


Des actions de positionnement sur chaque noeud sont alors possibles, à moins que le champ d'ordonnement ne soit en lecture seule ou qu'un filtre ne soit défini sur la hiérarchie.

En l'absence d'un noeud d'ordonnement, les noeuds enfants sont triés selon l'ordre alphabétique des libellés des noeuds.

Attention

Le champ d'ordonnement doit être un champ technique dédié, et non un champ contenant des données métier. En effet, ces données seront modifiées automatiquement par les actions de réordonnement.

Actions sur un noeud de hiérarchie

Chaque noeud d'une vue hiérarchique possède un menu correspondant  qui offre des actions contextuelles.

Les noeuds terminaux peuvent être détachés de leur parent en sélectionnant l'option 'Détacher du parent'. L'enregistrement devient ainsi un noeud orphelin dans l'arborescence, rangé sous un conteneur portant le nom 'non défini'.

Les noeuds terminaux peuvent aussi changer de noeud parent, grâce à l'option 'Attacher à un autre parent'. Si, selon le modèle de données, un noeud peut avoir plusieurs parents, le noeud sera à la fois sous le parent d'origine et rajouté également sous le nouveau parent. Sinon, le noeud terminal sera déplacé sous le nouveau noeud parent.

Publication de vue

Une vue peut être publiée afin de la rendre accessible à tous les autres utilisateurs via les composants web, les tâches utilisateurs du workflow, et les services de données. Pour publier une vue, sélectionner 'Publier' dans le menu *Vue > Gérer les vues > nom de la vue*. Dès que la vue sera publiée, les autres utilisateurs pourront appliquer cette vue à leur interface en la sélectionnant dans *Vue > nom de la vue* dans leur espace de travail.

Cette publication de vue sera accessible aux utilisateurs qui appartiennent aux profils autorisés dans la définition de la vue.

19.5 Gestion des vues

Gérer les vues recommandées

Quand un utilisateur se connecte sans spécifier de vue, la vue recommandée, si elle existe, s'applique. Sinon, la vue par défaut s'affiche. L'action 'Gérer les vues recommandées' permet de définir les règles d'attribution des vues recommandées par utilisateur et par rôle.

Les actions disponibles sur les vues recommandées sont les suivantes : changer l'ordre d'attribution des règles, ajouter une règle, éditer une règle, supprimer une règle existante.

Pour un utilisateur donné, la règle appliquée sera la première de la liste qui correspond au profil de l'utilisateur.

Note

La fonctionnalité 'Gérer les vues recommandées' est uniquement accessible au propriétaire du jeu de données.

Gérer les vues

Le sous-menu 'Gérer les vues' permet d'accéder aux actions suivantes :

Définir cette vue comme ma favorite	Uniquement disponible lorsque la vue en cours d'affichage n'est PAS la vue recommandée. La vue favorite sera automatiquement appliquée lors de l'accès à la table.
Définir la vue recommandée comme ma favorite	Uniquement disponible lorsqu'une vue favorite est définie. En cliquant sur cette action, la vue courante ne sera plus la vue favorite de l'utilisateur. Une vue recommandée, à l'instar de la vue favorite, est appliquée automatiquement lors de l'accès à la table. Cette action n'est pas affichée si aucune vue favorite n'a été définie.

CHAPITRE 20

Edition des données

Ce chapitre contient les sections suivantes :

1. [Actions sur les enregistrements dans l'interface utilisateur](#)
2. [Import et export de données](#)


20.1 Actions sur les enregistrements dans l'interface utilisateur

L'édition des enregistrements s'effectue dans l'espace de travail de l'interface utilisateur.

Note

Seuls les utilisateurs autorisés peuvent accéder à cet écran via la 'Perspective avancée' ou via une perspective spécifique.

Création d'un enregistrement

Dans la vue tabulaire, un nouvel enregistrement peut être créé à l'aide du bouton  situé en haut à gauche de la table.

Dans une vue hiérarchique, sélectionnez "Créer un enregistrement" dans le menu du noeud parent du nouvel enregistrement.

Dans les deux cas, un formulaire s'affiche, permettant d'entrer des données. Les données obligatoires sont repérées par une astérisque rouge.

Modification d'un enregistrement

Un enregistrement peut être édité par double clic. Le formulaire qui s'affiche permet d'éditer l'enregistrement, tandis que le bouton *Rétablir* permet de recharger le formulaire sans soumettre aucun des changements effectués.

Dupliquer un enregistrement

Pour dupliquer un enregistrement, sélectionnez-le, puis sélectionnez *Actions > Dupliquer*.

Un formulaire apparaît, pré-rempli à partir des valeurs de l'enregistrement copié. La clé primaire doit ensuite être modifiée pour pouvoir créer ce nouvel enregistrement, à moins qu'elle ne soit générée automatiquement (à l'exemple d'une valeur auto-incrémentée).

Supprimer

Pour supprimer un ou plusieurs enregistrements sélectionnés, sélectionnez *Actions > Supprimer*.

Comparer

Deux enregistrements sélectionnés peuvent être comparés en sélectionnant *Actions > Comparer*.

Note

La comparaison ne prend pas en compte le contenu des noeuds terminaux complexes, comme les listes agrégées ou les attributs utilisateurs. Toutes les différences portant sur de tels noeuds seront ignorées.

20.2 Import et export de données

Dans une table, les enregistrements peuvent être importés ou exportés, depuis ou vers les formats CSV ou XML.

Vous pouvez soit exporter l'ensemble de la table, soit sélectionner manuellement certains enregistrements à exporter, grâce aux cases à cocher.

Voir aussi

[Services CSV](#) [p 215]

[Services XML](#) [p 209]

CHAPITRE 21

Actions sur les jeux de données existants

Ce chapitre contient les sections suivantes :


1. [Validation du jeu de données](#)
2. [Duplication d'un jeu de données](#)
3. [Désactivation d'un jeu de données](#)
4. [Gestion des permissions de jeux de données](#)

21.1 Validation du jeu de données

Il est possible de valider un jeu de données en sélectionnant *Actions > Valider* depuis le panneau de navigation. Les éventuels messages issus de la validation du jeu de données sont présentés dans un rapport. Depuis le rapport de validation, cliquer sur le bouton *Revalider* pour mettre à jour ce rapport. Pour supprimer tous les messages de validation actuellement associés au jeu de données, et pouvoir relancer une validation complète, cliquer sur le bouton *Réinitialiser le rapport de validation*.

Dans la section 'Données', il est également possible de valider une table, en la sélectionnant dans le panneau de navigation et en utilisant l'action *Actions > Valider* dans l'espace de travail.

21.2 Duplication d'un jeu de données

Pour dupliquer un jeu de données existant, sélectionnez-le dans le menu "[Sélectionner le jeu de données](#) [p 81]"  dans le panneau de navigation, puis sélectionnez *Actions > Dupliquer*.

21.3 Désactivation d'un jeu de données

Si un jeu de données est activé, il sera sujet à la validation. Tous les éléments obligatoires doivent être définis pour que le jeu de données soit valide. Si un jeu de données est activé et valide, on considère qu'il peut être exporté de façon sécurisée vers des systèmes externes (ou utilisé par d'autres applications Java).

Il est possible de désactiver un jeu de données dont les éléments obligatoires ne sont pas définis, en spécifiant "Non" pour la propriété "Activé" dans *Actions > Informations*.

21.4 Gestion des permissions de jeux de données

Les permissions de jeux de données sont accessibles en sélectionnant *Actions* > *Permissions* dans le panneau de navigation.

Les permissions sont définies en créant des *profils*. Pour créer un nouveau profil de permissions, créez un nouvel enregistrement dans la table "Droits d'accès par profil".

Voir aussi [Profil](#) [p 17]

Profil	Indique le profil concerné par la permission définie.
Restriction d'accès	Indique si la permission définie restreint celles affectées à un utilisateur donné par des politiques définies pour d'autres profils.
Actions sur les jeux de données	Cette section spécifie les permissions des actions sur les jeux de données.
Créer un le jeu de données enfant	Indique si le profil peut créer un jeu de données enfants. L'héritage doit aussi être activé dans le modèle de données.
Dupliquer le jeu de données	Indique si le profil peut dupliquer le jeu de données.
Supprimer le jeu de données	Indique si le profil peut supprimer le jeu de données.
Activer/désactiver le jeu de données	Indique si le profil peut modifier la propriété "Activé" dans les informations du jeu de données. Voir Désactivation d'un jeu de données [p 95].
Créer une vue	Indique si le profil peut créer des vues et des hiérarchies.
Droits sur tables	Spécifie les permissions par défaut pour toutes les tables. Des permissions spécifiques peuvent être appliquées à une ou plusieurs tables en cliquant sur le bouton '+' sous Droits spécifiques par table.
Droits par défaut >Créer un nouvel enregistrement	Indique si le profil peut créer des enregistrements dans une table.
Droits par défaut >Surcharger des enregistrements	Indique si le profil peut remplacer des enregistrements hérités dans une table. Cette permission est utile quand on utilise l'héritage de jeu de données.
Droits par défaut >Occulter des enregistrements	Indique si le profil peut occulter des enregistrements hérités dans une table. Cette permission est utile quand on utilise l'héritage de jeu de données.
Droits par défaut >Supprimer un enregistrement	Indique si le profil peut supprimer des enregistrements dans une table.

Droits sur valeurs	<p>Spécifie les permissions d'accès par défaut pour tous les éléments (tables, groupes et champs) d'un jeu de données, et permet de définir des permissions pour des éléments spécifiques. Les permissions d'accès par défaut sont utilisées, à condition qu'il n'y ait pas de permission spécifique affectée à un élément.</p> <p>Le sélecteur de droits spécifiques permet d'attribuer des permissions d'accès spécifiques à un élément. Les liens "Lecture", "Ecriture" et "Non visible" déterminent les permissions d'accès correspondant à l'élément sélectionné.</p> <p>Il est possible de retirer une permission d'accès spécifique en utilisant le lien "(par défaut)".</p>
Droits sur les services	<p>Spécifie les permissions d'accès sur les services. Un service barré n'est pas accessible à un profil.</p>

CHAPITRE 22

Héritage entre jeux de données

En utilisant le concept d'héritage entre jeux de données, vous pouvez créer des jeux de données additionnels, à partir d'un jeu de données racine. Ces jeux de données enfants héritent des propriétés et des valeurs de leur parents, qui peuvent être surchargées si nécessaire. Plusieurs niveaux d'héritage peuvent être créés.

L'héritage peut être utilisé pour adapter des données de référence à divers contextes. Par exemple, il serait possible de définir des valeurs globales dans un jeu de données parent, et de créer des jeux de données enfants par zones géographiques. Ceci permettra à ces derniers d'hériter des valeurs de leur parent, et de les surcharger si besoin.

Note

Le comportement standard est d'interdire l'héritage de jeux de données. Il est donc nécessaire d'activer explicitement cette fonction au niveau du modèle de données.

Voir aussi [Configuration du modèle de données](#) [p 34]

Ce chapitre contient les sections suivantes :

1. [Structure de l'héritage entre jeux de données](#)
2. [Héritage de valeurs](#)

22.1 Structure de l'héritage entre jeux de données

Une fois le jeu de données racine créé, un jeu de données enfant peut être créé à l'aide du bouton , situé dans l'écran de sélection des jeux de données du panneau de navigation.

Note

- Un jeu de données ne peut pas être supprimé s'il a des jeux de données enfants. Ces enfants doivent être supprimés préalablement.
- Si un jeu de données enfant est dupliqué, le jeu de données nouvellement créé sera inséré dans l'arbre des jeux de données existants, au même niveau de l'arbre que le jeu de données dupliqué.

22.2 Héritage de valeurs

Quand un jeu de données enfant est créé, il hérite de toutes les valeurs des champs et des enregistrements de tables de son parent. Un champ ou un enregistrement peut soit hériter ses valeurs, soit les surcharger.


Dans une vue tabulaire, les valeurs héritées sont signalées par un repère dans le coin en haut à gauche de la cellule.

Le bouton  permet de surcharger une valeur.

Héritage d'enregistrement

Une table dans un jeu de données enfant hérite des enregistrements des tables de ses jeux de données ancêtres. La table dans le jeu de données enfant peut rajouter, éditer ou supprimer des enregistrements. Des états sont définis pour différencier les types d'enregistrement.

Racine	Un enregistrement racine est un enregistrement créé dans le jeu de données courant, qui n'existe pas dans les jeux de données ancêtres. Il sera hérité par les jeux de données enfants.
Hérité	Un enregistrement hérité est défini dans un des jeux de données ancêtres du jeu de données courant.
Surchargé	Un enregistrement surchargé est un enregistrement hérité dont les valeurs sont éditées dans le jeu de données courant. Les valeurs surchargées seront héritées par les jeux de données enfants.
Occulté	Un enregistrement occulté est un enregistrement hérité qui est supprimé du jeu de données courant. Il apparaîtra toujours dans le jeu de données courant comme un enregistrement barré, mais il ne sera pas hérité par les jeux de données enfants.

Quand le bouton  est activé, la valeur de l'enregistrement est héritée du jeu de données parent. Ce bouton peut être désactivé, afin de surcharger l'enregistrement ou la valeur. Pour un enregistrement occulté, l'activation de ce bouton restaure l'état hérité.

La table suivante résume le comportement des enregistrements lorsque l'on crée, modifie, ou supprime un enregistrement, selon son état initial.

Etat	Création	Edition	Suppression
Racine	Création normale d'un enregistrement. L'enregistrement nouvellement créé sera hérité par ses jeux de données enfant.	Edition normale d'un enregistrement. Les nouvelles valeurs seront héritées par les jeux de données enfants.	Suppression normale d'un enregistrement. L'enregistrement va disparaître du jeu de données courant ainsi que des jeux de données enfants.
Hérité	Si un enregistrement est créé à l'aide de la clé primaire d'un enregistrement hérité existant, l'état de l'enregistrement devient surchargé, et sa valeur sera celle soumise à sa création.	Un enregistrement hérité doit être déclaré comme surchargé pour que ses valeurs soient modifiables.	Supprimer un enregistrement hérité change son état à "occulté".
Surchargé	Non applicable. Il est impossible de créer un nouvel enregistrement si la clé primaire est déjà utilisée.	Un enregistrement surchargé peut être remis à l'état "hérité", mais sa valeur spécifique sera perdue. Les valeurs de l'enregistrement surchargé peuvent être héritées ou modifiées.	Supprimer un enregistrement surchargé change son état à "occulté".
Occulté	Si un enregistrement est créé en utilisant la clé primaire de l'enregistrement existant occulté, l'état de l'enregistrement devient "surchargé" et sa valeur sera celle soumise à la création.	Non applicable. Un enregistrement occulté ne peut plus être édité.	Non applicable. Un enregistrement occulté est déjà considéré comme supprimé, et ne peut donc pas être supprimé une deuxième fois.

Modèles de workflow

CHAPITRE 23

Introduction aux modèles de workflow

Ce chapitre contient les sections suivantes :

1. [Présentation](#)
2. [Utilisation de l'interface de modélisation de workflow](#)
3. [Modèles de message génériques](#)
4. [Limitations de workflows](#)

23.1 Présentation

Définition d'un modèle de workflow

Dans EBX5, les workflows facilitent la gestion collaborative des données dans le référentiel. Un workflow peut contenir des actions utilisateurs sur les données ainsi que des tâches automatiques, tout en émettant des notifications sur différents événements.

La première étape pour réaliser un workflow est de créer un *modèle de workflow* qui définit la succession d'étapes, les implications des utilisateurs, ainsi que le comportement du workflow.

Une fois qu'un modèle de workflow est défini, il peut être validé et publié comme *publication de workflow*. Ensuite, les workflows de données peuvent être lancés à partir de la publication de workflow pour exécuter les étapes définies dans le modèle de workflow.

Voir aussi

[Modèle de workflow \(glossaire\)](#) [p 22]

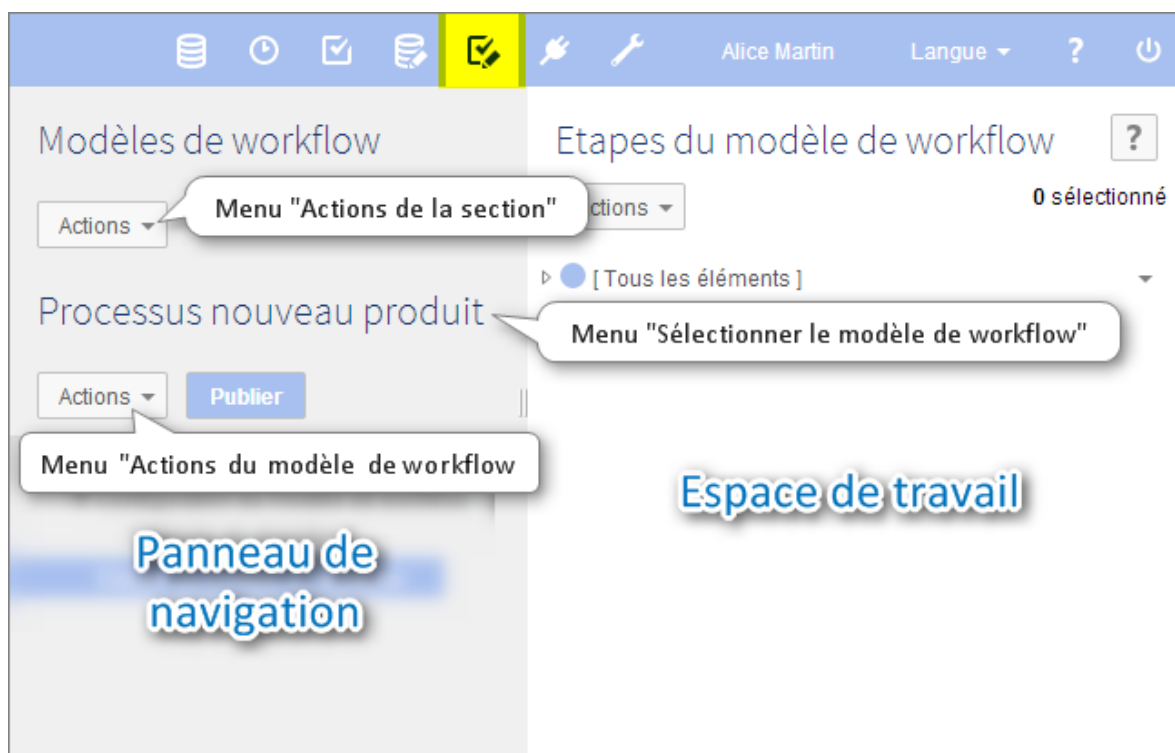
[Workflow de données \(glossaire\)](#) [p 24]

Concepts de base utilisés dans la modélisation des workflows

Une compréhension des termes suivants est nécessaire pour réaliser la création de modèles de workflows :

- [tâche automatique](#) [p 23]
- [tâche utilisateur](#) [p 23]
- [bon de travail](#) [p 24]
- [condition de workflow](#) [p 23]
- [appel à des sous-workflows](#) [p 23]
- [tâche d'attente](#) [p 23]
- [contexte des données](#) [p 23]

23.2 Utilisation de l'interface de modélisation de workflow



Note

Seuls les utilisateurs autorisés peuvent accéder à cet écran via la 'Perspective avancée'.
Seuls les utilisateurs autorisés peuvent accéder à ces interfaces spécifiques.

23.3 Modèles de message génériques

Des emails de notification peuvent être envoyés pour notifier les utilisateurs d'événements spécifiques pendant l'exécution d'un workflow.

Les modèles de message peuvent être définis et réutilisés dans n'importe quel modèle de workflow dans le référentiel. Pour modifier les modèles de message génériques, sélectionnez "Modèles de message" dans le menu 'Actions' de la section Modèles de Workflow.

Ces modèles, qui sont partagés par tous les modèles de workflows, sont figés et inclus dans chaque publication de workflow. Ainsi, pour prendre en compte les modifications des modèles de message, il sera nécessaire de mettre à jour les publications existantes en re-publiant les modèles de workflow concernés.

A la création d'un modèle de message générique, deux champs sont obligatoires :

- 'Libellé & Description' : spécifie les libellés et descriptions associés à ce modèle de message, localisé.
- 'Message' : spécifie l'objet de l'email et son corps de texte, localisés.

Le 'Type du message' est une donnée facultative.

Le message peut inclure des variables du contexte de données sous la forme `${nom.variable}`, qui seront évaluées lorsque le message sera envoyé. De plus, les variables système suivantes peuvent être incluses :

system.time	Heure système du référentiel.
system.date	Date système du référentiel.
workflow.lastComment	Dernier commentaire sur la tâche utilisateur précédente.
workflow.lastDecision	Dernières décisions sur la tâche utilisateur précédente.
user.fullName	Nom complet de l'utilisateur notifié.
user.login	Login de l'utilisateur notifié.
workflow.process.label	Libellé du workflow en cours.
workflow.process.description	Description du workflow en cours.
workflow.workItem.label	Libellé du bon de travail en cours.
workflow.workItem.description	Description du bon de travail en cours.
workflow.workItem.offeredTo	Rôle auquel le bon de travail courant a été proposé.
workflow.workItem.allocatedTo	Utilisateur, à qui le bon de travail en cours a été alloué.
workflow.workItem.link	Lien d'accès au bon de travail courant dans la corbeille, au moyen de l'API du composant web.
workflow.workItem.link.allocateAndStart	Lien d'accès au bon de travail courant dans la corbeille, au moyen de l'API du composant web. Si le bon de travail cible n'est pas encore démarré, il sera automatiquement alloué à l'utilisateur qui a cliqué sur le lien, puis démarré.
workflow.currentStep.label	Libellé de l'étape courante.

workflow.currentStep.description
Description de l'étape courante.

Exemple

Modèles de message générique :

Aujourd'hui à \${system.time}, un nouveau bon de travail vous a été proposé.

Email résultant :

Aujourd'hui à 15:19, un nouveau bon de travail vous a été proposé.

23.4 Limitations de workflows

Les fonctionnalités suivantes ne sont pas supportées :

- **Tâches programmées**, tâches exécutées dès lors que leur tour vient, et dont l'exécution ne peut pas être reportée.
- **Tâches événementielles** permettant au workflow de progresser quand il reçoit un événement, du type appel web service.
- **Limitation temporelle** sur la durée d'une tâche.

Concepts apparentés [Workflows de données](#) [p 128]

CHAPITRE 24

Modélisation du workflow

Ce chapitre contient les sections suivantes :

1. [Création d'un modèle de workflow](#)
2. [Implémentation des étapes](#)
3. [Tâche utilisateur](#)
4. [Tâche autonome](#)
5. [Conditions](#)
6. [Appels à des sous-workflows](#)
7. [Tâches d'attente](#)

24.1 Création d'un modèle de workflow

Un modèle de workflow peut être créé à partir de la section 'Modèle de workflow'. La seule information requise à la création est un nom de modèle unique dans le référentiel.

Les étapes du modèle de workflow sont initialisées avec une transition initiale. Pour implémenter le modèle de workflow, il faut définir la séquence des étapes qui suivent cette transition initiale.

24.2 Implémentation des étapes

Un modèle de workflow définit des étapes correspondant à des opérations et des conditions. Les étapes possibles sont les suivantes :

- Tâche utilisateur
- Tâche automatique
- Condition
- Appel à des sous-workflows
- Tâche d'attente

Un contexte de données est lié à chaque workflow de données. Ce contexte de données peut être utilisé pour définir des variables, qui pourront être utilisées en entrée et/ou en sortie dans les différentes étapes du workflow.

Stratégie d'avancement de l'étape suivante

Pour chaque type d'étape (excepté pour les appels de sous-workflows), une propriété est disponible pour préciser la stratégie d'avancement pour l'étape suivante. A la terminaison de l'étape, cette stratégie est évaluée pour conditionner la navigation lors de l'exécution du workflow. Par défaut, la stratégie d'avancement est 'Afficher la table des bons de travail'. Dans ce cas, après l'exécution de l'étape, la table des bons de travail est automatiquement affichée pour sélectionner le prochain bon de travail à ouvrir.

Une autre stratégie est disponible : 'Ouvrir automatiquement l'étape suivante'. Cette stratégie permet à l'utilisateur de garder la main sur ce workflow et d'exécuter directement l'étape suivante. Si, à la suite de cette exécution, un bon de travail est atteint et que l'utilisateur connecté peut le démarrer, le bon de travail est automatiquement ouvert (si plusieurs bons de travail sont atteints, le premier créé est ouvert). Sinon, la stratégie d'avancement de l'étape suivante est évaluée. Si aucun bon de travail n'est finalement atteint, la table des bons de travail est affichée.

Cette stratégie est préconisée pour exécuter plusieurs étapes à la suite sans repasser par la corbeille de tâches.

Il existe actuellement plusieurs limitations qui font que cette stratégie peut être ignorée. Dans ce cas, la table des bons de travail est automatiquement affichée. Les cas où cette propriété est ignorée sont les suivants : si l'étape suivante est un sous-workflow, ou si l'étape courante est une tâche utilisateur avec plus d'un bon de travail.

Dans le cas des conditions, deux autres stratégies sont proposées : 'Si vrai, ouvrir automatiquement l'étape suivante' et 'Si faux, ouvrir automatiquement l'étape suivante'. Ces stratégies permettent de conditionner la stratégie à appliquer en fonction du résultat de la condition.

24.3 Tâche utilisateur

Une tâche utilisateur est une étape qui nécessite une action de la part d'un utilisateur humain. Son libellé et sa description peuvent être localisés.

Participants

Les participants sont les rôles ou les utilisateurs auxquels la tâche utilisateur est destinée. Si un profil est un utilisateur, le bon de travail est automatiquement alloué à cet utilisateur. Si un profil est un rôle, le bon de travail est proposé aux membres du rôle.

Service

EBX5 propose les services prédéfinis suivants :

- Accéder à des données
- Accéder à l'interface de fusion des espace de données
- Accéder à un espace de données
- Comparer deux contenus
- Créer un nouvel enregistrement
- Dupliquer un enregistrement
- Exporter les données depuis une table au format CSV
- Exporter les données depuis une table au format XML
- Fusionner un espace de données
- Importer les données dans une table depuis un fichier CSV
- Importer les données dans une table depuis un fichier XML
- Valider un espace de données, une image ou un jeu de données

Voir aussi [Services prédéfinis de EBX5](#) [p 155]

Configuration

Options générales > Rejet activé

Par défaut, seulement l'action *accepter* est proposée à l'utilisateur lorsqu'il enregistre sa décision.

Il est possible d'activer l'action 'rejeter' en positionnant ce champ à 'Oui'.

Options générales > Demande de confirmation activée

Par défaut, quand un utilisateur enregistre sa décision en cliquant sur le bouton 'Accepter' ou 'Rejeter', une demande de confirmation est affichée.

Il est possible de désactiver cette demande de confirmation de la décision en positionnant ce champ à 'Non'.

Options générales > Caractère obligatoire du commentaire

Par défaut, le commentaire associé à un bon de travail est optionnel.

Il est possible de forcer l'utilisateur à saisir un commentaire avant d'enregistrer sa décision en positionnant ce champ sur le comportement attendu : 'toujours obligatoire', 'obligatoire seulement si le bon de travail a été accepté' ou 'obligatoire seulement si le bon de travail a été rejeté'.

Options générales > Libellés personnalisés

Durant l'exécution des tâches utilisateur, l'utilisateur peut accepter ou rejeter son bon de travail en cliquant sur le bouton correspondant. Dans la modélisation de workflow, il est possible pour certaines tâches utilisateur de définir un libellé et un message de confirmation personnalisés pour ces boutons. Cette fonctionnalité est particulièrement utile pour ajouter une signification particulière à l'action d'accepter ou rejeter un bon de travail.

Terminaison > Critère de fin de tâche

Une tâche utilisateur peut être assignée à plusieurs *participants* et générer plusieurs bons de travail durant l'exécution du workflow. Lors de la définition d'une tâche utilisateur dans le modèle de workflow, il est possible de sélectionner un des critères prédéfinis qui déterminent quand une tâche utilisateur est terminée, en se basant sur le statut des bons de travail associés. Lorsque la condition de sortie de la tâche utilisateur sera remplie, le workflow de données avancera jusqu'à l'étape suivante définie dans le modèle.

Par exemple, pour le cas d'une tâche utilisateur qui demande la validation de l'enregistrement d'un produit, il est possible de désigner trois participants. Le critère de fin de tâche permet de préciser si l'enregistrement du produit doit être validé par les trois participants, ou uniquement par le premier utilisateur à répondre.

Le critère de fin de tâche par défaut est 'Quand tous les bons de travail ont été acceptés'.

Terminaison > Tolérance de rejet

Par défaut, si un utilisateur rejette un bon de travail durant l'exécution d'un workflow, la tâche utilisateur est positionnée en erreur et l'avancement du workflow est stoppé.

Pour changer ce comportement par défaut, il est possible de définir un nombre de bons de travail rejetés à tolérer. Tant que la limite de rejets tolérés n'est pas dépassée, aucune erreur ne se produit et c'est le critère de fin de tâche qui détermine quand la tâche utilisateur est terminée.

Les critères de fin de tâche suivants tolèrent automatiquement tous les rejets :

- 'Quand tous les bons de travail ont été acceptés ou rejetés'
- 'Quand tous les bons de travail ont été acceptés, ou dès qu'un bon de travail a été rejeté'

Notification

Une notification par courrier électronique peut être envoyée aux utilisateurs quand des événements spécifiques se produisent. Pour chaque événement, vous pouvez spécifier un message type à utiliser. En outre, il est possible de définir un profil, auquel une copie de chaque courrier envoyé sera transmise.

Voir aussi [Modèles de message génériques](#) [p 105]

Relance

Des courriers électroniques de relance pour les bons de travail proposés ou alloués et inachevés peuvent être envoyés aux utilisateurs concernés de manière périodique.

Le contenu des courriers de relance dépend de l'état courant du bon de travail. En effet, si le bon de travail est proposé, la notification utilisera le modèle de courrier électronique 'Bons de travail proposés' ; si le bon de travail est alloué, la notification utilisera le modèle 'Bons de travail alloués'.

Echéance

Une tâche utilisateur peut avoir une échéance. Quand cette date est atteinte et si les bons de travail associés n'ont pas été terminés, un courrier électronique spécifique est envoyé aux utilisateurs concernés. Ce courrier électronique va alors être renvoyé tous les jours jusqu'à l'achèvement de la tâche.

Il y a deux types d'échéances :

- *Echéance absolue* : une date du calendrier.
- *Echéance relative* : durée (en heures, jours ou mois). La durée est évaluée à partir d'une date de référence : début d'une tâche utilisateur ou début d'un workflow.

24.4 Tâche autonome

Il existe deux types de tâches automatiques :

Script de la bibliothèque	EBX5 fournit des scripts de la bibliothèque prédéfinis, qui peuvent être utilisés directement.
Script spécifique	Spécifie une classe Java qui exécute des actions spécifiques. La classe associée doit être dans le même module que celui associé au modèle de workflow. Ses libellés et ses descriptions ne sont pas affichés dynamiquement aux utilisateurs dans un modèle de workflow.

Script de la bibliothèque

EBX5 inclut les scripts de la bibliothèque prédéfinis suivants :

- Créer un espace de données
- Créer une image
- Envoyer un courrier électronique
- Fermer un espace de données
- Fusionner un espace de données
- Importer une archive

Note

Certains scripts de bibliothèque prédéfinis sont marqués comme "obsolètes" car ils ne sont pas compatibles avec internationalisation. Il est recommandé d'utiliser les nouvelles tâches qui sont compatibles avec internationalisation.

24.5 Conditions

Il existe deux types de conditions :

Condition de la bibliothèque	EBX5 fournit des conditions de la bibliothèque prédéfinies, qui peuvent être utilisées directement.
Condition spécifique	Spécifie une classe Java qui implémente une condition spécifique. La classe associée doit être dans le même module que celui associé au modèle de workflow. Ses libellés et ses descriptions ne sont pas affichés dynamiquement aux utilisateurs dans un modèle de workflow.

Condition de la bibliothèque

EBX5 inclut les conditions de la bibliothèque prédéfinies suivantes :

- Dernière tâche utilisateur acceptée ?
- Espace de données modifié ?
- Espace de données valide ?
- Valeur nulle ou vide ?
- Valeurs égales ?

24.6 Appels à des sous-workflows

Les étapes du type appel à des sous-workflows positionnent le workflow de données courant dans l'état "en attente" et lancent un ou plusieurs sous-workflows.

Il est possible d'inclure un autre modèle de workflow dans le modèle de workflow courant en définissant un appel unique à ce sous-workflow dans une étape.

Si plusieurs sous-workflows sont appelés par une étape, ils sont exécutés simultanément, en parallèle. Tous les sous-workflows doivent être terminés pour que le workflow parent passe à l'étape suivante. Le libellé et la description de chaque sous-workflow peuvent être localisés.

Statique	<p>Définit un ou plusieurs sous-workflows à lancer chaque fois que cette étape est exécutée dans un workflow de données. Pour chaque sous-workflow, il est possible de spécifier ses libellés et ses descriptions, ainsi que les mappings d'entrée et de sortie dans son contexte de données.</p> <p>Ce mode est utile quand on sait à l'avance quels sous-workflows doivent être lancés, et comment le mapping de sortie doit être réalisé.</p>
Dynamique	<p>Spécifie une classe Java qui implémente un appel spécifique à des sous-workflows. Tous les workflows qui peuvent éventuellement être appelés comme sous-workflows par le code doivent être déclarés comme dépendances.</p> <p>Le contexte de données est directement accessible depuis le Java bean.</p> <p>Les appels de sous-workflows dynamiques doivent être déclarés dans un fichier <code>module.xml</code>.</p> <p>Ce mode est utile quand le lancement des sous-workflows est conditionnel (par exemple, lorsqu'il dépend d'une variable du contexte de données), ou quand le mapping de sortie dépend de l'exécution des différents sous-workflows.</p>

24.7 Tâches d'attente

Une étape d'attente dans un modèle de workflow met le workflow en pause en attendant la réception d'un événement donné.

Lorsqu'une tâche d'attente est appelée, le moteur de workflow génère un identifiant unique de réveil lié à la tâche d'attente. Cet identifiant est requis pour réveiller la tâche d'attente et, par conséquent, le workflow associé.

Note

L'administrateur built-in est toujours autorisé à réveiller un workflow.

CHAPITRE 25

Configuration du modèle de workflow

Ce chapitre contient les sections suivantes :

1. [Informations](#)
2. [Propriétés du modèle de workflow](#)
3. [Permissions sur les workflows de données associés](#)
4. [Historique des images du modèle de workflow](#)
5. [Suppression d'un modèle de workflow](#)

25.1 Informations

Pour visualiser et éditer les informations concernant le propriétaire et la documentation du modèle de workflow, sélectionnez "Informations" dans le menu ["Actions" du modèle de workflow](#) [p 105] dans le panneau de navigation.

Propriétaire	Personne pouvant éditer les informations du modèle de workflow et définir des règles de permission.
Documentation locale	Libellé et description associés au modèle de workflow localisés.
Activation	<i>Cette propriété est obsolète.</i> Spécifie si un modèle de workflow est activé. Un modèle de workflow doit être activé pour pouvoir être publié.

25.2 Propriétés du modèle de workflow

La configuration d'un modèle de workflow est accessible depuis le panneau de navigation.

Notification de démarrage	<p>Liste des profils qui doivent recevoir une notification (choisie dans la liste des modèles de messages) quand un workflow démarre.</p> <p>Voir Modèles de message génériques [p 105].</p>
Notification de fin	<p>Liste des profils qui doivent recevoir une notification (choisie dans la liste des modèles de messages), quand un workflow se termine. La notification n'est envoyée que si le workflow a été terminé "normalement" (pas par une action d'administration).</p> <p>Voir Modèles de message génériques [p 105].</p>
Priorité	<p>Par défaut, chaque workflow associé à ce modèle sera lancé avec cette priorité. Cette valeur est facultative. Si aucune valeur n'est définie ici, et une priorité par défaut est définie pour le référentiel, la priorité par défaut pour le référentiel sera appliquée à tous les workflows et bons de travail sans priorité.</p> <p>Voir Priorité de bons de travail [p 137] pour plus d'informations.</p> <p>Note : Seuls les utilisateurs définis en tant qu'administrateurs de workflows seront autorisés à modifier la priorité des workflows de données associés manuellement.</p>
Activer le lancement direct	<p>Par défaut, lorsqu'un workflow est lancé, il est proposé à l'utilisateur de saisir une documentation pour le nouveau workflow dans un formulaire intermédiaire. Cette documentation est facultative. Positionner la propriété "Activer le lancement direct" à "Oui" permet d'éviter cette étape de documentation et de lancer directement le workflow.</p>
Ouvrir automatiquement la première étape	<p>Permet de conditionner la navigation, suite à un lancement de workflow. Par défaut, une fois le workflow lancé, la table courante (lanceurs de workflow ou monitoring > publications) est automatiquement affichée.</p> <p>Activer cette propriété permet au créateur du workflow de garder la main sur le workflow lancé. Si, suite à l'exécution de la première étape du workflow, un bon de travail est atteint, et que ce bon de travail peut être démarré par le lanceur de workflow, le bon de travail est automatiquement</p>

ouvert (si plusieurs bons de travail sont atteints, le premier créé est ouvert). Cela évite au lanceur de sélectionner le bon de travail correspondant dans la corbeille de tâches.

Si aucun bon de travail n'est atteint, la stratégie d'avancement de l'étape suivante est évaluée.

Si aucun bon de travail n'est finalement ouvert, la table à partir de laquelle le workflow a été lancé est affichée.

Limitation : Cette propriété est ignorée si la première étape est un appel de sous-workflow.

Permissions	Définit les permissions pour les actions liées au workflows de données associés au modèle de workflow.
Permissions programmatiques	Définit le composant gérant les permissions du workflow. S'il est défini, il remplace l'ensemble des permissions définies dans la propriété 'Permissions'.

25.3 Permissions sur les workflows de données associés

Administration de workflow	Définit le profil autorisé à exécuter des tâches d'administration sur les workflows. Les actions d'administration sont : rejouer une étape, réveiller un workflow, terminer un workflow, désactiver une publication et dépublier. Pour pouvoir exécuter ces actions, ce profil bénéficie automatiquement de la permission "Visualiser les workflows". L'administrateur built-in est toujours autorisé à administrer les workflows.
Administration de workflow > Rejouer une étape	Définit le profil autorisé à rejouer une étape de workflow. Pour pouvoir exécuter cette action, ce profil bénéficie automatiquement de la permission "Visualiser les workflows". Pour rejouer une étape, un bouton est disponible dans la section "Monitoring > Workflows actifs". Un profil qui a la permission "Administration de workflow" est automatiquement autorisé à effectuer cette action spécifique. L'administrateur built-in est toujours autorisé à rejouer une étape de workflow.
Administration de workflow > Terminer un workflow	Définit le profil autorisé à terminer et supprimer un workflow. Pour pouvoir exécuter cette action, ce profil bénéficie automatiquement de la permission "Visualiser les workflows". Pour terminer et supprimer un workflow en cours, un bouton est disponible dans la section "Monitoring > Workflows actifs". Pour supprimer un workflow terminé, un bouton est disponible dans la section "Workflows terminés". Un profil qui a la permission "Administration de workflow" est automatiquement autorisé à effectuer cette action spécifique. L'administrateur built-in est toujours autorisé à terminer un workflow.
Administration de workflow > Forcer le réveil d'un workflow	Définit le profil autorisé à forcer le réveil d'un workflow en attente. Pour pouvoir exécuter cette action, ce profil bénéficie automatiquement de la permission "Visualiser les workflows". Pour réveiller un workflow, un bouton est disponible dans la section "Monitoring > Workflows actifs". Un profil qui a la permission "Administration de workflow" est automatiquement autorisé à effectuer cette action spécifique. L'administrateur built-in est toujours autorisé à réveiller un workflow.
Administration de workflow > Désactiver une publication	Définit le profil autorisé à désactiver une publication de workflow. Pour pouvoir exécuter cette action, ce profil bénéficie automatiquement de la permission "Visualiser les workflows". Pour désactiver, un bouton est disponible

dans la section "Monitoring > Publications" uniquement pour les publications actives. Un profil qui a la permission "Administration de workflow" est automatiquement autorisé à effectuer cette action spécifique. L'administrateur built-in est toujours autorisé à désactiver une publication.

Administration de workflow > Dépublier

Définit le profil autorisé à dépublier une publication de workflow. Pour pouvoir exécuter cette action, ce profil bénéficie automatiquement de la permission "Visualiser les workflows". Pour dépublier, un bouton est disponible dans la section "Monitoring > Publications" pour les publications désactivées uniquement. Un profil qui a la permission "Administration de workflow" est automatiquement autorisé à effectuer cette action spécifique. L'administrateur built-in est toujours autorisé à dépublier.

Gestion de l'allocation

Définit le profil autorisé à gérer l'allocation des bons de travail. Les actions d'allocation sont : allouer les bons de travail, réallouer les bons de travail et désallouer les bons de travail. Pour pouvoir exécuter ces actions, ce profil bénéficie automatiquement de la permission "Visualiser les workflows". L'administrateur built-in est toujours autorisé à gérer l'allocation des workflows.

Gestion de l'allocation > Allouer les bons de travail

Définit le profil autorisé à allouer les bons de travail. Pour pouvoir exécuter cette action, ce profil bénéficie automatiquement de la permission "Visualiser les workflows". Pour allouer un bon de travail, un bouton est disponible dans la section "Monitoring > Bons de travail" uniquement pour les bons de travail proposés. Un profil qui a la permission "Gestion de l'allocation" est automatiquement autorisé à effectuer cette action spécifique. L'administrateur built-in est toujours autorisé à allouer les bons de travail.

Gestion de l'allocation > Réallouer les bons de travail

Définit le profil autorisé à réallouer les bons de travail. Pour pouvoir exécuter cette action, ce profil bénéficie automatiquement de la permission "Visualiser les workflows". Pour réallouer, un bouton est disponible dans la section "Monitoring > Bons de travail" uniquement pour les bons de travail alloués. Un profil qui a la permission "Gestion de l'allocation" est automatiquement autorisé à effectuer cette action spécifique. L'administrateur built-in est toujours autorisé à réallouer les bons de travail.

Gestion de l'allocation > Désallouer les bons de travail

Définit le profil autorisé à désallouer les bons de travail. Pour pouvoir exécuter cette action, ce profil bénéficie automatiquement de la permission "Visualiser les workflows". Pour désallouer, un bouton est disponible dans

la section "Monitoring > Bons de travail" uniquement pour les bons de travail alloués. Un profil qui a la permission "Gestion de l'allocation" est automatiquement autorisé à effectuer cette action spécifique. L'administrateur built-in est toujours autorisé à désallouer les bons de travail.

Lancer les workflows	Définit le profil autorisé à lancer manuellement de nouveaux workflows. Cette permission permet de lancer des workflows à partir des publications actives dans la section "Lanceurs de workflow". L'administrateur built-in est toujours autorisé à lancer les workflows.
Visualiser les workflows	Définit le profil autorisé à visualiser les workflows. Par défaut, un utilisateur final peut uniquement voir les bons de travail qui lui sont proposés ou alloués dans la section "Boîte de réception". Cette permission permet en plus de visualiser les publications, workflows et bons de travail associés à ce modèle de workflow dans les sections "Monitoring" et "Workflows terminés". Ce profil bénéficie automatiquement de la permission "Visualiser les workflows terminés". L'administrateur built-in est toujours autorisé à visualiser les workflows.
Visualiser les workflows > Le créateur d'un workflow peut le visualiser	Si activé, le créateur d'un workflow a la permission de visualiser les workflows qu'il a lancés. Cette permission restreinte lui donne accès aux workflows qu'il a lancés et aux bons de travail associés dans les sections "Monitoring > Workflows actifs", "Monitoring > Bons de travail" et "Workflows terminés". La valeur par défaut est "Non".
Visualiser les workflows > Visualiser les workflows terminés	Définit le profil autorisé à visualiser les workflows terminés. Cette permission permet de visualiser les workflows terminés dans la section "Workflows terminés" et d'accéder à leur historique. Un profil qui a la permission "Visualiser les workflows" est automatiquement autorisé à effectuer cette action. L'administrateur built-in est toujours autorisé à visualiser les workflows terminés.

Note

Un utilisateur n'ayant aucun privilège particulier pourra voir un bon de travail uniquement si celui-ci lui est proposé ou personnellement alloué.

Voir aussi [Administration d'un workflow](#) [p 141]

25.4 Historique des images du modèle de workflow

L'historique des images d'un modèle de workflow peut être géré sous **Actions > Historique**.

La table d'historique affiche toutes les images contenant le modèle de workflow et indique si le modèle a été publié. Pour chaque image, il est possible d'exporter ou d'afficher le modèle de workflow correspondant en utilisant le bouton **Actions**.

25.5 Suppression d'un modèle de workflow

Un modèle de workflow peut être supprimé. Cependant, toute version publiée auparavant reste accessible dans la section Workflows de Données. D'autre part, si un modèle de workflow est recréé avec un nom identique, lors d'une publication, un message demandera la confirmation de remplacement de la publication précédente.

Voir aussi [Publication d'un modèle de workflow](#) [p 125]

CHAPITRE 26

Publication d'un modèle de workflow

Ce chapitre contient les sections suivantes :

1. [A propos des publications de workflow](#)
2. [Publication et images de modèle de données](#)
3. [Sous-workflows dans les publications](#)

26.1 A propos des publications de workflow

Dès qu'un modèle de workflow est défini, il doit être publié afin d'autoriser les utilisateurs à lancer des workflows de données associés. Une publication est réalisée en cliquant sur le bouton 'Publier' dans le panneau de navigation.

Si aucune étape d'appel à des sous-workflows n'est incluse dans le modèle courant, l'option de publier d'autres modèles en même temps vous sera proposée sur la page de publication. Si le modèle de workflow actuel contient des étapes d'appel à des sous-workflows, il doit être publié seul.

Les modèles de workflow peuvent être publiés plusieurs fois. Une publication est identifiée par son nom de publication.

26.2 Publication et images de modèle de données

Durant la publication d'un modèle de workflow, une image est enregistrée de son état actuel. Vous pouvez préciser un libellé et une description de l'image. Le libellé par défaut pour l'image est l'heure et la date au moment de publication. La description par défaut indique l'utilisateur qui a publié le modèle de workflow.

Pour chaque modèle de workflow publié, le nom de la publication doit être unique. Si un modèle de workflow a déjà été publié, il est possible de mettre à jour une publication existante en réutilisant le même nom de publication. Les noms des publications de workflow existantes sont proposés dans un menu. Dans le cas d'une mise à jour de publication, l'ancienne version ne sera plus disponible pour lancer des workflows de données ; mais elle permettra aux workflows existants de finir de s'exécuter. Le contenu des différentes images peut être consulté dans l'historique des images de modèle de workflow.

Voir aussi [Historique des images du modèle de workflow](#) [p 122]

26.3 Sous-workflows dans les publications

Quand un modèle de workflow, contenant un appel à des sous-workflows, est publié, il n'est pas nécessaire de publier séparément les sous-workflows appelés. D'un point de vue d'administration, le modèle du workflow principal (celui actuellement publié par l'utilisateur) et les modèles de sous-workflows sont publiés comme une entité unique.

La publication multiple n'est pas disponible pour un modèle de workflow contenant un appel à des sous-workflows. C'est pourquoi la première étape de publication (sélection des modèles de workflow à publier) n'est pas proposée dans ce cas.

La republication du modèle de workflow principal met automatiquement à jour les modèles de sous-workflows appelés.

Bien qu'un sous-workflow puisse être publié séparément comme modèle de workflow principal, cela ne modifiera pas la version utilisée par un autre modèle de workflow principal publié qui utilise ce sous-workflow.

Workflows de données

CHAPITRE 27

Introduction aux workflows de données

Ce chapitre contient les sections suivantes :

1. [Présentation](#)

27.1 Présentation

Un workflow de données est un processus exécuté sous la forme d'une succession d'étapes, définie par une publication de modèle de workflow. Le workflow de données permet aux utilisateurs, ainsi qu'aux procédures automatisées, d'effectuer des actions de façon collaborative sur les données. Une fois le modèle de workflow spécifié et publié, la publication résultante peut être utilisée pour lancer un workflow de données afin d'exécuter les étapes définies.

En fonction des permissions définies par le modèle de workflow, un utilisateur peut effectuer une ou plusieurs des actions suivantes sur les workflows de données associés :

- en tant qu'utilisateur avec les permissions par défaut, effectuer les actions attendues par les bons de travail qui lui sont destinés,
- en tant qu'utilisateur avec les permissions pour lancer les workflows, créer les nouveaux workflows de données depuis une publication du modèle de workflow,
- en tant qu'utilisateur avec les permissions de monitoring de workflow, suivre l'avancement des workflows de données en cours, et consulter l'historique des workflows de données terminés.
- en tant que gestionnaire d'allocation des bons de travail, modifier les allocations des bons de travail des autres utilisateurs manuellement.
- en tant qu'administrateur de workflow, effectuer différentes actions d'administration, comme par exemple, redémarrer une étape, terminer un workflow en cours ou rendre une publication indisponible pour le lancement de workflows.

Voir aussi

[Bons de travail](#) [p 135]

[Lancement et monitoring de workflows de données](#) [p 139]

[Administration de workflows de données](#) [p 141]

[Permissions sur les workflows de données associés](#) [p 120]

Concepts apparentés [*Modèles de workflow*](#) [p 104]

CHAPITRE 28

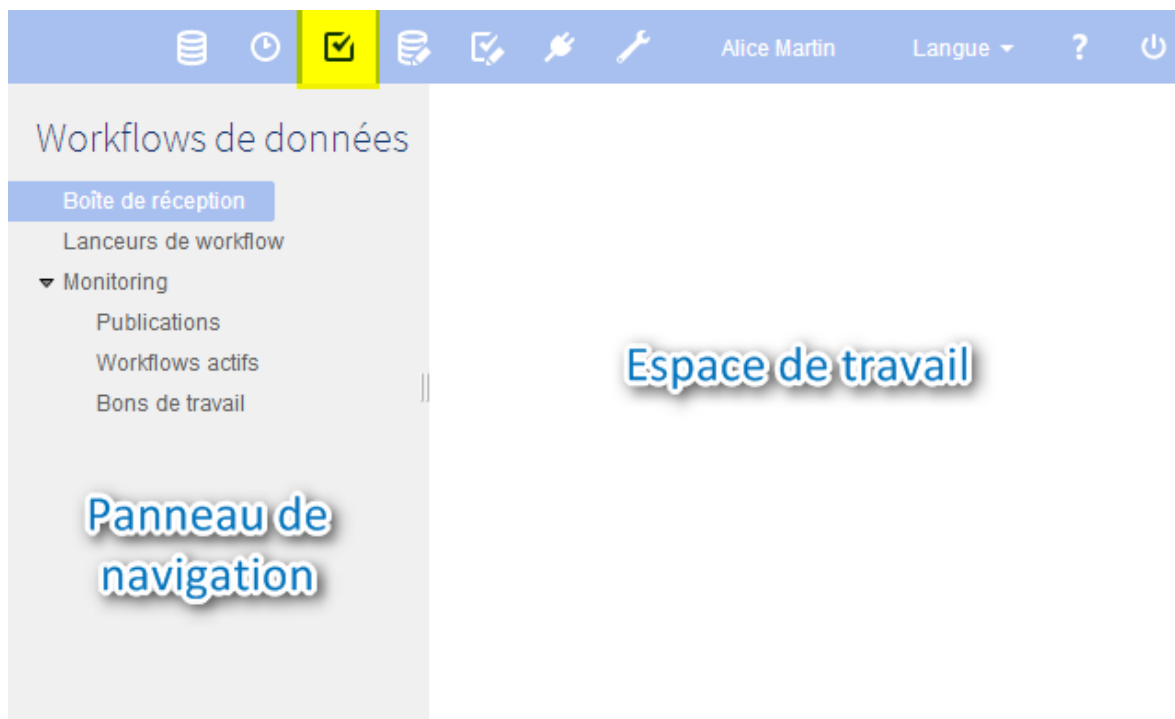
Utilisation de l'interface utilisateur de la section Workflow de données

Ce chapitre contient les sections suivantes :

1. [Navigation dans l'interface utilisateur](#)
2. [Règles de navigation](#)
3. [Filtrage d'items dans les vues](#)
4. [Vue graphique d'un workflow de données](#)

28.1 Navigation dans l'interface utilisateur

La fonctionnalité workflows de données se trouve dans la section **Workflows de données** dans l'interface utilisateur d'EBX5.



Note

Seuls les utilisateurs autorisés peuvent accéder à cet écran via la 'Perspective avancée' ou via une perspective spécifique. Seuls les utilisateurs autorisés peuvent accéder à ces interfaces spécifiques.

Le panneau de navigation est composé de plusieurs items. Ces items sont visibles en fonction des permissions globales associées. Les différents items sont :

Boîte de réception des bons de travail	Tous les bons de travail qui vous sont alloués ou proposés, pour lesquels vous devez effectuer les actions spécifiées.
Lanceurs de workflow	Liste des publications de workflow à partir desquelles vous pouvez lancer des workflows de données, en fonction de vos permissions.
Monitoring	Vues pour le monitoring des workflows de données en cours pour lesquels vous avez les permissions nécessaires de visualisation.
Publications	Publications pour lesquelles vous avez les permissions nécessaires de visualisation. Si vous disposez de permissions d'administration supplémentaires, vous pouvez également désactiver la possibilité de lancer les workflows de données depuis une publication à partir de cette vue.
Workflows actifs	Workflows de données, en cours d'exécution, pour lesquels vous avez les permissions nécessaires de visualisation. Si vous disposez de permissions d'administration supplémentaires, vous pouvez également effectuer des actions d'administration à partir de cette vue, comme par exemple redémarrer une étape, ou terminer un workflow en cours.
Bons de travail	Bons de travail pour lesquels vous avez les permissions nécessaires de visualisation. Si vous disposez de permissions d'administration supplémentaires, vous pouvez également effectuer des actions d'administration à partir de cette vue, comme par exemple allouer les bons de travail aux utilisateurs ou rôles.
Workflows terminés	Workflows de données, dont l'exécution est terminée, pour lesquels vous avez les permissions nécessaires de visualisation. Si vous disposez de permissions d'administration supplémentaires, vous pouvez également nettoyer les workflows terminés à partir de cette vue.

28.2 Règles de navigation

Corbeille de bons de travail

Par défaut, une fois qu'un bon de travail est exécuté, la corbeille de bons de travail est affichée.

Ce comportement peut être modifié en fonction de la stratégie d'avancement de l'étape suivante. Cette stratégie permet d'enchaîner plusieurs étapes à la suite sans repasser par la corbeille de tâches.

Voir [la stratégie d'avancement d'une étape de workflow](#) [p 110] dans la modélisation de workflow.

Lanceurs de workflow

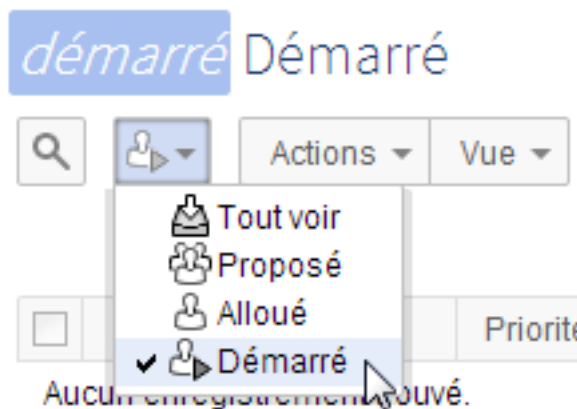
Par défaut, une fois qu'un workflow est lancé, la table des lanceurs de workflow est affichée.

Ce comportement est modifiable dans la configuration du modèle : il est possible d'ouvrir directement la première étape sans afficher la table des lanceurs de workflow.

Voir [l'ouverture automatique de la première étape d'un workflow](#) [p 118] dans la modélisation de workflow.


28.3 Filtrage d'items dans les vues

Dans certaines vues, comme la 'Boîte de réception' des bons de travail, vous pouvez filtrer les lignes affichées dans les tables en fonction de leur état. Dans ces vues, un menu est disponible à cet effet pour sélectionner l'état correspondant au filtre attendu.



28.4 Vue graphique d'un workflow de données

En tant qu'utilisateur avec un bon de travail à effectuer, ou en tant que superviseur ou administrateur de workflow de données, vous pouvez suivre l'avancement ou l'historique d'exécution d'un workflow

de données. Pour cela, cliquer sur le bouton 'Afficher'  dans la colonne 'Workflow de données' des tables de l'interface de workflows de données. Ce bouton ouvre une pop-up qui affiche une vue interactive graphique de l'exécution du workflow de données. Dans cette vue, vous pouvez suivre l'avancement global de l'exécution, et sélectionner une étape individuelle afin de consulter le détail de ses informations.

CHAPITRE 29

Bons de travail

Ce chapitre contient les sections suivantes :

1. [Présentation des bons de travail](#)
2. [Travail sur un bon de travail en tant que participant](#)
3. [Priorité de bons de travail](#)

29.1 Présentation des bons de travail

Un bon de travail est une tâche utilisateur unitaire qui doit être effectuée par un utilisateur humain. Par défaut, quand un modèle de workflow définit une tâche utilisateur, les workflows de données, lancés depuis les publications du modèle, génèrent un bon de travail individuel pour chacun des participants listés dans la tâche utilisateur.

Etats d'un bon de travail

Lorsqu'un bon de travail est émis, pendant l'exécution d'un workflow de données, pour une tâche utilisateur définie dans le modèle, le bon de travail peut prendre plusieurs états : proposé, alloué, démarré, et terminé.

Par défaut, pour chaque utilisateur défini comme participant de la tâche utilisateur, le workflow de données crée un bon de travail dans l'état *alloué*. L'utilisateur défini peut directement commencer à travailler sur le bon de travail alloué en effectuant l'action 'Prendre et démarrer'. Le bon de travail passe alors à l'état *démarré*.

Par défaut, pour chaque rôle défini comme participant de la tâche utilisateur, le workflows de données crée un bon de travail dans l'état *proposé*. Tout membre du rôle peut prendre le bon de travail en utilisant l'action 'Prendre et démarrer'. Le bon de travail passe ainsi à l'état *démarré*.

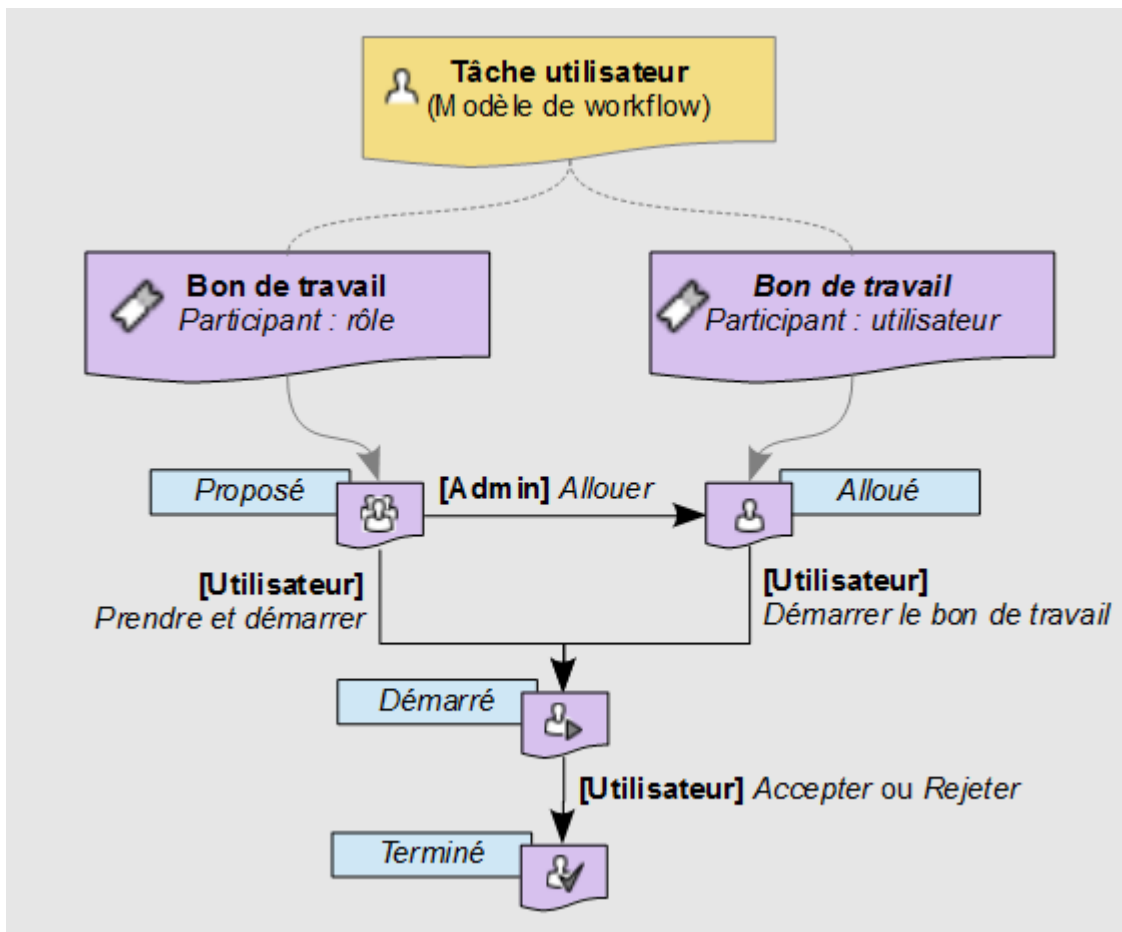
Avant qu'un utilisateur ait pris le bon de travail proposé, un gestionnaire du workflow de données peut intervenir pour affecter manuellement le bon de travail à un utilisateur spécifique, passant ainsi le bon de travail à l'état *alloué*. Puis, lorsque l'utilisateur commence à travailler sur le bon de travail via l'action 'Démarrer le bon de travail', le bon de travail passe à l'état *démarré*.

Note

Le comportement par défaut, décrit ci-dessus, peut être surchargé par une extension programmatique définie dans la tâche utilisateur. Dans ce cas, les bons de travail peuvent être générés programmatiquement, sans se baser obligatoirement sur la liste des participants de la tâche utilisateur.

Une fois que l'utilisateur, qui a démarré le bon de travail, a réalisé l'action demandée, l'action terminale 'Accepter' ou 'Rejeter' place le bon de travail dans l'état *terminé*. Lorsqu'un utilisateur termine un bon de travail, le workflow de données passe automatiquement à l'étape suivante définie dans le modèle de workflow.


Diagramme des états de bon de travail



29.2 Travail sur un bon de travail en tant que participant

Tous les bons de travail disponibles (qui vous sont soit proposés, soit alloués), sont affichés dans votre boîte de réception des bons de travail. Quand vous commencez à travailler sur un bon de travail, vous pouvez ajouter des commentaires associés qui seront visibles par les administrateurs, les superviseurs du workflow et les autres participants au workflow de données. Tant que vous êtes toujours en train de travailler sur le bon de travail, vous pouvez éditer ce commentaire.

Quand vous avez réalisé toutes les actions demandées par le bon de travail, vous devez signaler la fin du travail en cliquant soit sur le bouton **Accepter**, soit sur le bouton **Rejeter**. Les libellés de ces deux boutons peuvent varier en fonction du contexte du bon de travail.

Pour suivre l'avancement du workflow de données associé au bon de travail qui vous est destiné dans votre boîte de réception, cliquez sur le bouton "Afficher"  dans la colonne 'Workflow de données' de la table. Une pop-up affichera une vue graphique interactive du workflow de données.

jusqu'au moment actuel ainsi que les étapes à venir. Vous pouvez visualiser les détails d'une étape en sélectionnant l'étape.

Note

Si vous interrompez la session actuelle pendant un bon de travail, par exemple en fermant le navigateur ou en déconnectant, l'état actuel du bon de travail est préservé. Quand vous revenez sur le bon de travail, il continue à partir du même point.

29.3 Priorité de bons de travail

Les bons de travail peuvent porter une priorité, qui peut être utile pour trier et filtrer les bons de travail à compléter. La priorité d'un bon de travail est définie au niveau de son workflow de données, et n'est pas spécifique au bon de travail lui-même. Par conséquent, si le workflow de données est considéré comme urgent, tous les bons de travail ouverts associés sont aussi considérés comme urgent. Par défaut, il y a six niveaux de priorité, de "Très peu prioritaire" à "Urgent". Cependant, la représentation visuelle et le nommage des priorités dépendent de la configuration de votre référentiel EBX5.

Voir aussi [tâche utilisateur \(glossaire\)](#) [p 23]

Concepts apparentés [Tâche utilisateur](#) [p 110]

CHAPITRE 30

Lancement et monitoring de workflows de données

Ce chapitre contient les sections suivantes :

1. [Lancement d'un workflow de données](#)
2. [Activités de monitoring](#)
3. [Gestion de l'allocation des bons de travail](#)

30.1 Lancement d'un workflow de données

Quand un modèle de workflow vous autorise à lancer des workflows de données depuis ses publications, vous pouvez créer de nouveaux workflows en utilisant la vue 'Lanceurs de workflow' dans le panneau de navigation. Pour créer un nouveau workflow de données, depuis une publication de modèle de workflow, cliquer sur le bouton 'Lancer' de la ligne de la publication cible dans la table.

Vous pouvez alors définir des libellés et descriptions localisés pour le nouveau workflow de données que vous lancez.

30.2 Activités de monitoring

Quand un modèle de workflow vous donne les permissions de monitoring de workflow, vous avez la possibilité de suivre l'avancement des workflows de données qui sont en cours d'exécution. Vous pouvez accéder aux vues de monitoring, dans la section 'Monitoring' du panneau de navigation. Si vous disposez également de permissions d'administration de workflow, vous pouvez effectuer les actions autorisées associées depuis ces vues.

Lorsqu'un workflow de données, que vous êtes autorisé à suivre, a fini son exécution, il est affiché dans 'Workflows terminés', où vous pouvez consulter son historique.

30.3 Gestion de l'allocation des bons de travail

Lorsque vous êtes autorisé à gérer l'allocation des bons de travail, vous pouvez allouer manuellement des bons de travail durant l'exécution des workflows associés au modèle de workflow. Dans ce cas, vous pouvez effectuer une ou plusieurs des actions ci-dessous sur les bons de travail.

Sélectionnez 'Bons de travail' dans la section 'Monitoring' du panneau de navigation. Les actions que vous pouvez effectuer sont affichées dans le menu 'Actions' de l'entrée du bon de travail, selon son état actuel, dans la table.

Allouer	Allouer un bon de travail à un utilisateur spécifique. Cette action est disponible pour les bons de travail dans l'état <i>proposé</i> .
Désallouer	Remettre un bon de travail qui est actuellement dans l'état <i>alloué</i> dans l'état <i>proposé</i> .
Réallouer	Modifier l'utilisateur à qui le bon de travail est alloué. Cette action est disponible pour les bons de travail dans l'état <i>alloué</i> .

Voir aussi

[Bons de travail](#) [p 135]

[Permissions sur les workflows de données associés](#) [p 120]

Concepts apparentés [Modèles de workflow](#) [p 104]

CHAPITRE 31

Administration de workflows de données

Si vous disposez de permissions pour administrer des workflows de données, les vues 'Publications', 'Workflows actifs', et 'Bons de travail' associés seront accessibles sous le menu 'Monitoring' du panneau de navigation. Dans ces vues, sous les menus 'Actions' sur les lignes des tables, vous pourrez accéder aux actions d'administration.

Note

Quand un modèle de données vous donne des droits d'administration, vous aurez automatiquement les permissions de monitoring sur tous les objets associés à l'exécution de workflow, comme les publications, les workflows actifs, et les bons de travail.

Ce chapitre contient les sections suivantes :

1. [Présentation de l'exécution de workflow de données](#)
2. [Actions d'administration de workflow de données](#)

31.1 Présentation de l'exécution de workflow de données

Quand un workflow de données est lancé, un *jeton* qui représente l'étape en cours d'exécution est créé et positionné au début du workflow. A chaque fois qu'une étape est terminée, ce jeton se déplace sur la prochaine étape définie par le modèle de workflow associé à la publication du workflow de données.

Pendant l'exécution d'un workflow de données, le jeton est positionné sur un des types d'étape suivants:

- une tâche automatique, qui est lancée automatiquement et n'a pas besoin d'interaction utilisateur. La tâche automatique est terminée quand les actions définies finissent leur exécution.
- une tâche utilisateur, qui génère un ou plusieurs bons de travail effectués manuellement par les utilisateurs. Chaque bon de travail est terminé par une action 'Accepter' ou 'Rejeter', réalisée explicitement par l'utilisateur. La fin de la tâche utilisateur chapeau est déterminée en fonction du critère de fin de tâche défini pour la tâche utilisateur dans le modèle de workflow.
- une condition, qui est évaluée automatiquement afin de déterminer l'étape suivante de l'exécution du workflow de données.
- invocation de sous-workflows qui lance les sous-workflows associés et attend que les sous-workflows en cours soient terminés.
- tâche d'attente qui met en pause le workflow jusqu'à ce qu'un événement spécifique soit reçu.

Le jeton peut être dans les états suivants :

- **A exécuter** : Le jeton est en train de passer à la prochaine étape, en se basant sur le modèle de workflow.
- **En cours d'exécution** : Le jeton est positionné sur une tâche automatique ou une condition en train de s'exécuter.
- **Utilisateur** : Le jeton est positionné sur une tâche utilisateur et attend une action utilisateur.
- **En attente de sous-workflows** : Le jeton est positionné sur une invocation de sous-workflows et attend la terminaison de tous les sous-workflows lancés.
- **En attente d'événement** : Le jeton est positionné sur une tâche d'attente et attend de recevoir un événement donné.
- **Terminé** : Le jeton a atteint la fin du workflow de données.
- **Erreur** : Une erreur est survenue.

31.2 Actions d'administration de workflow de données

Actions sur les publications

Désactivation d'une publication de workflow

Afin d'éviter que de nouveaux workflows de données soient lancés depuis une publication de workflow, vous pouvez désactiver la publication. Sélectionnez la vue 'Publications' dans la panneau de navigation, puis sélectionnez *Actions > Désactiver* sur la ligne de la publication cible.

Une fois désactivée, la publication n'apparaîtra plus dans la vue 'Lanceurs de workflow' des utilisateurs. Toutefois, les workflows de données déjà lancés vont continuer à s'exécuter.

Note

Suite à la désactivation d'une publication, il n'est pas possible de la réactiver à partir de la section 'Workflows de données'. Seul un utilisateur avec le rôle built-in 'Administrateur' peut réactiver une publication inactive dans la section 'Administration'. Cependant, il n'est pas conseillé de modifier les tables techniques manuellement, car il est important de préserver l'intégrité des données techniques des workflows.

Dépublication d'une publication de workflow

Si une publication de workflow n'est plus utilisée, vous pouvez la supprimer de toutes les vues de la section 'Workflows de données' en la dépubliant. Pour faire cela,

1. Désactivez la publication de workflow afin d'éviter que des utilisateurs continuent de lancer des nouveaux workflows de données sur cette publication. Pour cela, suivez le processus décrit dans la section [Désactivation d'une publication de workflow](#) [p 142].
2. Dépublier la publication de workflow en sélectionnant *Actions > Dépublier* de la ligne de la publication cible.

Note

A la dépublication d'une publication de workflow, une confirmation vous sera demandée pour terminer et purger tous les workflows de données en cours qui ont été lancés depuis cette publication de workflow, ainsi que tout bon de travail associé. Toute perte de données résultant d'une fin prématurée est alors définitive.

Actions sur workflows de données

Dans les vues tabulaires des workflows de données, chaque enregistrement porte un menu *Actions* qui permet d'exécuter des services sur un workflow de données.

Ré-exécution d'une étape

Dans le cas d'une erreur inattendue pendant l'exécution d'une étape, par exemple, à cause d'un problème de permissions ou de ressources non disponibles, vous pouvez "rejouer" une étape en tant qu'administrateur de workflow. En rejouant une étape, l'environnement d'exécution associé est nettoyé, notamment les bons de travail et sous-workflows liés, et le jeton est repositionné au début de l'étape courante.

Pour rejouer l'étape courante dans un workflow de données, sélectionnez *Actions > Rejouer l'étape* dans la ligne du workflow cible dans la table 'Workflows actifs'.

Terminer un workflow de données actif et le purger

Pour terminer un workflow de données en cours d'exécution, sélectionnez *Actions > Terminer et purger* dans la ligne du workflow cible dans la table 'Workflows actifs'. L'action stoppe l'exécution du workflow de données et supprime le workflow, tous les bons de travail et sous-workflows associés.

Note

Cette action n'est pas disponible pour les workflows dans l'état 'En cours d'exécution' et pour les sous-workflows lancés par d'autres workflows.

Forcer la terminaison d'un workflow de données actif

Pour forcer la terminaison d'un workflow de données en cours d'exécution, sélectionnez *Actions > Forcer la terminaison* dans la ligne du workflow cible dans la table 'Workflows actifs'. L'action

stoppe l'exécution du workflow de données et supprime les éventuels bons de travail et sous-workflows associés.

Note

Cette action est disponible pour les sous-workflows, et pour les workflows en erreur bloqués sur la dernière étape.

Forcer le réveil d'un workflow en attente

Pour réveiller un workflow qui est en attente d'événement, sélectionner **Actions > Forcer le réveil** à partir du workflow dans la table 'Workflows actifs'. Cela entraîne le réveil du workflow. Avant d'effectuer cette action, l'administrateur doit mettre à jour le contexte de données afin de s'assurer que le workflow peut exécuter les tâches suivantes.

Note

Cette action est disponible uniquement pour les workflows qui sont à l'état 'en attente d'événement'.

Purge d'un workflow de données terminé

Quand un workflow de données a terminé son exécution, son historique est visible pour ses superviseurs et administrateurs dans la vue 'Workflows terminés'. Pour purger le workflow terminé et son historique, vous pouvez effectuer un nettoyage en sélectionnant *Actions > Purger* dans la ligne du workflow cible de la table 'Workflows terminés'.

Note

Cette action n'est pas disponible pour les sous-workflows lancés par d'autres workflows.

Modification de la priorité d'un workflow de données

Suite au lancement d'un workflow de données, un administrateur du workflow peut modifier son niveau de priorité. En modifiant la priorité du workflow de données, la priorité de tous les bons de travail existants et à venir de ce workflow sera modifiée. Pour modifier la priorité d'un workflow de données, sélectionnez *Actions > Modifier la priorité* dans la ligne du workflow cible dans la table 'Workflows actifs'.

Voir aussi [Permissions sur les workflows de données associés](#) [p 120]

Services de données

CHAPITRE 32

Introduction aux services de données

Ce chapitre contient les sections suivantes :

1. [Présentation](#)
2. [Utilisation de l'interface utilisateur de la section Services de données](#)

32.1 Présentation

Fonction du service de données

Un [service de données](#) [p 24] est un web service standard qui permet d'interagir avec EBX5. Il peut être utilisé pour accéder à une partie des fonctionnalités disponibles par l'interface utilisateur.

Les services de données peuvent être générés dynamiquement à partir d'un modèle de données dans la section 'Services de données'.

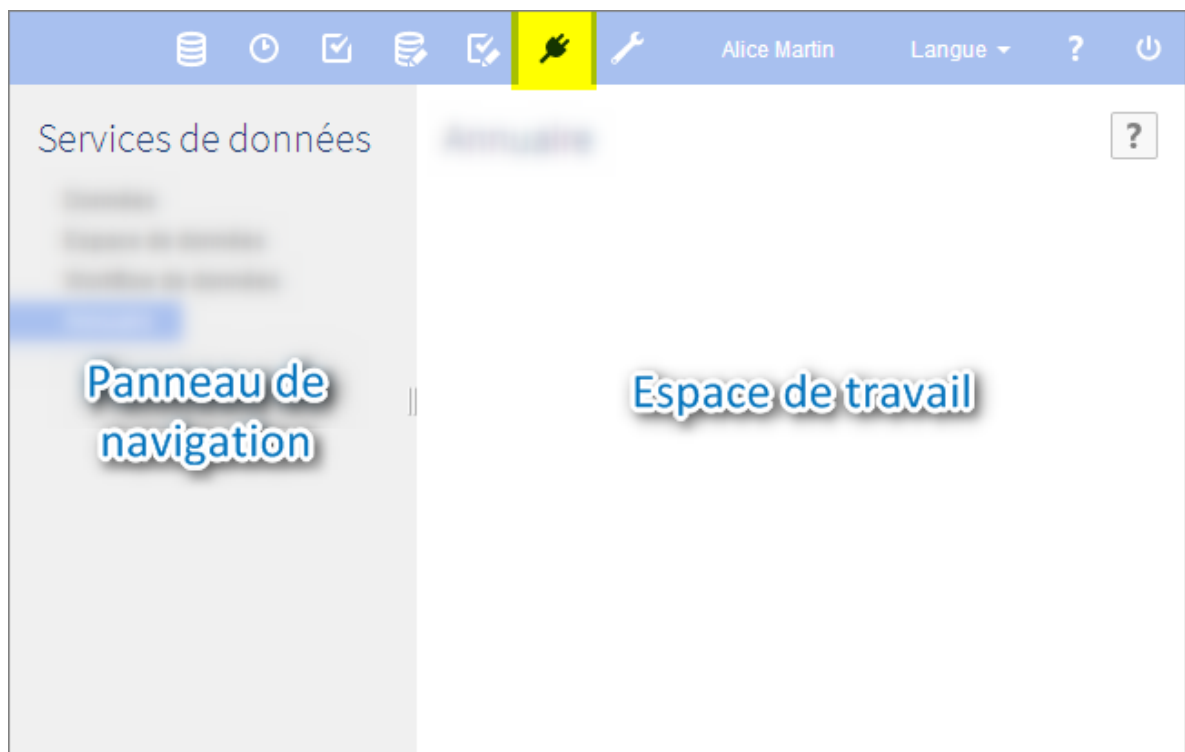
Lignage

Le [lignage](#) [p 25] établit des profils de droit d'accès utilisés par les services de données. Quand les services de données accèdent aux interfaces WSDL, ils utilisent les profils de droit d'accès définis par ce mécanisme.

Glossaire

Voir aussi [Services de données](#) [p 24]

32.2 Utilisation de l'interface utilisateur de la section Services de données



Note

Seuls les utilisateurs autorisés peuvent accéder à cette section via la 'Perspective avancée'.

Concepts apparentés

[Espaces de données](#) [p 62]

[Jeux de données](#) [p 80]

[Workflows de données](#) [p 128]

CHAPITRE 33

Génération de WSDL pour services de données

Ce chapitre contient les sections suivantes :

1. [Générer un WSDL pour accéder aux données](#)
2. [Générer un WSDL pour accéder à un espace de données](#)
3. [Générer un WSDL pour contrôler un workflow de données](#)
4. [Générer un WSDL pour un lignage](#)

33.1 Générer un WSDL pour accéder aux données

La génération de WSDL pour l'accès aux données est disponible en sélectionnant 'Données' dans la section 'Services de données'.

Les étapes de la génération d'un WSDL sont les suivantes :

1. Sélectionner si le WSDL sera utilisé pour des opérations sur un jeu de données ou sur une table.
2. Identifier l'espace de données et le jeu de données ciblés par les opérations.
3. Sélectionner les tables sur lesquelles les opérations sont autorisées, ainsi que les opérations permises.
4. Télécharger le fichier WSDL généré en cliquant sur le bouton 'Télécharger le WSDL'.

Opérations disponibles sur un jeu de données

Les opérations suivantes sur les jeux de données sont disponibles en utilisant le WSDL généré :

- Récupérer les changements sur table(s) du jeu de données entre espaces de données ou images
- Actualiser une unité de réplication en base de données

Opérations disponibles sur une table d'un jeu de données

Si sélectionnées, les opérations suivantes sur les tables sont disponibles en utilisant le WSDL généré :

- Sélectionner un (ou plusieurs) enregistrement(s)
- Insérer un (ou plusieurs) enregistrement(s)
- Mettre à jour un (ou plusieurs) enregistrement(s)
- Supprimer un (ou plusieurs) enregistrement(s)
- Compter des enregistrements
- Récupérer les changements entre espaces de données ou images
- Obtenir les droits d'accès

33.2 Générer un WSDL pour accéder à un espace de données

La génération de WSDL pour la manipulation d'un espace de données est accessible en sélectionnant 'Espace de données' dans la section 'Services de données'. Le WSDL généré n'est pas spécifique à un espace de données et aucune information n'est requise. Il peut être téléchargé grâce au bouton **Télécharger le WSDL**.

Opérations disponibles sur un espace de données

Les opérations suivantes sur les espaces de données sont disponibles en utilisant le WSDL généré :

- Créer un espace de données
- Créer une image
- Fermer un espace de données
- Fermer une image
- Fusionner un espace de données
- Valider un espace de données ou une image
- Valider un jeu de données
- Verrouiller un espace de données
- Déverrouiller un espace de données

33.3 Générer un WSDL pour contrôler un workflow de données

La génération d'un WSDL pour le contrôle d'un workflow est accessible en sélectionnant 'Workflow de données' dans la section 'Services de données'. Le WSDL généré n'est pas spécifique à une publication de workflow et aucune information n'est requise. Il peut être téléchargé grâce au bouton **Télécharger le WSDL**.

Opérations disponibles pour contrôler un workflow de données

Les opérations suivantes sur les espaces de données sont disponibles en utilisant le WSDL généré :

- Démarrer un workflow
- Réveiller un workflow
- Terminer un workflow

33.4 Générer un WSDL pour un lignage

La génération d'un WSDL pour un lignage est accessible en sélectionnant 'Lignage' dans la section 'Services de données', sous réserve que des profils aient été autorisés par un profil administrateur dans *Administration > Lignage*.

Les WSDL générés pour accéder aux tables sont les mêmes que ceux utilisés pour la [génération d'un WSDL d'accès aux données](#) [p 149].

Les étapes de la génération de ce WSDL sont les suivantes :

1. Sélectionner un rôle ou un utilisateur, dont les permissions seront appliquées. Un rôle ou un utilisateur doit être autorisé à être utilisé pour le lignage par un administrateur.
2. Identifier l'espace de données et le jeu de données ciblés par les opérations.
3. Sélectionner les tables sur lesquelles les opérations sont autorisées, ainsi que les opérations permises.
4. Télécharger le fichier WSDL généré en cliquant sur le bouton **Télécharger le WSDL**.

Voir aussi [Lignage](#) [p 146]

Manuel de référence (en anglais)

Intégration

CHAPITRE 34

Built-in UI services

EBX5 includes a number of built-in UI services. Built-in UI services can be used:

- when defining [workflow model tasks](#) [p 110]
- when defining [perspective action menu items](#) [p 12]

This reference page describes the built-in UI services and their parameters.

Ce chapitre contient les sections suivantes :

1. [Access a data space](#)
2. [Access data \(default service\)](#)
3. [Access the data space merge view](#)
4. [Compare contents](#)
5. [Create a new record](#)
6. [Data workflows](#)
7. [Duplicate a record](#)
8. [Export data to a CSV file](#)
9. [Export data to an XML file](#)
10. [Import data from a CSV file](#)
11. [Import data from an XML file](#)
12. [Merge a data space](#)
13. [Validate a data space, a snapshot or a data set](#)

34.1 Access a data space

A workflow automatically considers that the data space selection service is complete.

Service name parameter: `service=@selectDataSpace`

Input parameters

Parameter	Label	Description
branch	Data space	The identifier of the specified data space - A data space is required for this service.
scope	Scope	Defines the scope of the user navigation for this service, namely, the entities that the user is able to select during their session. If left blank, the default value will be used. For perspectives, the default value is always 'node'. For workflows, the default value depends on the selected entities or service.
trackingInfo	Tracking information	Tracking information is logged into 'history' logs. It may also be used for any other purpose like access control or additional export information.
version	Snapshot	The identifier of the specified snapshot - A data space is required for this service.

34.2 Access data (default service)

The default service. By default, workflows automatically consider this service as complete. That is, the 'Accept' button is always available.

This is the default service used if no service is specified.

Input parameters

Parameter	Label	Description
branch	Data space	The identifier of the specified data space - A data space is required for this service.
disableAutoComplete	Disable Accept at start	By default, the interaction associated with this service is directly considered as complete. Therefore, the Accept button is automatically displayed at the opening of the work item. This parameter is useful to disable this behavior. If the value is 'true', the developer will be in charge of completing the interaction by using SessionInteraction in a UI service or a trigger, for example. The default value is 'false'. Perspectives do not use this parameter.
instance	Data set	The value must be the reference of a data set that exists in the selected data space.
scope	Scope	Defines the scope of the user navigation for this service, namely, the entities that the user is able to select during their session. If left blank, the default value will be used. For perspectives, the default value is always 'node'. For workflows, the default value depends on the selected entities or service.
trackingInfo	Tracking information	Tracking information is logged into 'history' logs. It may also be used for any other purpose like access control or additional export information.
version	Snapshot	The identifier of the specified snapshot - A data space is required for this service.
viewPublication	View	The publication name of the view to display. The view must be configured for the selected table.
xpath	Data set node (XPath)	The value must be a valid absolute location path in the selected data set. The notation must conform to a simplified XPath, in its abbreviated syntax.

34.3 Access the data space merge view

The merge.view service is automatically considered complete.

Service name parameter: `service=@merge.view`

Input parameters

Parameter	Label	Description
branch	Data space	The identifier of the specified data space - A data space is required for this service.
scope	Scope	Defines the scope of the user navigation for this service, namely, the entities that the user is able to select during their session. If left blank, the default value will be used. For perspectives, the default value is always 'node'. For workflows, the default value depends on the selected entities or service.
trackingInfo	Tracking information	Tracking information is logged into 'history' logs. It may also be used for any other purpose like access control or additional export information.

34.4 Compare contents

Workflows automatically consider the compare service as complete.

Service name parameter: `service=@compare`

Input parameters

Parameter	Label	Description
branch	Data space	The identifier of the specified data space - A data space or snapshot and a data space or snapshot to compare to are required for this service.
compare.branch	Data space to compare	The identifier of the data space to compare - A data space or snapshot and a data space or snapshot to compare to are required for this service.
compare.filter	Comparison filter	To ignore inheritance and computed values fields in the comparison (disable resolved mode), the filter "persistedValuesOnly" must be specified. By default, when no filter is defined, the comparison uses resolved mode.
compare.instance	Data set to compare	The value must be the reference of a data set that exists in the selected data space to compare.
compare.version	Snapshot to compare	The identifier of the snapshot to compare - A data space or snapshot and a data space or snapshot to compare to are required for this service.
compare.xpath	Table or record to compare (XPath)	The value must be a valid absolute location path of a table or a record in the selected data set to compare. The notation must conform to a simplified XPath, in its abbreviated syntax.
instance	Data set	The value must be the reference of a data set that exists in the selected data space.
scope	Scope	Defines the scope of the user navigation for this service, namely, the entities that the user is able to select during their session. If left blank, the default value will be used. For perspectives, the default value is always 'node'. For workflows, the default value depends on the selected entities or service.
trackingInfo	Tracking information	Tracking information is logged into 'history' logs. It may also be used for any other purpose like access control or additional export information.
version	Snapshot	The identifier of the specified snapshot - A data space or snapshot and a data space or snapshot to compare to are required for this service.
xpath	Table or record (XPath)	The value must be a valid absolute location path of a table or a record in the selected data set. The notation must

Parameter	Label	Description
		conform to a simplified XPath, in its abbreviated syntax.

34.5 Create a new record

For a workflow, the creation service is considered complete when the first successful submit is performed (record has been created). If this service is called whereas it is already complete, the created record is displayed in update or read-only mode (depending on the user rights).

Service name parameter: `service=@creation`

Input parameters

Parameter	Label	Description
branch	Data space	The identifier of the specified data space - This field is required for this service.
instance	Data set	The value must be the reference of a data set that exists in the selected data space - This field is required for this service.
scope	Scope	Defines the scope of the user navigation for this service, namely, the entities that the user is able to select during their session. If left blank, the default value will be used. For perspectives, the default value is always 'node'. For workflows, the default value depends on the selected entities or service.
trackingInfo	Tracking information	Tracking information is logged into 'history' logs. It may also be used for any other purpose like access control or additional export information.
xpath	Data set table (XPath)	The value must be a valid absolute location path of a table in the selected data set. The notation must conform to a simplified XPath, in its abbreviated syntax - This field is required for this service.

Output parameters

Parameter	Label	Description
created	Created record	Contains the XPath of the created record.

34.6 Data workflows

This service provides access to the data workflows user interfaces.

Service name parameter: `service=@workflow`

Note

This service is for perspective only.

Input parameters

Parameter	Label	Description
scope	Scope	Defines the scope of the user navigation for this service.
workflowView	Workflow view	Specifies the workflow view type. Value can be one of the following: "inbox", "launcher", "monitoringPublications", "monitoringWorkflows", "monitoringWorkItems" or "completedWorkflows".
xpath	Filter (XPath)	An optional filter. The syntax should conform to a XPath predicate surrounded by "[" and "]".

34.7 Duplicate a record

For a workflow, the duplicate service is considered complete when the first successful submit is performed (record has been created). If this service is called whereas it is already complete, the created record is displayed in update or read-only mode (depending on the user rights).

Service name parameter: `service=@duplicate`

Input parameters

Parameter	Label	Description
branch	Data space	The identifier of the specified data space - This field is required for this service.
instance	Data set	The value must be the reference of a data set that exists in the selected data space - This field is required for this service.
scope	Scope	Defines the scope of the user navigation for this service, namely, the entities that the user is able to select during their session. If left blank, the default value will be used. For perspectives, the default value is always 'node'. For workflows, the default value depends on the selected entities or service.
trackingInfo	Tracking information	Tracking information is logged into 'history' logs. It may also be used for any other purpose like access control or additional export information.
xpath	Record to duplicate (XPath)	The value must be a valid absolute location path of an existing record. The notation must conform to a simplified XPath, in its abbreviated syntax - This field is required for this service.

Output parameters

Parameter	Label	Description
created	Created record	Contains the XPath of the created record.

34.8 Export data to a CSV file

Workflows consider the exportToCSV service as complete when export is done and file downloaded.

Service name parameter: `service=@exportToCSV`

Input parameters

Parameter	Label	Description
branch	Data space	The identifier of the specified data space - A data space is required for this service.
instance	Data set	The value must be the reference of a data set that exists in the selected data space - This field is required for this service.
scope	Scope	Defines the scope of the user navigation for this service, namely, the entities that the user is able to select during their session. If left blank, the default value will be used. For perspectives, the default value is always 'node'. For workflows, the default value depends on the selected entities or service.
trackingInfo	Tracking information	Tracking information is logged into 'history' logs. It may also be used for any other purpose like access control or additional export information.
version	Snapshot	The identifier of the specified snapshot - A data space is required for this service.
xpath	Data set table to export (XPath)	The value must be a valid absolute location path of a table in the selected data set. The notation must conform to a simplified XPath, in its abbreviated syntax - This field is required for this service.

34.9 Export data to an XML file

The exportToXML service is considered complete when export is done and file downloaded.

Service name parameter: service=@exportToXML

Input parameters

Parameter	Label	Description
branch	Data space	The identifier of the specified data space - A data space is required for this service.
instance	Data set	The value must be the reference of a data set that exists in the selected data space - This field is required for this service.
scope	Scope	Defines the scope of the user navigation for this service, namely, the entities that the user is able to select during their session. If left blank, the default value will be used. For perspectives, the default value is always 'node'. For workflows, the default value depends on the selected entities or service.
trackingInfo	Tracking information	Tracking information is logged into 'history' logs. It may also be used for any other purpose like access control or additional export information.
version	Snapshot	The identifier of the specified snapshot - A data space is required for this service.
xpath	Data set table to export (XPath)	The value must be a valid absolute location path of a table in the selected data set. The notation must conform to a simplified XPath, in its abbreviated syntax - This field is required for this service.

34.10 Import data from a CSV file

Workflows consider the importFromCSV service as complete when import is performed.

Service name parameter: service=@importFromCSV

Input parameters

Parameter	Label	Description
branch	Data space	The identifier of the specified data space - A data space is required for this service.
instance	Data set	The value must be the reference of a data set that exists in the selected data space - This field is required for this service.
scope	Scope	Defines the scope of the user navigation for this service, namely, the entities that the user is able to select during their session. If left blank, the default value will be used. For perspectives, the default value is always 'node'. For workflows, the default value depends on the selected entities or service.
trackingInfo	Tracking information	Tracking information is logged into 'history' logs. It may also be used for any other purpose like access control or additional export information.
xpath	Data set table to import (XPath)	The value must be a valid absolute location path of a table in the selected data set. The notation must conform to a simplified XPath, in its abbreviated syntax - This field is required for this service.

34.11 Import data from an XML file

Workflows consider the importFromXML service as complete when import is performed.

Service name parameter: `service=@importFromXML`

Input parameters

Parameter	Label	Description
branch	Data space	The identifier of the specified data space - A data space is required for this service.
instance	Data set	The value must be the reference of a data set that exists in the selected data space - This field is required for this service.
scope	Scope	Defines the scope of the user navigation for this service, namely, the entities that the user is able to select during their session. If left blank, the default value will be used. For perspectives, the default value is always 'node'. For workflows, the default value depends on the selected entities or service.
trackingInfo	Tracking information	Tracking information is logged into 'history' logs. It may also be used for any other purpose like access control or additional export information.
xpath	Data set table to import (XPath)	The value must be a valid absolute location path of a table in the selected data set. The notation must conform to a simplified XPath, in its abbreviated syntax - This field is required for this service.

34.12 Merge a data space

Workflows consider the merge service as complete when merger is performed and data space is closed.

Service name parameter: `service=@merge`

Input parameters

Parameter	Label	Description
branch	Data space	The identifier of the specified data space - A data space is required for this service.
scope	Scope	Defines the scope of the user navigation for this service, namely, the entities that the user is able to select during their session. If left blank, the default value will be used. For perspectives, the default value is always 'node'. For workflows, the default value depends on the selected entities or service.
trackingInfo	Tracking information	Tracking information is logged into 'history' logs. It may also be used for any other purpose like access control or additional export information.

Output parameters

Parameter	Label	Description
mergeResult	Merge success	Contains 'true' if merge succeeded, otherwise 'false'.
mergeState	Merge state	Contains the return code of the merge. It is strongly recommended to parse this value by using the InteractionMergeState UIHttpManagerComponentReturnCode.

34.13 Validate a data space, a snapshot or a data set

Workflows automatically consider the validation service as complete.

Service name parameter: `service=@validation`

Input parameters

Parameter	Label	Description
branch	Data space	The identifier of the specified data space - A data space or snapshot is required for this service.
instance	Data set	The value must be the reference of a data set that exists in the selected data space.
scope	Scope	Defines the scope of the user navigation for this service, namely, the entities that the user is able to select during their session. If left blank, the default value will be used. For perspectives, the default value is always 'node'. For workflows, the default value depends on the selected entities or service.
trackingInfo	Tracking information	Tracking information is logged into 'history' logs. It may also be used for any other purpose like access control or additional export information.
version	Snapshot	The identifier of the specified snapshot - A data space or snapshot is required for this service.

Output parameters

Parameter	Label	Description
hasError	Found errors	Contains 'true' if validation has produced errors.
hasFatal	Found fatal errors	Contains 'true' if validation has produced fatal errors.
hasInfo	Found informations	Contains 'true' if validation has produced informations.
hasWarning	Found warnings	Contains 'true' if validation has produced warnings.

CHAPITRE 35

Introduction

Ce chapitre contient les sections suivantes :

1. [Overview of data services](#)
2. [SOAP interactions](#)
3. [Data services security](#)
4. [Known limitations](#)

35.1 Overview of data services

Data services allow external systems to interact with the data governed in the EBX5 repository using the SOAP and Web Services Description Language (WSDL) standards.

A number of WSDLs can be dynamically generated from data models, then used to perform operations such as:

- Selecting, inserting, updating, deleting, or counting records
- Getting the differences on a table between data spaces or snapshots, or between two data sets based on the same data model
- Getting the credentials of records

Other generic operations for:

- Creating, merging, or closing a data space
- Creating or closing a snapshot
- Validating a data set, data space, or a snapshot
- Starting, resuming or ending a data workflow

35.2 SOAP interactions

Input and output message encoding

All input messages must be *exclusively* in UTF-8. All output messages are in UTF-8.

Tracking information

Depending on the data services operation being called, it may be possible to specify session tracking information in an optional SOAP header. For example:

```
<SOAP-ENV:Header>
  <!-- optional security header here -->
  <m:session xmlns:m="urn:ebx-schemas:dataservices_1.0">
    <trackingInformation>String</trackingInformation>
  </m:session>
</SOAP-ENV:Header>
```

Exceptions handling

Exceptions are re-thrown to the consumer through the soap:fault element within a standard exception. For example:

```
<soapenv:Fault>
  <faultcode>soapenv:java.lang.IllegalArgumentException</faultcode>
  <faultstring />
  <faultactor>admin</faultactor>
  <detail>
    <m:StandardException xmlns:m="urn:ebx-schemas:dataservices_1.0">
      <code>java.lang.IllegalArgumentException</code>
      <label/>
      <description>java.lang.IllegalArgumentException:
        Parent home not found at
        com.orchestranetworks.XX.YY.ZZ.AA.BB(AA.java:44) at
        com.orchestranetworks.XX.YY.ZZ.CC.DD(CC.java:40) ...
      </description>
    </m:StandardException>
  </detail>
</soapenv:Fault>
```

35.3 Data services security

Authentication

Authentication is mandatory. Several authentication methods are available and described below. The descriptions are ordered by priority (EBX5 applies the highest priority authentication method first).

- 'Basic Authentication Scheme' method based on the HTTP-Header Authorization in base 64, as described in [RFC 2324](#).

If the user agent wishes to send the userid "Alibaba" and password "open sesame", it will use the following header field: Authorization: Basic QWxpyYmFiYTpvcGVuIHNLc2FtZQ==

- A simple authentication based on the specification [Web Services Security UsernameToken Profile 1.0](#).

Only the mode PasswordText is supported. This is done with the following SOAP header defined in the WSDL:

```
<SOAP-ENV:Header>
  <wsse:Security xmlns:wsse="http://schemas.xmlsoap.org/ws/2002/04/secext">
    <wsse:UsernameToken>
      <wsse:Username>String</wsse:Username>
      <wsse:Password Type="wsse:PasswordText">String</wsse:Password>
    </wsse:UsernameToken>
  </wsse:Security>
</SOAP-ENV:Header>
```

Note

Only available for [Operations](#) [p 183] from HTTP request.

- Standard authentication based on HTTP Request. User and password are extracted from request parameters. For more information, see [Request parameters](#) [p 177].

Note

Only available for [WSDL generation](#) [p 175].

Overriding the SOAP security header

It is possible to override the default WSS header in order to define another security authentication mechanism. Such an override is taken into account for both HTTP and JMS. To define and override,

use the 'SOAP Header Security declaration' configuration settings under Administration > Lineage, which includes the following fields:

Schema location	The URI of the Security XML Schema to import into the WSDL.
Target namespace	The target namespace of elements in the schema.
Namespace prefix	The prefix for the target namespace.
Message name	The message name to use in the WSDL.
Root element name	The root element name of the security header. The name must be the same as the one declared in the schema.
wSDL:part element name	The name of the wSDL:part of the message.

The purpose of overriding the default security header is to change the declaration of the WSDL message matching the security header so that it contains the following:

```
<wSDL:definitions ... xmlns:MyPrefix="MyTargetNameSpace" ...
...
<xs:schema ...>
  <xs:import namespace="MyTargetNameSpace" schemaLocation="MySchemaURI"/>
  ...
</xs:schema>
...
<wSDL:message name="MySecurityMessage">
  <wSDL:part name="MyPartElementName" element="MyPrefix:MySecurityRootElement"/>
</wSDL:message>
...
<wSDL:operation name="...">
  <soap:operation soapAction="..." style="document"/>
  <wSDL:input>
    <soap:body use="literal"/>
    <soap:header message="impl:MySecurityMessage" part="MyPartElementName" use="literal"/>
  ...
</wSDL:operation>
</wSDL:definitions>
```

The corresponding XML Schema header declaration would be as follows:

```
<schema xmlns="http://www.w3.org/2001/XMLSchema" targetNamespace="MyNameSpace"
  xmlns:MyPrefix="MyNameSpace">
  <element name="MySecurityRootElement" type="MyPrefix:SpecificSecurity"/>
  <complexType name="SpecificSecurity">
    <sequence>
      <element name="AuthToken" type="string"/>
    </sequence>
  </complexType>
</schema>
```

A SOAP message using the XML schema and configuration above would have the following header:

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <SOAP-ENV:Header>
    <m:MySecurityRootElement xmlns:m="MyNameSpace">
      <AuthToken>String</AuthToken>
    </m:MySecurityRootElement>
  ...
</SOAP-ENV:Header>
<SOAP-ENV:Body>
  ...
</SOAP-ENV:Body>
```

</SOAP-ENV:Envelope>

NoteOnly available for [Operations](#) [p 183].***Lookup mechanism***

Using HTTP, the web service connector attempts authentication in the following order:

1. Using an HTTP Request
2. Using the HTTP Header Authorization
3. Looking for a security header (WSS or custom)

When using JMS, the authentication process only looks for a security header (WSS or custom).

35.4 Known limitations

Naming convention

Due to the naming convention of the data service operations, each table defined within a data model must have a unique name for WSDL generation.

Date, time & dateTime format

Data services only support the following date and time formats:

Type	Format	Example
xs:date	yyyy-MM-dd	2007-12-31
xs:time	HH:mm:ss or HH:mm:ss.SSS	11:55:00
xs:dateTime	yyyy-MM-ddTHH:mm:ss or yyyy-MM-ddTHH:mm:ss.SSS	2007-12-31T11:55:00

CHAPITRE 36

WSDL generation

Ce chapitre contient les sections suivantes :

1. [Supported standard](#)
2. [Operation types](#)
3. [Supported access methods](#)
4. [WSDL download from the data services user interfaces](#)
5. [WSDL download using a HTTP request](#)

36.1 Supported standard

EBX5 generates a WSDL that complies with the [W3C Web Services Description Language 1.1](#) standard.

36.2 Operation types

A WSDL can be generated for different types of operations:

Operation type	WSDL description
custom	WSDL for EBX5 add-ons.
dataset	WSDL for data set and replication operations.
directory	WSDL for default EBX5 directory operations. It is also possible to filter data using the tablePaths [p 178] or operations [p 178] parameters.
repository	WSDL for data space or snapshot management operations.
tables	WSDL for operations on the tables of a specific data set.
userInterface	WSDL for user interface management operations (these operations can only be accessed by administrators).
workflow	WSDL for EBX5 workflow management operations.

36.3 Supported access methods

EBX5 supports the following downloading methods:

- From the data services user interface
- From an HTTP request

Global access permissions can be independently defined for each method. For more information see .

A WSDL can only be downloaded by authorized profiles:

Operation type	Access right permissions
custom	All profiles, if at least one web service is registered.
dataset	All profiles.
directory	All profiles, if the following conditions are valid: <ul style="list-style-type: none"> • No specific directory implementation is used. (The built-in Administrator role is only subject to this condition). • Global access permissions are defined for the administration. • 'Directory' data set permissions have writing access for the current profile.
repository	All profiles.
tables	All profiles.
userInterface	Built-in Administrator role or delegated administrator profiles, if all conditions are valid: <ul style="list-style-type: none"> • Global access permissions are defined for the administration. • 'User interface' data set permissions have writing access for the current profile.
workflow	All profiles.

36.4 WSDL download from the data services user interfaces

An authorized user can download an EBX5 WSDL from the data services administration area.

Note

See [generating data service WSDLs](#) [p 149] in the user guide for more information.

36.5 WSDL download using a HTTP request

An application can download an EBX5 WSDL using an HTTP GET or POST request. The application has to be authenticated using a profile with appropriate rights.

Request details

- HTTP(S) URL format:

`http[s]://<host>[:<port>]/ebx-dataservices/<pathInfo>?<key - values>`

Both <pathInfo> and <key - values> are mandatory. For more information on possible values see: [Request parameters](#) [p 177]. An error is returned for incorrect parameters.

- HTTP(S) response status codes:

Status code	Information
200	The WSDL content was successfully generated and is returned by the request (optionally in an attachment [p 179]).
400	Bad request: request argument is incorrect.
401	Unauthorized: request requires an authenticated user.
403	Forbidden: request is not allowed for the authenticated user.
405	Method not allowed: request is not allowed in this configuration.
500	An internal error occurred: request generates an error (a stack trace and a detailed error message are returned).

Note

A detailed error message is returned for the HTTP response with status code 4xx.

- HTTP(S) parameters size restrictions:

Request type	Information
GET	2048 octets or more (HTTP Protocol Parameters). <i>Note: Servers ought to be cautious about depending on URI lengths above 255 bytes, because some older client or proxy implementations might not properly support these lengths.</i>
POST	2 mega octets or more (depending on the servlet/JSP container). Each key - value parameter is limited to 1024 characters.

Request parameters

A request parameter can be specified by one of the following methods:

- a path info on the URL (recommended)
- key values in a standard HTTP parameter.

For more detail, refer to the following table (some parameters do not have a path info representation):

Parameter name	As path info	As key - values	Required	Description
WSDL	no	yes	yes	Used to indicate the WSDL download. Empty value.
login	no	yes	no	A user identifier. Required when the standard authentication method is used. String type value.
password	no	yes	no	A password. Required when the standard authentication method is used. String type value.
type	yes	no	yes	An operation type [p 175]. Possible values are: custom, dataset, directory, userInterface, repository, tables or workflow. String type value.
branch version	yes	yes	(*)	A data space or a snapshot identifier. (*) required for tables and dataset types, otherwise ignored. String type value.
instance	yes	yes	(*)	A data set identifier. String type value.
tablePaths	no	yes	no	A list of table paths. Optional for tables or directory types, otherwise ignored. If not defined, all tables are selected. Each table path is separated by a comma character. String type value.
operations	no	yes	no	Allows generating a WSDL for a subset of operations. Optional for tables or directory operation types, otherwise ignored. If not defined, all operations for the given type are generated. This parameter's value is a concatenation of one or more of the following characters: <ul style="list-style-type: none"> • C = Count record(s) • D = Delete record(s) • E = Get credentials • G = Get changes

Parameter name	As path info	As key - values	Required	Description
				<ul style="list-style-type: none"> • I = Insert record(s) • U = Update record(s) • R = Read operations (equivalent to CEGS) • S = Select record(s) • W = Write operations (equivalent to DIU) String type value.
namespaceURI	yes	yes	(**)	<p>Unique name space URI of the custom web service.</p> <p>(**)Is required when type parameter is set to custom types. Otherwise is ignored.</p> <p>URI type value.</p>
attachmentFilename	no	yes	(***)	<p>The attachment file name.</p> <p>(***) optional if isContentInAttachment parameter is defined and set to true. Otherwise is ignored.</p> <p>String type value.</p>
isContentInAttachment	no	yes	no	<p>If value is true, the WSDL is downloaded as an attachment.</p> <p>Boolean type value.</p> <p>Default value is false.</p>

Request examples

Some of the following examples are displayed in two formats: *path info* and *key - values*.

- The WSDL will contain all repository operations.

```
http[s]://<host>[:<port>]/ebx-dataservices/repository?
WSDL&login=<login>&password=<password>
```

- The WSDL will contain all workflow operations.

```
http[s]://<host>[:<port>]/ebx-dataservices/workflow?
WSDL&login=<login>&password=<password>
```

- The WSDL will contain all tables operations for the data set 'dataset1' in data space 'dataspace1'.

Path info

```
http[s]://<host>[:<port>]/ebx-dataservices/tables/dataspace1/dataset1?
WSDL&login=<login>&password=<password>
```

Key - values

```
http[s]://<host>[:<port>]/ebx-dataservices/tables?
WSDL&login=<login>&password=<password>&
branch=<dataspace1>&instance=<dataset1>
```

- The WSDL will contain all tables with only readable operations for the data set 'dataset1' in data space 'dataspace1'.

Path info

```
http[s]://<host>[:<port>]/ebx-dataservices/tables/dataspace1/dataset1?
WSDL&login=<login>&password=<password>&operations=R
```

Key - values

```
http[s]://<host>[:<port>]/ebx-dataservices/tables?
WSDL&login=<login>&password=<password>&
branch=dataspace1&instance=dataset1&operations=R
```

- The WSDL will contain two selected tables operations for the data set 'dataset1' in data space 'dataspace1'.

Path info

```
http[s]://<host>[:<port>]/ebx-dataservices/tables/dataspace1/dataset1?
WSDL&login=<login>&password=<password>&tablePaths=/root/table1,/root/
table2
```

Key - values

```
http[s]://<host>[:<port>]/ebx-dataservices/tables?
WSDL&login=<login>&password=<password>&
branch=dataspace1&instance=dataset1&tablePaths=/root/table1,/root/table2
```

- The WSDL will contain custom web service operations for the dedicated URI.

Path info

```
http[s]://<host>[:<port>]/ebx-dataservices/custom/urn:ebx-
test:com.orchestranetworks.dataservices.WSDemo?
WSDL&login=<login>&password=<password>
```

Key - values

```
http[s]://<host>[:<port>]/ebx-dataservices/custom?  
WSDL&login=<login>&password=<password>&namespaceURI=urn:ebx-  
test:com.orchestranetworks.dataservices.WSDemo
```


CHAPITRE 37

Operations

Ce chapitre contient les sections suivantes :

1. [Operations generated from a data model](#)
2. [Operations on data sets and data spaces](#)
3. [Operations on data workflows](#)

37.1 Operations generated from a data model

For a data model used in an EBX5 repository, it is possible to dynamically generate a corresponding WSDL defining operations used to access the data sets that are an instance of this data model. For example, for a table located at the path `/root/XX/exampleTable`, the generated requests would follow the structure of its underlying data model and include the name of the table `<m:{operation}_exampleTable xmlns:m="urn:ebx-schemas:dataservices_1.0">`.

Attention

Since the WSDL and the SOAP operations tightly depend on the data model structure, it is important to redistribute the up-to-date WSDL after any data model change.

Content policy

Access to the content of records, the presence or absence of XML elements, depend on the of the authenticated user session. Additional aspects, detailed below, can impact the content.

Disabling fields from data model

The `hiddenInDataServices` property, defined in the data model, allows always hiding fields in data services, regardless of the user profile. This parameter has an impact on the generated WSDL: any hidden field or group will be absent from the request and response structure.

Modifying the `hiddenInDataServices` parameter value has the following impact on a client which would still use the former WSDL:

- On request, if the data model property has been changed to `true`, and if the concerned field is present in the WSDL request, an exception will be thrown.
- On response, if the schema property has been changed to `false`, WSDL validation will return an error if it is activated.

This setting of "Default view" is defined inside data model.

Association field

Read-access on table records can export the association fields as displayed in UI Manager. This feature can be coupled with the 'hiddenInDataServices' model parameter.

Note

Limitations: change and update operations do not manage association fields. Also, the select operation only exports the first level of association elements (the content of associated objects cannot contain association elements).

Common request parameters

Several parameters are common to several operations and are detailed below.

Element	Description	Required
branch	The identifier of the data space to which the data set belongs.	Either this parameter or the 'version' parameter must be defined. Required for the 'insert', 'update' and 'delete' operations.
version	The identifier of the snapshot to which the data set belongs.	Either this parameter or the 'branch' parameter must be defined
instance	The unique name of the data set which contains the table to query.	Yes
predicate	XPath predicate [p 221] defines the records on which the request is applied. If empty, all records will be retrieved, except for the 'delete' operation where this field is mandatory.	Only required for the 'delete' operation
data	Contains the records to be modified, represented within the structure of their data model. The whole operation is equivalent to an XML import. The details of the operations performed on data content are specified in the section Import [p 209].	Only required for the insert and update operations
viewPublication	This parameter can be combined with the predicate [p 185] parameter as a logical AND operation. The behavior of this parameter is described in the section . It cannot be used if the 'viewId' parameter is used, and cannot be used on hierarchical views.	No
viewId	<i>Deprecated since version 5.2.3.</i> This parameter has been replaced by the parameter 'viewPublication'. While it remains available for backward compatibility, it will eventually be removed in a future version. This parameter cannot be used if the 'viewPublication' parameter is used.	No
blockingConstraintsDisabled	This property is available for all table updates data service operations. If true, the validation process disables blocking constraints defined in the data model. If this parameter is not present, the default is false.	No

Select operation

Select request

```
<m:select_{TableName} xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <branch>String</branch>
```

```

<version>String</version>
<instance>String</instance>
<predicate>String</predicate>
<viewPublication>String</viewPublication>
<exportCredentials>boolean</exportCredentials>
<pagination>
  <previousPageLastRecordPredicate>String</previousPageLastRecordPredicate>
  <pageSize>Integer</pageSize>
</pagination>
</m:select_{TableName}>

```

with:

Element	Description	Required
branch	See the description under Common parameters [p 185].	
version	See the description under Common parameters [p 185].	
instance	See the description under Common parameters [p 185].	
predicate	See the description under Common parameters [p 185]. This parameter can be combined with the viewPublication [p 185] parameter as a logical AND operation.	
viewPublication	See the description under Common parameters [p 185].	
includesTechnicalData	The response will contain technical data if true. See also the optimistic locking [p 198] section. Each returned record will contain additional attributes for this technical information, for instance: <pre> ... <table ebxd:lastTime="2010-06-28T10:10:31.046" ebxd:lastUser="Uadmin" ebxd:uuid="9E7D0530-828C-11DF-B733-0012D01B6E76">... </pre>	No
exportCredentials	If true the select will also return the credentials for each record.	No
pagination	Enables pagination, see child elements below.	No
pageSize (nested under the pagination element)	When pagination is enabled, defines the number of records to retrieve.	When pagination is enabled, yes
previousPageLastRecordPredicate (nested under the pagination element)	When pagination is enabled, XPath predicate that defines the record after which the page must fetched, this value is provided by the previous response, as the element lastRecordPredicate. If the passed record is not found, the first page will be returned.	No

Select response

```

<ns1:select_{TableName}Response xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <data>
    <XX>
      <TableName>
        <a>key1</a>
        <b>valueb</b>
        <c>1</c>
        <d>1</d>
      </TableName>
    </XX>
  </data>
</ns1:select_{TableName}Response>

```

```

</XX>
</data>
<credentials>
  <XX>
    <TableName predicate="/a='key1'">
      <a>W</a>
      <b>W</b>
      <c>W</c>
      <d>W</d>
    </TableName>
  </XX>
</credentials>
<lastRecordPredicate>/a='key1'</lastRecordPredicate>
</ns1:select_{TableName}Response>

```

See also the [optimistic locking](#) [p 198] section.

Delete operation

Delete request

```

<m:delete_{TableName} xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <branch>String</branch>
  <version>String</version>
  <instance>String</instance>
  <predicate>String</predicate>
  <occultIfInherit>boolean</occultIfInherit>
  <checkNotChangedSinceLastTime>dateTime</checkNotChangedSinceLastTime>
  <blockingConstraintsDisabled>boolean</blockingConstraintsDisabled>
</m:delete_{TableName}>

```

with:

Element	Description	Required
branch	See the description under Common parameters [p 185].	
instance	See the description under Common parameters [p 185].	
predicate	See the description under Common parameters [p 185].	
occultIfInherit	Occults the record if it is in inherit mode. Default value is false.	No
checkNotChangedSinceLastTime	Timestamp used to ensure that the record has not been modified since the last read. Also see the optimistic locking [p 198] section.	No
blockingConstraintsDisabled	See the description under Common parameters [p 185].	

Delete response

```

<ns1:delete_{TableName}Response xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <status>String</status>
</ns1:delete_{TableName}Response>

```

with:

Element	Description
status	'00' indicates that the operation has been executed successfully. '95' indicates that at least one operation has violated a blocking constraint, resulting in the overall operation being aborted.

Count operation

Count request

```
<m:count_{TableName} xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <branch>String</branch>
  <version>String</version>
  <instance>String</instance>
  <predicate>String</predicate>
</m:count_{TableName}>
```

with:

Element	Description
branch	See the description under Common parameters [p 185].
version	See the description under Common parameters [p 185].
instance	See the description under Common parameters [p 185].
predicate	See the description under Common parameters [p 185].

Count response

```
<ns1:count_{TableName}Response xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <count>Integer</count>
</ns1:count_{TableName}Response>
```

with:

Element	Description
count	The number of records that correspond to the predicate in the request.

Update operation

Update request

```
<m:update_{TableName} xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <branch>String</branch>
  <version>String</version>
  <instance>String</instance>
  <updateOrInsert>boolean</updateOrInsert>
  <byDelta>boolean</byDelta>
  <blockingConstraintsDisabled>boolean</blockingConstraintsDisabled>
  <data>
    <XX>
      <TableName>
        <a>String</a>
        <b>String</b>
        <c>String</c>
        <d>String</d>
        ...
      </TableName>
    </XX>
  </data>
</m:update_{TableName}>
```

with:

Element	Description	Required
branch	See the description under Common parameters [p 185].	
instance	See the description under Common parameters [p 185].	
updateOrInsert	If true and the record does not currently exist, the operation creates the record.	No
byDelta	If true and an element does not currently exist in the incoming message, the target value is not changed. If false and node is declared hiddenInDataServices, the target value is not changed. The complete behavior is described in the sections Insert and update operations [p 210].	No
blockingConstraintsDisabled	See the description under Common parameters [p 185].	
data	See the description under Common parameters [p 185].	

Update response

```
<ns1:update_{TableName}Response xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <status>String</status>
</ns1:update_{TableName}Response>
```

with:

Element	Description
status	'00' indicates that the operation has been executed successfully. '95' indicates that at least one operation has violated a blocking constraint, resulting in the overall operation being aborted.

Insert operation

Insert request

```
<m:insert_{TableName} xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <branch>String</branch>
  <version>String</version>
  <instance>String</instance>
  <byDelta>boolean</byDelta>
  <blockingConstraintsDisabled>boolean</blockingConstraintsDisabled>
  <data>
    <XX>
      <TableName>
        <a>String</a>
        <b>String</b>
        <c>String</c>
        <d>String</d>
        ...
      </TableName>
    </XX>
  </data>
</m:insert_{TableName}>
```

with:

Element	Description	Required
branch	See the description under Common parameters [p 185].	
instance	See the description under Common parameters [p 185].	
byDelta	If true and an element does not currently exist in the incoming message, the target value is not changed. If false and node is declared hiddenInDataServices, the target value is not changed. The complete behavior is described in the sections Insert and update operations [p 210].	No
blockingConstraintsDisabled	See the description under Common parameters [p 185].	
data	See the description under Common parameters [p 185].	

Insert response

```
<ns1:insert_{TableName}Response xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <status>String</status>
  <inserted>
    <predicate>./a='String'</predicate>
  </inserted>
</ns1:insert_{TableName}Response>
```

with:

Element	Description
status	'00' indicates that the operation has been executed successfully. '95' indicates that at least one operation has violated a blocking constraint, resulting in the overall operation being aborted.
predicate	A predicate matching the primary key of the inserted record. When several records are inserted, the predicates follow the declaration order of the records in the input message.

Get changes operations

Get changes requests

Changes between two data sets:

```
<m:getChangesOnDataSet_{schemaName} xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <branch>String</branch>
  <version>String</version>
  <instance>String</instance>
  <compareWithBranch>String</compareWithBranch>
  <compareWithVersion>String</compareWithVersion>
  <compareWithInstance>String</compareWithInstance>
  <resolvedMode>boolean</resolvedMode>
</m:getChangesOnDataSet_{schemaName}>
```

Changes between two tables:

```
<m:getChanges_{TableName} xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <branch>String</branch>
  <version>String</version>
  <instance>String</instance>
  <compareWithBranch>String</compareWithBranch>
  <compareWithVersion>String</compareWithVersion>
  <resolvedMode>boolean</resolvedMode>
</m:getChanges_{TableName}>
```

with:

Element	Description	Required
branch	See the description under Common parameters [p 185].	
version	See the description under Common parameters [p 185].	
instance	See the description under Common parameters [p 185].	
compareWithBranch	The identifier of the data space with which to compare.	One of either this parameter or the ' compareWithVersion [p 192]' parameter must be defined.
compareWithVersion	The identifier of the snapshot with which to compare.	One of either this parameter or the ' compareWithBranch [p 192]' parameter must be defined.
compareWithInstance	The identifier of the data set with which to compare. If it is undefined, instance [p 192] parameter is used.	No
resolvedMode	Defines whether or not the difference is calculated in resolved mode. Default is true.	No
pagination	<p>Enables pagination context for the operations <code>getChanges</code> and <code>getChangesOnDataSet</code>.</p> <p>Allows client to define pagination context size. Each page contains a collection of inserted, updated and/or deleted records of tables according to the maximum size.</p> <p>Get changes persisted context is built at first call according to the page size parameter in request.</p> <p>For creation: Defines <code>pageSize</code> parameter.</p> <p>For next: Defines context element with <code>identifier</code> from previous response.</p> <p>Enables pagination, see child elements below.</p>	No
pageSize (nested under pagination element)	Defines maximum number of records in each page. Minimal size is 50.	No (Only for creation)
context (nested under pagination element)	Defines content of pagination context.	No (Only for next)
identifier (nested under context element)	Pagination context identifier.	Yes

Note

If neither *compareWithBranch* nor *compareWithVersion* are specified, the comparison will be made with its parent:

- if the current data space or snapshot is a data space, the comparison is made with its initial snapshot (includes all changes made in the data space);
- if the current data space or snapshot is a snapshot, the comparison is made with its parent data space (includes all changes made in the parent data space since the current snapshot was created);
- returns an exception if the current data space is the 'Reference' data space.

Get changes responses

Changes between two data sets:

```
<ns1:getChangesOnDataSet_{schemaName}Response xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <getChanges_{TableName1}>
    ... see the getChanges between tables response example ...
  </getChanges_{TableName1}>
  <getChanges_{TableName2}>
    ... see the getChanges between tables response example ...
  </getChanges_{TableName2}>
  ...
</ns1:getChangesOnDataSet_{schemaName}Response>
```

Changes between two tables:

```
<ns1:getChanges_{TableName}Response xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <inserted>
    <XX>
      <TableName>
        <a>AVALUE3</a>
        <b>BVALUE3</b>
        <c>CVALUE3</c>
        <d>DVALUE3</d>
      </TableName>
    </XX>
  </inserted>
  <updated>
    <changes>
      <change predicate="/a='AVALUE2'">
        <path>/b</path>
        <path>/c</path>
      </change>
    </changes>
  <data>
    <XX>
      <TableName>
        <a>AVALUE2</a>
        <b>BVALUE2.1</b>
        <c>CVALUE2.1</c>
        <d>DVALUE2</d>
      </TableName>
    </XX>
  </data>
</updated>
  <deleted>
    <predicate>./a='AVALUE1'</predicate>
  </deleted>
</ns1:getChanges_{TableName}Response>
```

with:

Element	Description	Required
inserted	Contains inserted record(s) from choice <code>compareWithBranch</code> or <code>compareWithVersion</code> . Content under this element corresponding to an XML export of inserted records.	No
updated	Contains updated record(s).	No
changes (nested under context updated)	Only the group of field have been updated.	Yes
change (nested under context changes)	Group of fields have been updated with own XPath predicate attribute of the record.	Yes
data (nested under context changes)	Content under this element corresponding to an XML export of updated records under changes element.	No
deleted	Records have been deleted from context of request. Content corresponding to a list of predicate element who contains the XPath predicate of record.	No
pagination	When pagination is enabled on request. Get changes persisted context allows invoking the next page until last page or when a timeout is reached. For next: Defines context element with <code>identifier</code> . For last: Defines context element without <code>identifier</code> . Enables pagination, see child elements below.	No
context (nested under pagination element)	Defines content of pagination context.	Yes (Only for next and last)
identifier (nested under context element)	Pagination context identifier. Not defined at last returned page.	No
pageNumber (nested under context element)	Current page number in pagination context.	Yes
totalPages (nested under context element)	Total pages in pagination context.	Yes

Get changes operation with pagination enabled

Only pagination element and sub elements have been described.

For creation:

Extract of request:

```
...
<pagination>
  <!-- on first request for creation -->
  <pageSize>Integer</pageSize>
```

```
</pagination>
...
```

Extract of response:

```
...
<pagination>
  <!-- on next request to continue -->
  <context>
    <identifier>String</identifier>
    <pageNumber>Integer</pageNumber>
    <totalPages>Integer</totalPages>
  </context>
</pagination>
...
```

For next:

Extract of request:

```
...
<pagination>
  <context>
    <identifier>String</identifier>
  </context>
</pagination>
...
```

Extract of response:

```
...
<pagination>
  <!-- on next request to continue -->
  <context>
    <identifier>String</identifier>
    <pageNumber>Integer</pageNumber>
    <totalPages>Integer</totalPages>
  </context>
</pagination>
...
```

For last:

Extract of request:

```
...
<pagination>
  <context>
    <identifier>String</identifier>
  </context>
</pagination>
...
```

Extract of response:

```
...
<pagination>
  <context>
    <pageNumber>Integer</pageNumber>
    <totalPages>Integer</totalPages>
  </context>
</pagination>
...
```

Get credentials operation

Get credentials request

```
<m:getCredentials_{TableName} xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <branch>String</branch>
  <version>String</version>
  <instance>String</instance>
  <predicate>String</predicate>
  <viewPublication>String</viewPublication>
</m:getCredentials_{TableName}>
```

with:

Element	Description	Required
branch	See the description under Common parameters [p 185].	
version	See the description under Common parameters [p 185].	
instance	See the description under Common parameters [p 185].	
viewPublication	See the description under Common parameters [p 185].	

Get credentials response

```
<ns1:getCredentials_{TableName}Response xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <XX>
    <TableName>
      <a>R</a>
      <b>W</b>
      <c>H</c>
      <d>W</d>
      ...
    </TableName>
  </XX>
</ns1:getCredentials_{TableName}Response>
```

With the following possible values:

- R: for read-only
- W: for read-write
- H: for hidden

Multiple chained operations

Multiple operations request

It is possible to run multiple operations across tables in the data set, while ensuring a consistent response. The operations are executed sequentially, according to the order defined on the client side.

All operations are executed in a single transaction with a **SERIALIZABLE** isolation level. If all requests in the multiple operation are read-only, they are allowed to run fully concurrently along with other read-only transactions, even in the same data space.

When an error occurs during one operation in the sequence, all updates are rolled back and the client receives a **StandardException** error message with details.

```
<m:multi_ xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <branch>String</branch>
  <version>String</version>
  <instance>String</instance>
  <blockingConstraintsDisabled>boolean</blockingConstraintsDisabled>
  <request id="id1">
    <{operation}_{TableName}>
      ...
    </{operation}_{TableName}>
  </request>
  <request id="id2">
    <{operation}_{TableName}>
      ...
    </{operation}_{TableName}>
  </request>
</m:multi_>
```

with:

Element	Description	Required
branch	See the description under Common parameters [p 185].	
version	See the description under Common parameters [p 185].	
instance	See the description under Common parameters [p 185].	
blockingConstraintsDisabled	See the description under Common parameters [p 185].	
request	<p>This element contains one operation, like a single operation without branch, version and instance parameters. This element can be repeated multiple times for additional operations. Each request can be identified by an 'id' attribute. In a response, this 'id' attribute is returned for identification purposes.</p> <p>Operations such as count, select, getChanges, getCredentials, insert, delete or update.</p>	Yes

Note:

- Does not accept a limit on the number of request elements.
- The request id attribute must be unique in multi-operation requests.
- If all operations are read only (count, select, getChanges, or getCredentials) then the whole transaction is set as read-only for performance considerations.

Limitation:

- The multi operation applies to one model and one data set (parameter instance).
- The select operation cannot use the pagination parameter.

Multiple operations response

See each response operation for details.

```
<ns1:multi_Response xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <response id="id1">
    <{operation}_{TableName}Response>
      ...
    </{operation}_{TableName}Response>
  </response>
  <response id="id2">
    <{operation}_{TableName}Response>
      ...
    </{operation}_{TableName}Response>
  </response>
</ns1:multi_Response>
```

with:

Element	Description
response	<p>This element contains the response of one operation. It is repeated multiple times for additional operations. Each response is identified by an 'id' attribute set in the request or automatically generated.</p> <p>The content of the element corresponds to the response of a single operation, such as count, select, getChanges, getCredentials, insert, delete or update.</p>

Optimistic locking

To prevent an update or a delete operation from being performed on a record that was read earlier but may have changed in the meantime, an optimistic locking mechanism is provided.

In the select request, it is possible to ask for technical information by adding the element `includesTechnicalData`:

```
<m:select_{TableName} xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <branch>String</branch>
  <version>String</version>
  <instance>String</instance>
  <predicate>String</predicate>
  <includesTechnicalData>boolean</includesTechnicalData>
</m:select_{TableName}>
```

The value of the `lastTime` attribute can then be used in the update request. If the record has been changed since the specified time, the update will be cancelled. The attribute `lastTime` has to be added on the record to be updated.

```
<m:update_{TableName} xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <branch>String</branch>
  <version>String</version>
  <instance>String</instance>
  <updateOrInsert>true</updateOrInsert>
  <data>
    <XX>
      <TableName ebxd:lastTime="2010-06-28T10:10:31.046">
        <a>String</a>
        <b>String</b>
        <c>String</c>
        <d>String</d>
        ...
      </TableName>
    </XX>
  </data>
</m:update_{TableName}>
```

The value of the `lastTime` attribute can also be used to prevent deletion on a modified record:

```
<m:delete_{TableName} xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <branch>String</branch>
  <version>String</version>
  <instance>String</instance>
  <predicate>String</predicate>
  <checkNotChangedSinceLastTime>2010-06-28T10:10:31.046</checkNotChangedSinceLastTime>
</m:delete_{TableName}>
```

The element `checkNotChangedSinceLastTime` can be used for one and only one record upon request. This means that if the predicate returns more than one record, the request will fail if the element `checkNotChangedSinceLastTime` is set.

37.2 Operations on data sets and data spaces

Parameters for operations on data spaces and snapshots are as follows:

<i>Element</i>	<i>Description</i>	<i>Required</i>
branch	Identifier of the target data space on which the operation is applied. When not specified, the 'Reference' data space is used except for the merge data space operation where it is required.	One of either this parameter or the 'version' parameter must be defined. Required for the data space merge, locking, unlocking and replication refresh operations.
version	Identifier of the target snapshot on which the operation is applied.	One of either this parameter or the 'branch' parameter must be defined
versionName	Identifier of the snapshot to create. If empty, it will be defined on the server side.	No
childBranchName	Identifier of the data space child to create. If empty, it will be defined on the server side.	No
instance	The unique name of the data set on which the operation is applied.	Required for the replication refresh operation.
ensureActivation	Defines if validation must also check whether this instance is activated.	Yes
details	<p>Defines if validation returns details.</p> <p>The optional attribute <code>severityThreshold</code> defines the lowest severity level of message to return. The possible values are, in descending order of severity, 'fatal', 'error', 'warning', or 'info'. For example, setting the value to 'error' will return error and fatal validation messages. If this attribute is not defined, all levels of messages are returned by default.</p> <p>The optional attribute <code>locale</code> (default 'en-US') defines the language in which the validation messages are to be returned.</p>	No. If not specified, no details are returned.
owner	Defines the owner.	No
branchToCopyPermissionFrom	Defines the identifier of the data space from which to copy the permissions.	No
documentation	<p>Documentation for a dedicated language.</p> <p>Multiple documentation elements may be used for several languages.</p>	No

<i>Element</i>	<i>Description</i>	<i>Required</i>
locale (nested under the documentation element)	Locale of the documentation.	Only required when the documentation element is used
label (nested under the documentation element)	Label for the language.	No
description (nested under the documentation element)	Description for the language.	No

Validate a data space

Validate data space request

```
<m:validate xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <branch>String</branch>
  <version>String</version>
</m:validate>
```

Validate data space response

```
<ns1:validate_Response xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <validationReport>
    <instanceName>String</instanceName>
    <fatals>boolean</fatals>
    <errors>boolean</errors>
    <infos>boolean</infos>
    <warnings>boolean</warnings>
  </validationReport>
</ns1:validate_Response>
```

Validate a data set

Validate data set request

```
<m:validateInstance xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <branch>String</branch>
  <version>String</version>
  <instance>String</instance>
  <ensureActivation>boolean</ensureActivation>
  <details severityThreshold="fatal|error|warning|info" locale="en-US"/>
</m:validateInstance>
```

Validate data set response

```
<ns1:validateInstance_Response xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <validationReport>
    <instanceName>String</instanceName>
    <fatals>boolean</fatals>
    <errors>boolean</errors>
    <infos>boolean</infos>
    <warnings>boolean</warnings>
    <details>
      <reportItem>
        <severity>{fatal|error|warning|info}</severity>
        <message>
          <internalId />
          <text>String</text>
        </message>
        <subject>
          <table>Path</table>
          <predicate>String</predicate>
          <path>Path</path>
        </subject>
      </reportItem>
    </details>
```



```
</validationReport>
</ns1:validateInstance_Response>
```

Create a data space

Create data space request

```
<m:createBranch xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <branch>String</branch>
  <owner>String</owner>
  <branchToCopyPermissionFrom>String</branchToCopyPermissionFrom>
  <documentation>
    <locale>Locale</locale>
    <label>String</label>
    <description>String</description>
  </documentation>
  <childBranchName>String</childBranchName>
</m:createBranch>
```

Create data space response

```
<ns1:createBranch_Response xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <status>String</status>
  <childBranchName>String</childBranchName>
</ns1:createBranch_Response>
```

with:

Element	Description
status	'00' indicates that the operation has been executed successfully.

Create a snapshot

Create snapshot request

```
<m:createVersion xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <branch>String</branch>
  <versionName>String</versionName>
  <owner>String</owner>
  <documentation>
    <locale>Locale</locale>
    <label>String</label>
    <description>String</description>
  </documentation>
</m:createVersion>
```

Create snapshot response

```
<ns1:createVersion_Response xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <status>String</status>
  <versionName>String</versionName>
</ns1:createVersion_Response>
```

with:

Element	Description
status	'00' indicates that the operation has been executed successfully.

Locking a data space

Lock data space request

```
<m:lockBranch xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <branch>String</branch>
  <durationToWaitForLock>Integer</durationToWaitForLock>
  <message>
    <locale>Locale</locale>
    <label>String</label>
  </message>
</m:lockBranch>
```

with:

Element	Description	Required
durationToWaitForLock	This parameter defines the maximum duration (in seconds) that the operation waits for a lock before aborting.	No, does not wait by default
message	User message of the lock. Multiple message elements may be used.	No
locale (nested under the message element)	Locale of the user message.	Only required when the message element is used
label (nested under the message element)	The user message.	No

Lock data space response

```
<ns1:lockBranch_Response xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <status>String</status>
</ns1:lockBranch_Response>
```

with:

Element	Description
status	'00' indicates that the operation has been executed successfully. '94' indicates that the data space has been already locked by another user. Otherwise, a SOAP exception is thrown.

Unlocking a data space

Unlock data space request

```
<m:unlockBranch xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <branch>String</branch>
</m:unlockBranch>
```

Unlock data space response

```
<ns1:unlockBranch_Response xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <status>String</status>
</ns1:unlockBranch_Response>
```

with:

Element	Description
status	'00' indicates that the operation has been executed successfully. Otherwise, a SOAP exception is thrown.

Merge a data space

Merge data space request

```
<m:mergeBranch xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <branch>String</branch>
  <deleteDataOnMerge>boolean</deleteDataOnMerge>
  <deleteHistoryOnMerge>boolean</deleteHistoryOnMerge>
</m:mergeBranch>
```

with:

Element	Description	Required
deleteDataOnMerge	This parameter is available for the merge data space operation. Sets whether the specified data space and its associated snapshots will be deleted upon merge.	No
deleteHistoryOnMerge	This parameter is available for the merge data space operation. Sets whether the history associated with the specified data space will be deleted upon merge. Default value is false.	No

Note

The merge decision step is bypassed during merges performed through data services. In such cases, the data in the child data space automatically overrides the data in the parent data space.

Merge data space response

```
<ns1:mergeBranch_Response xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <status>String</status>
</ns1:mergeBranch_Response>
```

with:

Element	Description
status	'00' indicates that the operation has been executed successfully.

Close a data space or snapshot

Close data space or snapshot request

Close data space request:

```
<m:closeBranch xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <branch>String</branch>
  <deleteDataOnClose>boolean</deleteDataOnClose>
  <deleteHistoryOnClose>boolean</deleteHistoryOnClose>
</m:closeBranch>
```

Close snapshot request:

```
<m:closeVersion xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <version>String</version>
  <deleteDataOnClose>boolean</deleteDataOnClose>
</m:closeVersion>
```

with:

Element	Description	Required
deleteDataOnClose	This parameter is available for the close data space and close snapshot operations. Sets whether the specified snapshot, or data space and its associated snapshots, will be deleted upon closure.	No
deleteHistoryOnClose	This parameter is available for the close data space operation. Sets whether the history associated with the specified data space will be deleted upon closure. Default value is false.	No

Close data space or snapshot response

Close data space response:

```
<ns1:closeBranch_Response xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <status>String</status>
</ns1:closeBranch_Response>
```

Close snapshot request:

```
<ns1:closeVersion_Response xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <status>String</status>
</ns1:closeVersion_Response>
```

37.3 Operations on data workflows

Parameters for operations on data workflows are as follows:

Element	Description	Required
parameters	Input parameters for the process instance to be created.	No
parameter (nested under the parameters element). Multiple parameter elements may be used.	An input parameter for the process instance.	No
name (nested under the parameter element)	Name of the parameter.	Yes
value (nested under the parameter element)	Value of the parameter.	No

Start a workflow

Start a workflow process instance from a published process key. It is possible to start a workflow with localized documentation and specific input parameters (with name and optional value).

Sample request:

```
<m:workflowProcessInstanceStart xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <publishedProcessKey>String</publishedProcessKey>
  <documentation>
    <locale>Locale</locale>
    <label>String</label>
    <description>String</description>
  </documentation>
  <parameters>
    <parameter>
      <name>String</name>
      <value>String</value>
    </parameter>
  </parameters>
</m:workflowProcessInstanceStart>
```

with:

Element	Description	Required
publishedProcessKey	Key of the published workflow to start.	Yes
documentation	See the description under Common parameters [p 199].	No
parameters	See the description under Common parameters [p 205].	No

Sample response:

```
<m:workflowProcessInstanceStart_Response xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <processInstanceKey>String</processInstanceKey>
</m:workflowProcessInstanceStart_Response>
```

with:

Element	Description	Required
processInstanceId	<i>Deprecated since version 5.6.1</i> This parameter has been replaced by the parameter 'processInstanceKey'. While it remains available for backward compatibility, it will eventually be removed in a future major version.	No
processInstanceKey	Key of the workflow process instance.	No

Resume a workflow

Resume a workflow process instance in a wait step from a resume identifier. It is possible to defined specific input parameters (with name and optional value).

Sample request:

```
<m:workflowProcessInstanceResume xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <resumeId>String</resumeId>
  <parameters>
    <parameter>
      <name>String</name>
      <value>String</value>
    </parameter>
  </parameters>
</m:workflowProcessInstanceResume>
```

with:

Element	Description	Required
resumeId	Resume identifier of the waiting task.	Yes
parameters	See the description under Common parameters (p 205).	No

Sample response:

```
<m:workflowProcessInstanceResume_Response xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <status>String</status>
  <processInstanceKey>String</processInstanceKey>
</m:workflowProcessInstanceResume_Response>
```

with:

Element	Description	Required
status	'00' indicates that the operation has been executed successfully. '20' indicates that the process instance has not been found. '21' indicates that the event has already been received.	Yes
processInstanceKey	Key of the workflow process instance. This parameter is returned if the operation has been executed successfully.	No

End a workflow

End a workflow process instance from its key representation.

Sample request:

```
<m:workflowProcessInstanceEnd xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <processInstanceKey>String</processInstanceKey>
</m:workflowProcessInstanceEnd>
```

with:

Element	Description	Required
processInstanceKey	Key of the workflow process instance.	Either this parameter or 'publishedProcessKey' and 'processInstanceId' parameters must be defined.
publishedProcessKey	<i>Deprecated since version 5.6.1</i> Due to a limitation this parameter has been replaced by the 'processInstanceKey' parameter. While it remains available for backward compatibility, it will eventually be removed in a future major version.	No
processInstanceId	<i>Deprecated since version 5.6.1</i> Due to a limitation this parameter has been replaced by the 'processInstanceKey' parameter. While it remains available for backward compatibility, it will eventually be removed in a future major version.	No

Sample response:

```
<m:workflowProcessInstanceEnd_Response xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <status>String</status>
</m:workflowProcessInstanceEnd_Response>
```

with:

Element	Description	Required
status	'00' indicates that the operation has been executed successfully.	Yes

CHAPITRE 38

XML import and export

XML imports and exports can be performed on tables through the user interface using the **Actions** menu in the workspace.

Both imports and exports are performed in the context of a data set.

Ce chapitre contient les sections suivantes :

1. [Imports](#)
2. [Exports](#)
3. [Handling of field values](#)
4. [Known limitations](#)

38.1 Imports

Attention

Imported XML documents must be encoded in UTF-8 and its structure must conform to the underlying data model of the target data set.

Import mode

When importing an XML file, you must specify one of the following import modes, which will dictate how the import procedure handles the source records.

Insert mode	Only record creations are allowed. If a record exists in the target table with the same primary key as the source record, an error is returned and the whole import operation is cancelled.
Update mode	Only modifications of existing records are allowed. If no record exists in the target table with the same primary key as the source record, an error is returned and the whole import operation is cancelled.
Update or insert mode	If a record with the same primary key as the source record already exists in the target table, that record is updated. Otherwise, a new record is created.
Replace (synchronization) mode	If a record with the same primary key as the source record already exists in the target table, that record is updated. Otherwise, a new record is created. If a record exists in the target table but is not present in the source XML file, that record is deleted from the table.

Insert and update operations

The mode *'by delta'* allows ignoring data model elements that are missing from the source XML document. This mode can be enabled through data services or the Java API. The following table

summarizes the behavior of insert and update operations when elements are not present in the source document.

State in source XML document	Behavior
Element does not exist in the source document	<p>If 'by delta' mode is disabled (default):</p> <p>Target field value is set to one of the following:</p> <ul style="list-style-type: none"> • If the element defines a default value, the target field value is set to that default value. • If the element is of a type other than a string or list, the target field value is set to null. • If the element is an aggregated list, the target field value is set to an empty list. • If the element is a string that distinguishes null from an empty string, the target field value is set to null. If it is a string that does not distinguish between the two, an empty string. • If the element (simple or complex) is hidden in data services, the target value is not changed. <p>Note: The user performing the import must have the permissions necessary to create or change the target field value. Otherwise, the value will remain unchanged.</p> <p>If 'by delta' mode has been enabled through data services or the Java API:</p> <ul style="list-style-type: none"> • For the update operation, the field value remains unchanged. • For the insert operation, the behavior is the same as when byDelta mode is disabled.
Element exists but is empty (for example, <fieldA/>)	<ul style="list-style-type: none"> • For nodes of type xs:string (or one of its sub-types), the target field's value is set to null if it distinguishes null from an empty string. Otherwise, the value is set to empty string. • For non-xs:string type nodes, an exception is thrown in conformance with XML Schema.
Element is present and null (for example, <fieldA xsi:nil="true"/>)	<p>The target field is always set to null except for lists, for which it is not supported.</p> <p>In order to use the xsi:nil="true" attribute, you must import the namespace xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance".</p>

It may happen that the XML document contains elements that do not exist in the target data model. By default, in this case, the import procedure will fail. It is possible, however, to allow users to launch import procedures that will ignore the extra columns defined in the XML files. This can be done in the configuration parameters of the import wizard for XML. The default value of this parameter can be configured in the 'User interface' configuration under the 'Administration' area.

Optimistic locking

If the technical attribute `x:lastTime` exists in the source XML file, the import mechanism performs a verification to prevent an update operation on a record that may have changed since the last read. The timestamp associated with the current record will be compared to this timestamp. If they are different, the update is rejected.

38.2 Exports

Note

Exported XML documents are always encoded in UTF-8.

When exporting to XML, if the table has filters applied, only the records that correspond to the filter are included in the exported file.

The XML export options are as follows:

Download file name	Specifies the name of the XML file to be exported. This field is pre-populated with the name of the table from which the records are being exported.
User-friendly mode	<p>Specifies whether exported values will be represented in a user-friendly way, or in the standard XML raw format. For example, in user-friendly mode, dates and numbers are formatted according to the user's locale, and foreign keys and enumerated values display their associated labels.</p> <p>Note: If this option is selected, the exported file will not be able to be re-imported.</p>
Include technical data	<p>Specifies whether internal technical data will be included in the export.</p> <p>Note: If this option is selected, the exported file will not be able to be re-imported.</p>
Include computed values	Specifies whether computed values will be exported.
Is indented	Specifies whether the file should be indented to improve readability.

38.3 Handling of field values

Date, time & dateTime format

The following date and time formats are supported:

Type	Format	Example
xs:date	yyyy-MM-dd	2007-12-31
xs:time	HH:mm:ss or HH:mm:ss.SSS	11:55:00
xs:dateTime	yyyy-MM-ddTHH:mm:ss or yyyy-MM-ddTHH:mm:ss.SSS	2007-12-31T11:55:00

38.4 Known limitations

Association fields

The XML import and export services do not support association values.

Exporting such nodes will not cause any error, however, no value will be exported.

Importing such nodes will cause an error, and the import procedure will be aborted.

CHAPITRE 39

CSV import and export

CSV imports and exports can be performed on tables through the user interface using the **Actions** menu in the workspace. Both imports and exports are performed in the context of a data set.

Ce chapitre contient les sections suivantes :

1. [Exports](#)
2. [Imports](#)
3. [Handling of field values](#)
4. [Known limitations](#)

39.1 Exports

When exporting to CSV, if the table has filters applied, only the records that correspond to the filter are included in the exported file.

The CSV export options are as follows:

Download file name	Specifies the name of the CSV file to be exported. This field is pre-populated with the name of the table from which the records are being exported.
File encoding	Specifies the character encoding to use for the exported file. The default is UTF-8.
Enable inheritance	<p>Specifies if inheritance will be taken into account during a CSV export. If inheritance is enabled, resolved values of fields are exported with the technical data that define the possible inheritance mode of the record or the field. If inheritance is disabled, resolved values of fields are exported and occluded records are ignored. By default, this option is disabled.</p> <p>Note:Inheritance is always ignored, if the table data set has no parent or if the table has no inherited field.</p>
User-friendly mode	<p>Specifies whether exported values will be represented in a user-friendly way, or in a raw format. For example, in user-friendly mode, dates and numbers are formatted according to the user's locale, and foreign keys and enumerated values display their associated labels.</p> <p>Note: If this option is selected, the exported file will not be able to be re-imported.</p>
Include technical data	<p>Specifies whether internal technical data will be included in the export.</p> <p>Note: If this option is selected, the exported file will not be able to be re-imported.</p>
Include computed values	Specifies whether computed values will be exported.
Column header	<p>Specifies the whether or not to include column headers in the CSV file.</p> <ul style="list-style-type: none"> • No header • Label: For each column in the spreadsheet, the CSV displays its label. Each label is localized according to the locale preference of the current session. If no user-friendly label is defined for a node, the technical name of the node is used. • XPath: For each column in the spreadsheet, the CSV displays the path to the node in the table.

Field separator	Specifies the field separator to use for exports. The default separator is comma.
List separator	Specifies the separator to use for values lists. The default separator is line return.

39.2 Imports

When importing a CSV file, you must specify one of the following import modes, which will determine how the import procedure handles the source records.

Insert mode	Only record creation is allowed. If a record exists in the target table with the same primary key as the source record, an error is returned and the whole import operation is cancelled.
Update mode	Only modifications of existing records are allowed. If no record exists in the target table with the same primary key as the source record, an error is returned and the whole import operation is cancelled.
Update or insert mode	If a record with the same primary key as the source record already exists in the target table, that record is updated. Otherwise, a new record is created.
Replace (synchronization) mode	If a record with the same primary key as the source record already exists in the target table, that record is updated. Otherwise, a new record is created. If a record exists in the target table but is not present in the source XML file, that record is deleted from the table.

In order to consider the inheritance during a CSV import, the following option has to be specified.

Enable inheritance	<p>Specifies whether the inheritance will be taken into account during a CSV import. If technical data in the CSV file define an inherit mode, corresponding fields or records are forced to be inherited. If technical data define an occult mode, corresponding records are forced to be occulted. Otherwise, fields are overwritten with values read from the CSV file. By default, this option is disabled.</p> <p>Note: Inheritance is always ignored if the data set of the table has no parent or if the table has no inherited field.</p>
---------------------------	--

39.3 Handling of field values

Aggregated lists

The CSV import and export services support multi-valued fields, namely aggregated lists. This is only supported for simple typed lists, such as lists of `string`, `date`, or `int`, and for table references. If a table reference is linked to a composite primary key, each item in the list is a formatted string, for example, "true|99". Aggregated lists of groups are not exported.

At export, the items in the list are separated using line separators. In cases where the exported field already contains a line separator, for example in an `osd:html` or an `osd:text`, the code `_crnl_` is inserted in place of the field value's line separators. The same formatting is expected at import, with the whole field value surrounded by quotes.

Hidden fields

Hidden fields are exported as `ebx-csv:hidden` strings. An imported hidden string will not modify a node's content.

'Null' value for strings

Using CSV import and export services, a string with a value set to `null` is exported as an empty string. Therefore, a round trip export-import procedure will end up replacing `null` string values with empty strings.

Date, time & dateTime format

The following date and time formats are supported:

Type	Format	Example
xs:date	yyyy-MM-dd	2007-12-31
xs:time	HH:mm:ss or HH:mm:ss.SSS	11:55:00
xs:dateTime	yyyy-MM-ddTHH:mm:ss or yyyy-MM-ddTHH:mm:ss.SSS	2007-12-31T11:55:00

39.4 Known limitations

Aggregated lists of groups

The CSV import and export services do not support importing multi-valued groups, that is, aggregated lists of complex type elements. Exporting such nodes will not cause any error, however, no value will be exported.

Terminal groups

In a CSV file, it is impossible to differentiate a created terminal group that contains only empty fields from a non-created one.

As a consequence, some differences may appear during comparison after performing an export followed by an import. To ensure the symmetry of import and export, use XML import and export instead. See [XML import and export](#) [p 209].

Column label headers

If two columns share the same label header, an export of the table can be performed successfully, but exported data cannot later be re-imported.

Association fields

The CSV import and export services do not support importing association values, that is multi-valued of complex type elements.

Exporting such nodes will not cause any error, however, no value will be exported.

Importing such nodes will cause an error and the import procedure will be aborted.

CHAPITRE 40

Supported XPath syntax

Ce chapitre contient les sections suivantes :

1. [Overview](#)
2. [Example expressions](#)
3. [Syntax specifications for XPath expressions](#)

40.1 Overview

The XPath notation used in EBX5 must conform to the *abbreviated syntax* of the [XML Path Language \(XPath\) Version 1.0](#) standard, with certain restrictions. This document details the abbreviated syntax that is supported.

40.2 Example expressions

The general XPath expression is:

path[*predicate*]

Absolute path

/library/books/

Relative paths

./Author

../Title

Root and descendant paths

//books

Table paths with predicates

../..books/[author_id = 0101 and (publisher = 'harmattan')]

/library/books/[not(publisher = 'dumesnil')]

Complex predicates

starts-with(col3, 'xxx') and ends-with(col3, 'yyy') and osd:is-not-null(./col3))

```
contains(col3 , 'xxx') and ( not(col1=100) and date-greater-  
than(col2, '2007-12-30') )
```

Predicates on label

```
osd:label(./delivery_date)='12/30/2014' and ends-with(osd:label(../  
adress), 'Beijing - China')
```

40.3 Syntax specifications for XPath expressions

Overview

Expression	Format	Example
XPath expression	<container path>[predicate]	/books[title='xxx']
<container path>	<absolute path> or <relative path>	
<absolute path>	/a/b or //b	//books
<relative path>	../b, ./b or b	../b

Predicate specification

Expression	Format	Notes/Example
<predicate>	Example: A and (B or not(C)) A,B,C: <atomic expression>	Composition of: logical operators braces, not() and atomic expressions.
<atomic expression>	<path> <comparator> <criterion> or method(<path>, <criterion>)	royalty = 24.5 starts-with(title, 'Johnat')booleanValue = true
<path>	<relative path> or osd:label(<relative path>)	Relative to the table that contains it: ../authorstitle
<comparator>	<boolean comparator>, <numeric comparator> or <string comparator>	
<boolean comparator>	= or !=	
<numeric comparator>	=, !=, <, >, <=, or >=	
<string comparator>	=	
<method>	<date method>, <string method>, osd:is-null method or osd:is-not- null method	
<date, time & dateTime method>	date-less-than, date-equal or date- greater-than	
<string method>	matches, starts-with, ends-with, contains, osd:is-empty, osd:is- not-empty, osd:is-equal-case- insensitive, osd:starts-with-case- insensitive, osd:ends-with-case- insensitive, or osd:contains-case- insensitive	
<criterion>	<boolean criterion>, <numeric criterion>, <string criterion>, <date criterion>, <time criterion>, or <dateTime criterion>	
<boolean criterion>	true, false	
<numeric criterion>	An integer or a decimal	-4.6
<string criterion>	Quoted character string	'azerty'
<date criterion>	Quoted and formatted as 'yyyy-MM-dd'	'2007-12-31'

Expression	Format	Notes/Example
<time criterion>	Quoted and formatted as 'HH:mm:ss' or 'HH:mm:ss.SSS'	'11:55:00'
<dateTime criterion>	Quoted and formatted as 'yyyy-MM-ddTHH:mm:ss' or 'yyyy-MM-ddTHH:mm:ss.SSS'	'2007-12-31T11:55:00'

XPath 1.0 formula

It is possible to use an XPath 1.0 formula in the criterion value part of an atomic predicate expression (right-hand side).

For example, instead of `[./a=3]`, you may use the expression `[./a=(floor(./d)+ 2.0)]`.

Predicate on label

The `osd:label()` function can be applied to the path portion of the atomic predicate, in order to resolve the predicate on the label instead of the value. In this case, only string operators and string criteria can be used, i.e. `ends-with(osd:label(./price), '99')`.

A predicate on label is localized, so the criterion must be expressed in the same locale as the predicate-filtered request. For example: `request.setLocale(Locale.FRENCH); request.setXPathFilter("osd:label(./delivery_date)='30/12/2014'");`

Note

It is forbidden to use the `osd:label` function if the right part of the predicate is a contextual value.

Note

If the `osd:label` function is used in a data model, for example on a selection or in the filter predicate of a table reference node, the default locale of the data model (as defined in its module declaration) must be used for the criterion format (even though this is generally not recommended).

Contextual values

For predicates that are relative to a selected node, the criterion value (that is, the right-hand side of the predicate) can be replaced with a contextual path using the syntax `${<relative-path>}` where `<relative-path>` is the location of the element relative to the selected node.

Note

When calling a method, the criterion is the second parameter, and the first parameter cannot be a relative value.

Aggregated lists

For predicates on aggregated lists, the predicate returns true regardless of the comparator if one of the list elements verifies the predicate.

Note

Special attention must be paid to the comparator `!=`. For example, for an aggregated list, `./list != 'a'` is not the same as `not(./list = 'a')`. Where the list contains the elements (e1,e2,...), the first predicate is equivalent to `e1 != 'a' or e2 != 'a' ...`, while the second is equivalent to `e1 != 'a' and e2 != 'a' ...`.

'Null' values

Null values must be explicitly treated in a predicate using the operators `osd:is-null` and `osd:is-not-null`.

For example, `/root/products[./price<100]` or `/root/products[./price!=100]` will not return any products whose prices are not set (null). For the latter case to return unset values as well, the predicate must instead be: `/root/products[./price!=100 or osd:is-null(./price)]`.

How to manage single and double quotes in literal expressions

By default, a literal expression is delimited by single quotes (`'`). If the literal expression contains single quotes and no double quotes, the expression must be delimited by double quotes (`"`). If the literal expression contains both single and double quotes, the single quotes must be doubled.

Examples of using `encodeLiteralStringWithDelimiters`

Value of Literal Expression	Result of this method
Coeur	'Coeur'
Coeur d'Alene	"Coeur d'Alene"
He said: "They live in Coeur d'Alene".	'He said: "They live in Coeur d'Alene".'

Extraction of foreign keys

The standard XPath syntax has been extended so as to extract the value of any targeted primary key field.

Example

If the table `/root/tableA` has an `osd:tableRef` field named 'fkB' whose target is `/root/tableB` and the primary key of `tableB` has two fields, `id` of type `xs:int` and `date` of type `xs:date`, then the following expressions would be valid:

- `/root/tableB[fkB = '123|2008-01-21']`, where the string "123|2008-01-21" is a representation of the entire primary key value.
- `/root/tableB[fkB/id = 123 and date-equal(fkB/date, '2008-01-21')]`, where this predicate is a more efficient equivalent to the one in the previous example.
- `/root/tableB[fkB/id >= 123]`, where any number operator could be used, as the targeted primary key field is of type `xs:int`.
- `/root/tableB[date-greater-than(./fkB/date, '2007-01-01')]`, where any date operator could be used, as the targeted primary key field is of type `xs:date`;
- `/root/tableB[fkB = ""]` is not valid as the targetted primary key has two columns.
- `/root/tableB[osd:is-null(fkB)]` checks if a foreign key is null (notdefined).