



Documentation Produit

EBX5 Version 5.4.0 Fix 002

Table des matières

Guide utilisateur

Introduction

1. Notions clés.....	9
2. Interface utilisateur.....	11
3. Glossaire.....	15

Modèles de données

4. Introduction aux modèles de données.....	26
5. Utilisation de l'interface utilisateur de la section Modèles de Données.....	29

Implémentation des modèles de données

6. Création du modèle de données.....	31
7. Configuration du modèle de données.....	33
8. Modélisation de la structure des données.....	35
9. Propriétés des éléments du modèle de données.....	41
10. Contrôles sur les éléments du modèle de données.....	47
11. Actions sur les modèles de données existants.....	51

Publication et gestion de versions des modèles de données

12. Publication du modèle de données.....	53
13. Gestion des versions de modèles de données embarqués.....	55

Espaces de données

14. Introduction aux espaces de données.....	60
15. Actions principales.....	63
16. Image.....	67
17. Fusion d'un espace de données.....	69
18. Permissions.....	73

Jeux de données

19. Introduction aux jeux de données.....	78
20. Actions principales.....	81
21. Vues personnalisées.....	87
22. Héritage.....	91
23. Permissions.....	95

Modèles de workflow

24. Introduction aux modèles de workflow.....	98
25. Actions principales.....	101
26. Modélisation du workflow.....	105
27. Tâche utilisateur.....	111
28. Publication d'un modèle de workflow.....	115

Workflows de données

29. Introduction aux workflows de données.....	118
30. Utilisation de l'interface utilisateur de la section Workflow de données.....	121
31. Bons de travail.....	125

Gestion de workflows de données

32. Lancement et monitoring de workflows de données.....	129
33. Administration de workflows de données.....	131

Services de données

34. Introduction aux services de données.....	136
35. Actions principales.....	139

Manuel de référence (en anglais)

Intégration

36. Using EBX5 as a web component.....	145
37. Built-in UI services.....	149
38. Data services.....	161

Services d'import et export

39. XML import and export.....	185
40. CSV import and export.....	189
41. Supported XPath syntax.....	193

Divers

42. Permissions.....	200
43. Inheritance and value resolution.....	209
44. Criteria Editor.....	213

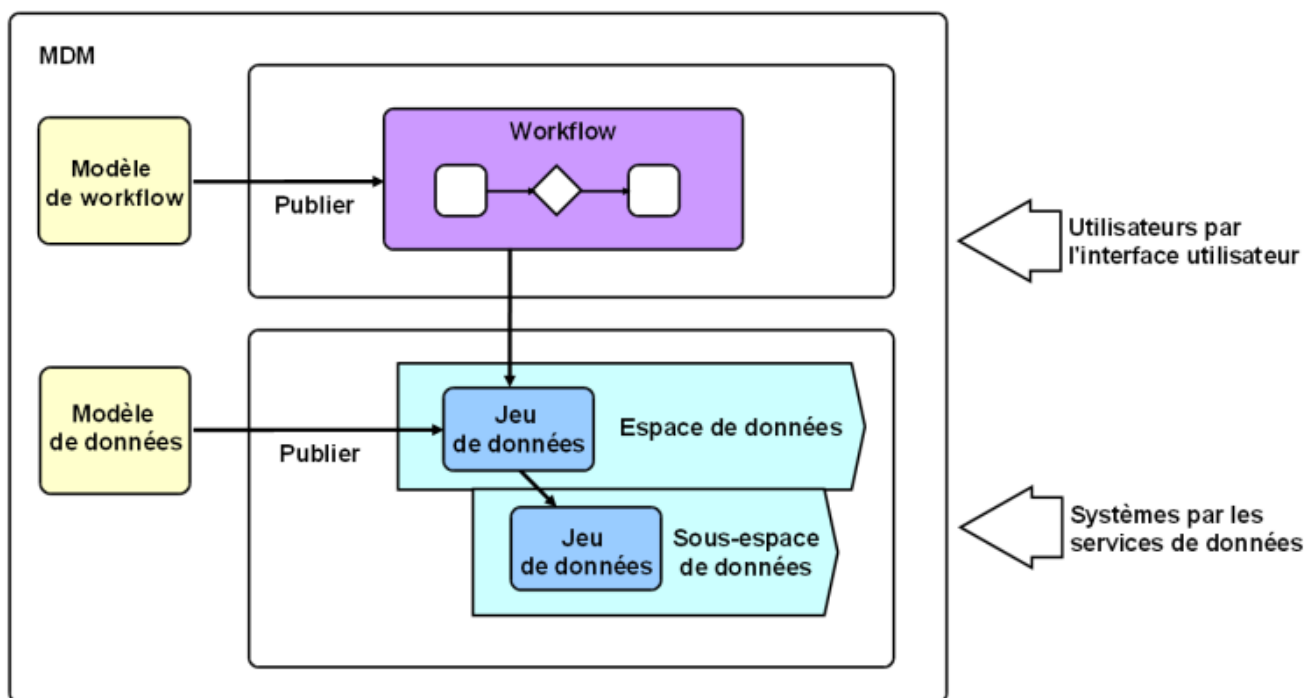
Guide utilisateur

Introduction

CHAPITRE 1

Notions clés

Concepts et outils associés



La gestion des données de référence (plus connue sous l'acronyme MDM pour Master Data Management) est un moyen de mieux modéliser, gérer et ultimement gouverner vos données partagées. La duplication des données au sein de plusieurs systèmes informatiques et son partage par des équipes professionnelles multiples rend critique le besoin d'avoir une version unique et une gouvernance de vos données de référence.

Avec EBX5, les utilisateurs métier et les informaticiens peuvent collaborer sur une seule et unique solution unifiée de manière à 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 des données, la gestion hiérarchique des données, le contrôle de version, et la sécurité.

Un projet MDM sur EBX5 commence par la création d'un modèle de données. C'est là que vous définissez les tables, les champs, les liens et les règles métier permettant de décrire vos données de référence. De bons exemples sont les catalogues de produits, les hiérarchies financières, la liste des fournisseurs ou simplement les tables de référence.

Votre modèle de données peut ensuite être publié en tant que jeu de données, stockant le contenu actuel de vos données de référence. Un jeu de données se trouve dans un espace de données. Un espace de données est un conteneur, qui permet d'isoler des mises à jour ; cela est très utile si vous avez besoin de travailler sur plusieurs versions parallèles de vos données, de réaliser une analyse d'impact ou de travailler en "espaces de démonstration".

Une fois que tout est prêt à l'emploi, vous pouvez définir des processus de gestion des données, que vous allez exprimer sous forme de modèles de workflow dans EBX5. Ces modèles détaillent les tâches à accomplir ainsi que les responsabilités associées. Après publication de ces modèles, il sont disponibles à l'usage sous forme de workflows.

Les workflows sont de grande valeur si vous avez besoin de réaliser une gestion contrôlée du changement ou une validation des données pas à pas impliquant des utilisateurs multiples. Une fois démarrés, des notifications sont envoyées aux utilisateurs sur le nombre de bons de travail mis à leur disposition dans un contexte de travail collaboratif.

Les services de données aide à intégrer EBX5 à des systèmes tiers (middlewares), en leur permettant d'accéder aux données, ou de gérer des espaces de données et/ou des workflows.

Les mots-clés à comprendre sont :

- [Modèle de données](#)
- [Jeu de données](#)
- [Espace de données](#)
- [Modèle de workflow](#)
- [Workflow de données](#)
- [Service de données](#)

Des définitions détaillées peuvent être trouvées dans notre [glossaire](#).

CHAPITRE 2

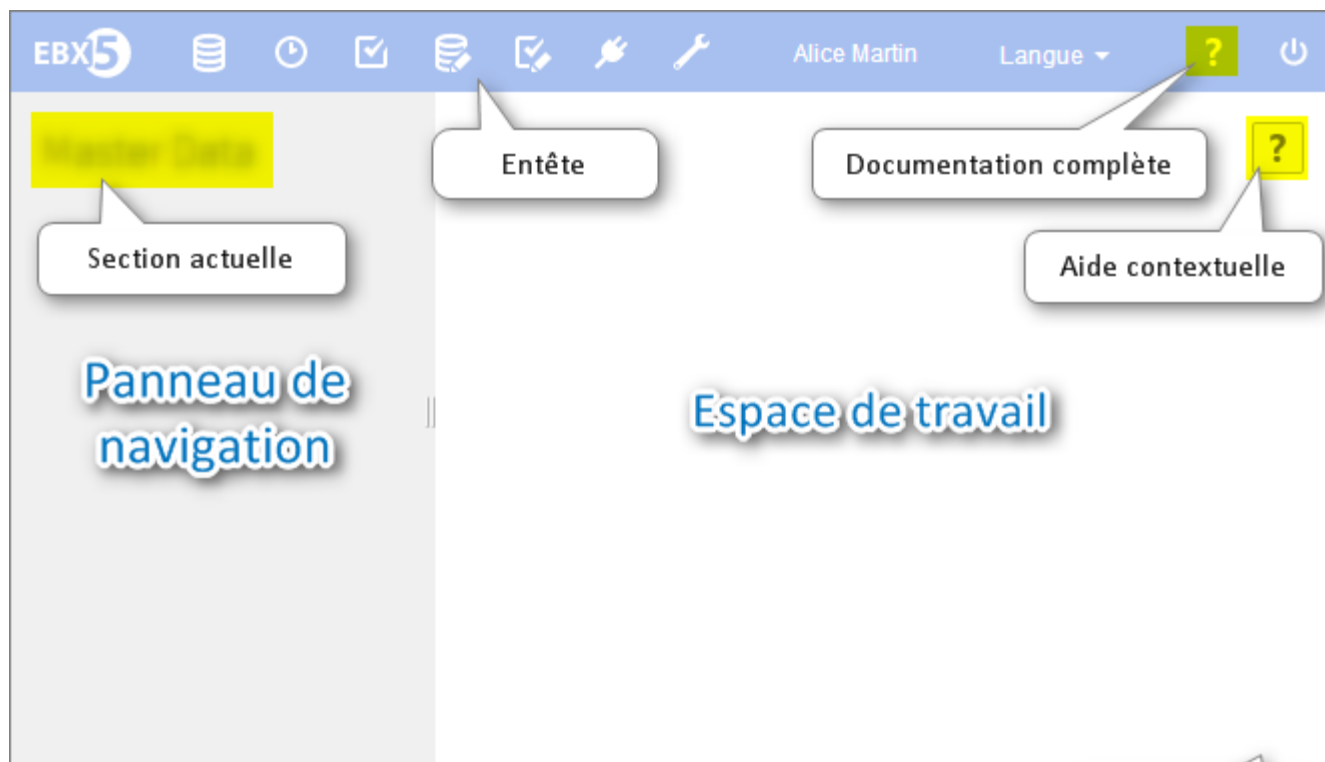
Interface utilisateur

Comment vous y retrouver dans EBX5

L'interface de EBX5 est divisée en plusieurs régions générales, référencées dans la documentation en utilisant les termes suivants :

- **Entête** : le nom de l'utilisateur courant s'affiche dans cette zone, ainsi que la sélection de langue, un lien vers la documentation et un bouton pour fermer la session courante.
- **Barre de menu** : cette zone comprend toutes les fonctionnalités accessibles à l'utilisateur 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** : il s'agit d'une zone de travail. 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*.



Où trouver de l'aide sur EBX5

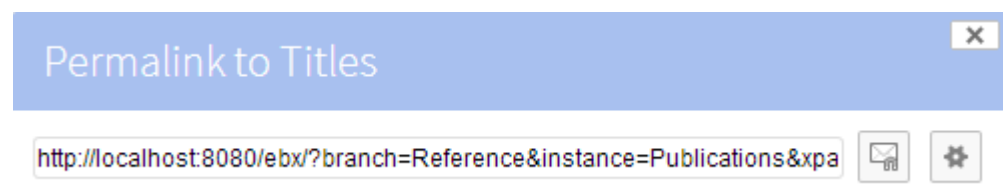
A part de la documentation de produit complète, l'aide est accessible de différentes façons dans l'interface.

Aide contextuelle

Lorsque la souris survole un élément ayant sa propre aide, un point d'interrogation apparaît. Quand vous cliquez sur cet élément, un petit panneau apparaît avec les informations relatives à cet élément.



De plus, lorsque cela est possible, un bouton est disponible à droite du libellé permettant de récupérer un permalien vers l'élément.



CHAPITRE 3

Glossaire

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](#).

profil

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

Voir [utilisateur](#), [rôle](#).

utilisateur

Entité créée dans le référentiel afin de permettre à des personnes physiques ou des systèmes externes de s'authentifier et accéder à EBX5. Un utilisateur peut être affecté à un ou plusieurs rôles, et possède diverses informations de compte tels que nom, prénom, login, email etc.

rôle

Classification d'utilisateur, utilisée pour les règles de permission et les workflows de données, pouvant être affectée à des utilisateurs. 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 de façon à définir à qui les bons de travail sont proposés. Lors de l'exécution du workflow de données, ce bon de travail pourra être proposé à tous les utilisateurs membres de ce rôle.

Modélisation de données

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

modèle de données


Définition structurée des données à gérer dans le référentiel EBX5. Un modèle de données comprend des descriptions détaillées de toutes les données incluses, en termes d'organisation, de types de données 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 qui contient les données gérées par le référentiel.

Voir [jeu de données](#).

Concept apparenté [Modèles de données](#).

champ

Élément de base du modèle de données qui est défini par un nom et un type. 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 maximum supérieure à 1. Chaque élément de la liste ainsi définie sera du même type que le champ initial. Les champs peuvent être organisés en groupes pour faciliter l'organisation du modèle de données.

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

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

Concept apparentés [Propriétés des éléments de structure](#), [Contrôles sur les champs de données](#).

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](#).

table (modèle de données)

Element 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 utiliser 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](#), [clé primaire](#), [type réutilisable](#).

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 utiliser 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](#).

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. Il permet de définir la structure d'un modèle, de créer et éditer ses éléments, de configurer et publier le modèle.


Voir [Modèles de données](#).

Gestion des données

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

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 \(jeu de données\)](#), [clé primaire](#).

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](#), [clé primaire](#).

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 \(jeu de données\)](#), [champ](#), [groupe](#), [vue personnalisée](#).


Les jeux de données sont représentés par l'icône .

Concept apparenté [Jeux de données](#).

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. Cette fonctionnalité est utile par exemple lorsqu'on définit un modèle de données où il y a des données au niveau du parent qui doivent avoir la même valeur par défaut au niveau des jeux de données enfant. Il est possible de surcharger les valeurs héritées du parent dans les jeux de données enfants. 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 au grain d'un champ et non plus du jeu de données.

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, et permet de spécifier l'affichage des enregistrements en mode hiérarchique ou tabulaire, ainsi que des critères de filtrage et de tri.

Voir [vue hiérarchique](#).

Concept apparenté [Vues personnalisées](#).

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](#).

Concept apparenté [Hiérarchies](#).

Cycle de vie des données

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

espace de données

Conteneur de 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 sur 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](#), [référentiel](#), [fusion](#).

Concept apparenté [Data spaces](#).

espace de données de référence

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

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

fusion

Integration des changements réalisés dans un espace de données enfant depuis sa création dans son espace de données parent. L'espace de données enfant est fermé après que la fusion ait été 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 modifications sur les 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](#).

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](#)

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 comme l'export, la comparaison, les filtres, sont 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](#), [vue historique des transactions](#), [profil d'historisation](#).

profil d'historisation

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

Voir [profil d'historisation](#).

vue historique de table

Vue contenant la trace de toutes les modifications effectuées sur une table donnée, notamment les créations, les mises à jour et les suppressions. Chaque entrée présente les informations transactionnelles comme l'estampillage et l'utilisateur ayant effectué l'action, ainsi que les données à

la fin 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/toucher plusieurs tables dans un ou plusieurs jeux de données, cette vue montre toutes les opérations qui se sont effectuées dans le périmètre en question pour chaque transaction.

Modélisation de workflow

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

modèle de workflow

Définition de la succession des 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 terme d'états ou d'actions associées à effectuer par des utilisateurs et des tâches automatiques.

Concept apparenté [Modèles de workflow](#).

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](#).

tâche utilisateur

Tâche de workflow qui est décomposée en 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](#).

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 .

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.

Workflows de données

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

publication de workflow

Version particulière d'un modèle de workflow de données qui est mise à disposition des utilisateurs qui ont les permissions nécessaires afin de pouvoir 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](#).

Concept apparenté [Workflows de données](#).

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](#)

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](#).

Services de données

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

service de données

Méthode par laquelle des systèmes externes peuvent interagir programmatiquement avec EBX5 et les données gérées dans le référentiel EBX5. Les services de données sont des services web standard, qui peuvent être utilisés pour accéder à une partie des fonctionnalités disponibles par le biais de l'interface utilisateur.

Concept apparenté [Services de données](#).

lignage

Mécanisme grâce auquel des profils de droit d'accès peuvent être mis en place pour des utilisateurs non humains, nommés services de données. Les profils de droit d'accès ainsi définis sont utilisés lors de l'accès aux données via des interfaces WSDL.

Voir [service de données](#).

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

Modèles de données

CHAPITRE 4

Introduction aux modèles de données

Présentation

La fonction d'un modèle de données

La première étape de toute gouvernance de données dans EBX5 est de développer 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'organisations, de types de données, et de relations sémantiques. Une fois qu'un modèle de données a été défini et publié, vous pouvez créer des jeux de données à partir de celui-ci.

Afin de définir un modèle de données dans le référentiel, vous devrez d'abord créer un nouveau modèle de données, puis définir sa structure et les propriétés de ses éléments (tables, champs et groupes). Lorsque vous aurez défini votre modèle de données, vous le publierez pour le rendre disponible et ainsi vous et les autres utilisateurs pourrez créer des jeux de données basés sur cette publication pour contenir les données qui sont 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](#)
- [clé primaire](#)
- [clé étrangère](#)
- [table](#)
- [groupe](#)
- [type réutilisable](#)
- [règle de validation](#)

Concepts apparentés :

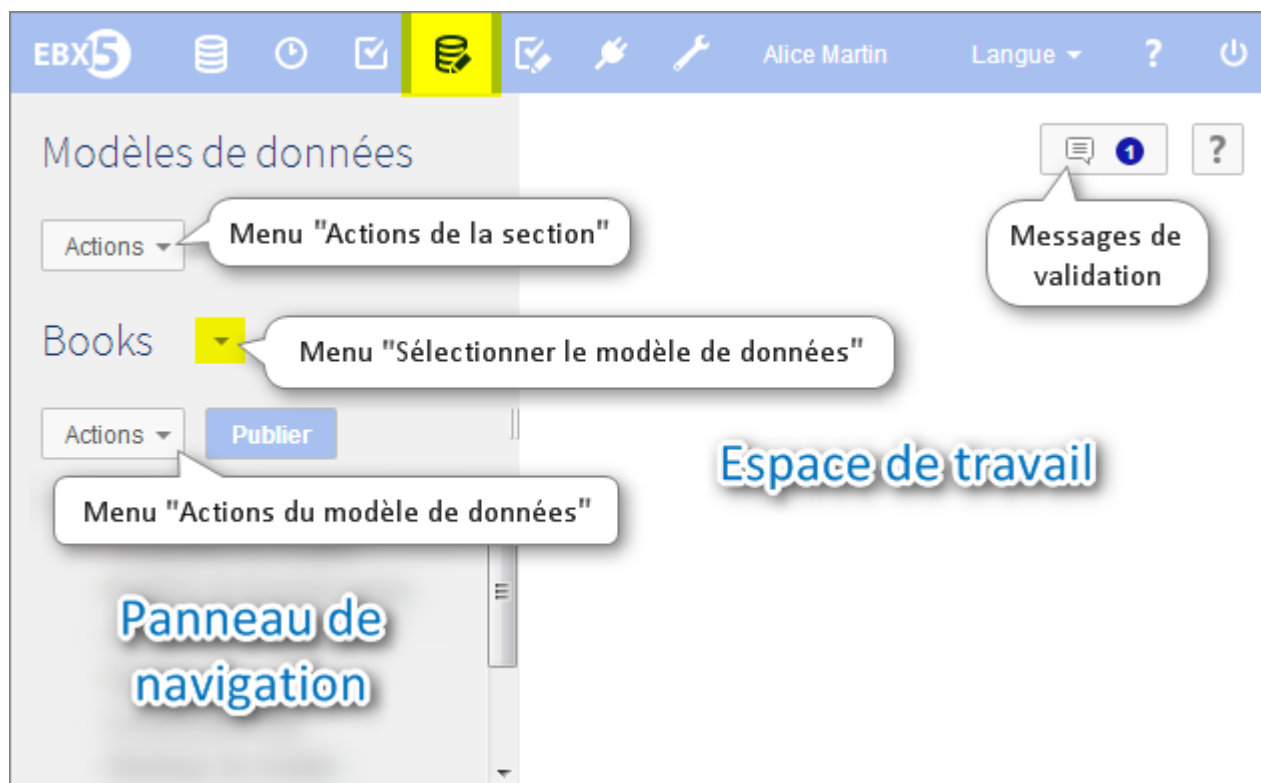
- [*Espaces de données*](#)
- [*Jeux de données*](#)

CHAPITRE 5

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 **Modélisation > Modèles de données**. Le Data Model Assistant de EBX5 aide le développement des modèles de données.



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.

Icônes des éléments du modèle de données

 [champ](#)

 [clé primaire](#)

 [clé étrangère](#)

 [table](#)

 [groupe](#)

Voir aussi :

- [Modélisation de la structure des données](#)
- [Configuration du modèle de données](#)
- [Types réutilisables](#)

CHAPITRE 6

Création du modèle de données

Création d'un modèle de données

Pour créer un modèle de données, utilisez le menu "[Sélectionner le modèle de données](#)"  dans le panneau de navigation, cliquez sur le bouton **Créer** dans la pop-up, puis suivez les instructions de l'assistant de création de modèle de données.

CHAPITRE 7

Configuration du modèle de données

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** dans le panneau de navigation.

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.

Permissions

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

La configuration des permissions d'un modèle de données est identique aux options des permissions d'un jeu de données, expliquée dans la section [Permissions](#).

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 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.

Modèles de données inclus

Vous pouvez utiliser les types de données définis dans un autre modèle de données 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 l'enregistrement du modèle inclus depuis cette table, vous trouverez des informations techniques liées à ce modèle sous l'onglet **Information**. Car des erreurs de validation peuvent se produire dans un modèle de données inclus plus tard, par exemple à cause de ressources Java supprimés, cette vue vous donnera les informations sur les erreurs.

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

Les noms des types de données doivent être unique pour tous les définitions de types de données définis en local et inclus. Autrement dit, les types de données inclus ne peuvent pas avoir des noms qui correspondent avec ceux des types de données définis dans le modèle de données courant ou un autre modèle de données inclus.

CHAPITRE 8

Modélisation de la structure des données

Pour définir la structure du modèle de données, sélectionnez le modèle de données avec lequel vous voulez travailler en utilisant le menu "[Sélectionner le modèle de données](#)"  dans le panneau de navigation.


La structure du modèle de données est accessible dans 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.

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

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

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 pourra plus être modifié.


Vous avez la possibilité 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 localisé de l'élément.

Supprimer des éléments

Tous les éléments sauf le noeud `root` 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. Il est nécessaire de spécifier 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, et en dernière position. Si vous dupliquez un élément qui contient autres éléments, tous les sous éléments sont dupliqués avec leurs noms originaux.

Note

Si vous dupliquez 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

Il est possible de déplacer un élément en utilisant les flèches orientées vers le haut et le bas situées à côté de l'élément à déplacer.

Note

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



Types réutilisables


Les types réutilisables sont des éléments partagés qui sont créés une fois, puis réutilisés par différents éléments du modèle de données.

Note

En modifiant la définition d'un type réutilisable, vous modifiez la structure de tous les éléments basés sur ce type réutilisable. L'arborescence "Structure de données" affiche la structure en lecture seule d'un groupe ou d'une table qui utilise un type réutilisable. Pour éditer la structure du type réutilisable associé, vous devez accéder 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  associé aux sections "Types de données simples" ou "Types de données complexes" dans le panneau de navigation, vous pouvez créer des types simples et des types complexes réutilisables. Vous avez aussi la possibilité de créer un type réutilisable lors de la création d'un élément complexe (table ou groupe). Dans ce cas, après la création de l'élément, le type réutilisable sera disponible pour la création d'autres éléments avec la même structure et les mêmes propriétés. Vous pouvez également 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 est utilisé par aucun élément, vous pouvez le supprimer 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 lors de la création d'un élément. L'élément créé utilisera la structure et les propriétés du type réutilisable.

Inclusion des types de données définis dans autres modèles de données

Les types réutilisables peuvent aussi être partagé entre plusieurs modèles de données. En configurant l'inclusion d'un modèle de données externe, vous pouvez utiliser les types de données inclus pour créer des nouveaux éléments dans la structure de données de la même façon que les types réutilisables définis en local.

Note

Car les noms de types de données doivent être unique pour tous les types définis en local et inclus, vous ne pouvez pas créer un nouveau type réutilisable qui a le même nom qu'un type de données dans un modèle de données inclus. Egalement, ce n'est pas possible d'inclure un modèle de données externe qui définit un type de données avec le même nom qu'un type réutilisable défini en local ou dans un autre modèle de données inclus.

Les types de données inclus apparaît dans les sections "Types de données simples inclus" et "Types de données complexes inclus" dans le panneau de navigation. Vous pouvez consulter les détails de ces types réutilisables, mais il ne peuvent être édités que dans leurs modèles de données d'origine.

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

Détails de la création des éléments du modèle de données

Création de champs

Quand vous créez un champ, vous êtes obligé de sélectionner un type de données. Ce type 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.

Création de tables

Lors la création d'une nouvelle table, vous avez la possibilité de créer un type réutilisable, basé sur la définition de cette table. Cela permet de réutiliser la même structure de table à différents endroits du modèle de données. Vous pouvez également utiliser un type réutilisable existant pour la définition de la nouvelle table. Voir [Types réutilisables](#) pour plus d'informations.

A la fin de l'assistant de création de table, vous pouvez créer directement une clé primaire qui sera associée à la nouvelle table. Chaque table nécessite la désignation d'au moins un champ clé primaire. Si vous décidez de ne pas créer un champ clé primaire lors de la création de la table, il sera possible de créer une clé primaire ultérieurement à partir du menu d'actions disponible sur la table dans la section "Structure de données " du panneau de navigation.


Création de groupes

Lors de la création d'un groupe, vous avez la possibilité de créer un type réutilisable, ou d'utiliser un type réutilisable existant. Voir [Types réutilisables](#) pour plus d'informations.

Création de clés primaires

Une clé primaire est nécessaire pour chaque table. Après la création d'une table, vous avez la possibilité de créer directement des champs appartenant à la clé primaire de la table à partir de l'assistant de création de tables.

Il est toutefois possible de créer une clé primaire à n'importe quel moment, 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 en utilisant le menu avec la flèche  situé à droite de ce champ.

Création ou définition de clés étrangères

Les champs associés à une clé étrangère ont le type de données "Chaîne de caractères". Vous pouvez créer une clé étrangère vers une table située dans le même modèle directement depuis la structure de données, ou vous pouvez 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.

Création d'un catalogue d'attributs utilisateurs


Un catalogue d'attributs utilisateurs est nécessaire pour créer des champs de type "attribut utilisateur". Lorsque vous créez un nouveau catalogue d'attributs utilisateurs, vous devez définir son nom. Une fois créé, ce catalogue pourra être utilisé lors de la création de champs de type "attribut utilisateur".

Création d'un attribut utilisateur

Lorsque vous créez un attribut utilisateur, vous devez spécifier un catalogue existant de d'attributs utilisateurs. Ce catalogue contient les attributs qui seront associés avec le nouvel attribut utilisateur. Après la création de ce nouvel attribut, le catalogue référencé pourra être modifié en modifiant la propriété "Chemin du catalogue UDA" dans la section "Propriétés avancées".

Modification des éléments existants

Suppression d'un champ de la clé primaire

Tous champs appartenant à la clé primaire peut être supprimé de la clé primaire d'une table en utilisant le menu avec la flèche  situé à droite du champ. Le champ est supprimé de la définition de la clé primaire mais n'est pas supprimé de la structure de données.

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

CHAPITRE 9

Propriétés des éléments du modèle de données

Après la création d'un élément, vous pouvez définir des propriétés supplémentaires pour compléter sa définition.

Voir aussi : [*Contrôles sur les éléments du modèle de données*](#)

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". Pour les éléments de type "Noeud de sélection" le nombre minimum de valeurs 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 <code>Mot de passe</code>, les types réutilisables, les champs dans les types complexes réutilisables, et les noeuds de sélection. Cette propriété est utilisée pour définir des règles de validation riches et complexes avec 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 des autres champs.</p> <p>En utilisant l'assistant associé, vous pouvez définir des libellés localisés pour la règle de validation, ainsi qu'un message localisé avec sévérité qui sera affiché 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 dans les formulaires de création de nouveaux enregistrements. Le type de la valeur par défaut doit être en accord 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 en accord avec le type du champ courant.
Règle de calcul	<p>Cette propriété est disponible pour les champs dans les tables sauf 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 peuvent être utilisées seulement avec les champs simples dans une table.• Les règles de calcul ne peuvent pas être définies sur les champs du type <code>OResource</code> ou <code>Password</code>.• Les règles de calcul ne peuvent pas être définies sur les noeuds de sélection et les champs clés primaires.• Les règles de calcul ne peuvent pas être définies en accédant à l'élément depuis le rapport de validation.

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" ou "décimal" contenus dans une table. Si elle est activée, la valeur du champ est calculée automatiquement lors de la création d'un

EBX5 documentation > Guide utilisateur > Modèles de données > Implémentation des modèles de données > Propriétés des éléments du modèle de données
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 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.

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](#) dans le glossaire et la section [Inherited fields](#) pour plus d'informations.

Propriétés avancées des tables

Les propriétés avancées suivantes sont spécifiques aux tables.

Table

Clé primaire	<p>Une liste des champs composant la clé primaire de la table. Vous pouvez ajouter ou supprimer des champs de la clé primaire.</p> <p>Chaque champ de la clé est dénoté par un chemin XPath absolu qui débute sous la table.</p>
Index	<p>Définit la liste des champs à indexer dans la table. L'indexation accélère les accès à la table pour les requêtes impliquant les champs indexés. Il n'est pas possible de définir deux index portant exactement sur la même liste de champs.</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 dénoté par un chemin XPath absolu qui débute sous la table.</p>
Profil d'historisation	<p>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.</p>
Libellé	<p>Définit les champs composant le libellé par défaut et les libellés localisés pour les enregistrements de la table.</p>
Filtres spécifiques	<p>Définit des filtres pour afficher uniquement les enregistrements correspondant à certains critères.</p>
Présentation	<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 plus 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>

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". Ceci est fait afin d'éviter les complexités d'affichage qui se posent quand les liens et les onglets sont dans la même interface utilisateur.

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*](#)

CHAPITRE 10

Contrôles sur les éléments du modèle de données

Après la création d'un élément, vous pouvez définir des contrôles supplémentaires pour compléter sa définition.

Voir aussi : [*Propriétés des éléments du modèle de données*](#)

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	Le nombre exact de caractères requis pour ce champ.
Longueur minimale	Nombre minimum de caractères requis pour ce champ.
Longueur maximum	Nombre maximum de caractères autorisé pour ce champ.
Pattern	Une 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 autorisé dans la partie décimale de la valeur d'un champ de type décimal.
Nombre total de chiffres	Nombre maximum de chiffres composant la valeur d'un champ de type entier ou décimal.
Enumération	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 donnée alors une énumération représentant l'intersection entre ces deux énumérations sera utilisées 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.

Contrôles avancés

Il est possible d'établir des contrôles avancés sur le contenu d'un élément en définissant des contrôles dynamiques et 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 dans un autre jeu de données dans le même espace de données, ou un jeu de données dans un autre espace de données.
Mode	<p>Emplacement de la table ciblée par la clé étrangère.</p> <p>"Défaut" : le modèle courant.</p> <p>"Autre jeu de données" : un jeu de données qui se trouve dans le même espace de données.</p> <p>"Autre espace de données" : un jeu de données appartenant à un espace de données différent.</p>
Table référencée	Expression XPath indiquant l'emplacement de la table. Par exemple, <code>/racine/MaTable</code> .
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. Le 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.
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 minimale non autorisée pour ce champ.</p> <p>Valeur minimale exclue : La valeur maximale non autorisée pour ce champ.</p>
Enumération 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.

Messages 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 ces messages 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'.
Message	Le 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](#)

CHAPITRE 11

Actions sur les modèles de données existants

Lorsque votre modèle de données est créé, vous pouvez effectuer des actions qui sont disponibles dans le menu **"Actions" du modèle de données** dans le panneau de navigation.

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, vous pouvez cliquer sur le bouton **Revalider** pour mettre à jour ce rapport, ou vous pouvez cliquer 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.

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** 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 doit être conforme aux règles de validation du référentiel EBX5. Si le document déclare de 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, vous ne pourrez pas importer le fichier XSD. Voir [Propriétés du modèle de données](#) 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.

Vous pouvez importer un document XML Schema (XSD) 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.

Vous pouvez 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".

Duplication d'un modèle de données

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

Suppression d'un modèle de données

Pour supprimer un modèle de données, sélectionner "Supprimer" dans le menu [Actions du modèle de données](#). Quand vous supprimez un modèle de données, toutes les publications associées au modèle restent disponibles. Si vous créez un nouveau modèle de données avec le même nom que celui que vous avez supprimé, le nouveau modèle de données sera reassocié avec toutes les publications existantes dans le référentiel. Au moment de la publication, vous aurez l'opportunité 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](#) pour plus d'informations sur le processus de publication.

CHAPITRE 12

Publication du modèle de données

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 **Data Model Assistant**. Une publication est créée la première fois que vous publiez un modèle de données en utilisant le bouton **Publier** dans le 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](#) pour plus d'informations.

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

Modes de publication

Vous pouvez publier un modèle de données en mode "Embarqué" ou en mode "Dans un module". Le mode de publication "Embarqué" génère une publication qui est 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 quand vous cliquez 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.

Mode de publication "Embarqué"

La première fois que vous publiez un modèle de données 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 vous devrez sélectionner la publication à mettre à jour.

Pendant le processus de publication, si le modèle de données a été déjà publié, vous avez l'opportunité de visualiser les différences structurelles introduites par la nouvelle publication dans une interface de comparaison.

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 état précédent 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 de versions du modèle de données embarqué](#) pour plus d'informations sur les versions des modèles de données.

CHAPITRE 13

Gestion des versions de modèles de données

A propos des versions

Vous pouvez créer des *versions* pour les modèles de données afin 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.

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**.

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

Actions sur les versions

Dans l'espace de travail, en utilisant le menu avec la flèche vers le bas ▼ à côté de chaque version, vous pouvez effectuer les 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 leurs contenus évoluent indépendamment.
Définir en tant que version par défaut	Définit la version par défaut qui est sélectionnée lorsqu'un utilisateur accède au modèle de données.
Exporter une archive	Exporte le modèle de données sélectionné dans une archive contenant le contenu de la version, ses permissions et ses informations. L'archive exportée est localisée dans le répertoire d'archives, qui est accessible aux 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 la droite d'une version. 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 leurs cases 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 la gauche et la plus récente à la droite.

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 jours 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 utilisées par un modèle de données embarqué ne sont pas versionnées lorsqu'une version est créée. En effet, seul les références des ressources sont sauvegardées, et c'est la responsabilité des développeurs de s'assurer que les ressources référencées restent compatibles avec les différentes versions les utilisant.

Espaces de données

CHAPITRE 14

Introduction aux espaces de données

Un espace de données est un récipient de données, dont le contenu peut être mis à jour en toute isolation, c'est-à-dire sans affecter les données situées à l'extérieur de celui-ci. Après avoir décidé comme vous souhaitez organiser vos données, vous pouvez être amené à :

- créer des espaces de données (voir [création](#)),
- les étiqueter et en décrire le contenu et l'utilité (voir [information](#)),
- exporter ou importer un jeu de données dans l'espace de données, puis valider ce changement (voir [import d'archive](#) ou [export archive](#)),
- comparer le contenu de deux espaces de données (voir [actions](#)),
- prendre une image de l'espace de données avant d'y appliquer tout autre changement (voir [image](#)),
- appliquer les mêmes changements d'un espace de données fils à son parent (voir [fusion](#)),
- gérer les droits d'accès à un espace de données (voir [permissions](#)),
- fermer un espace de données, qui n'est plus requis (voir [fermeture](#)).

Un espace de données est toujours créé à partir d'un autre espace de données, à l'exception de l'espace de données de référence, qui est la racine de tous les autres espaces de données.

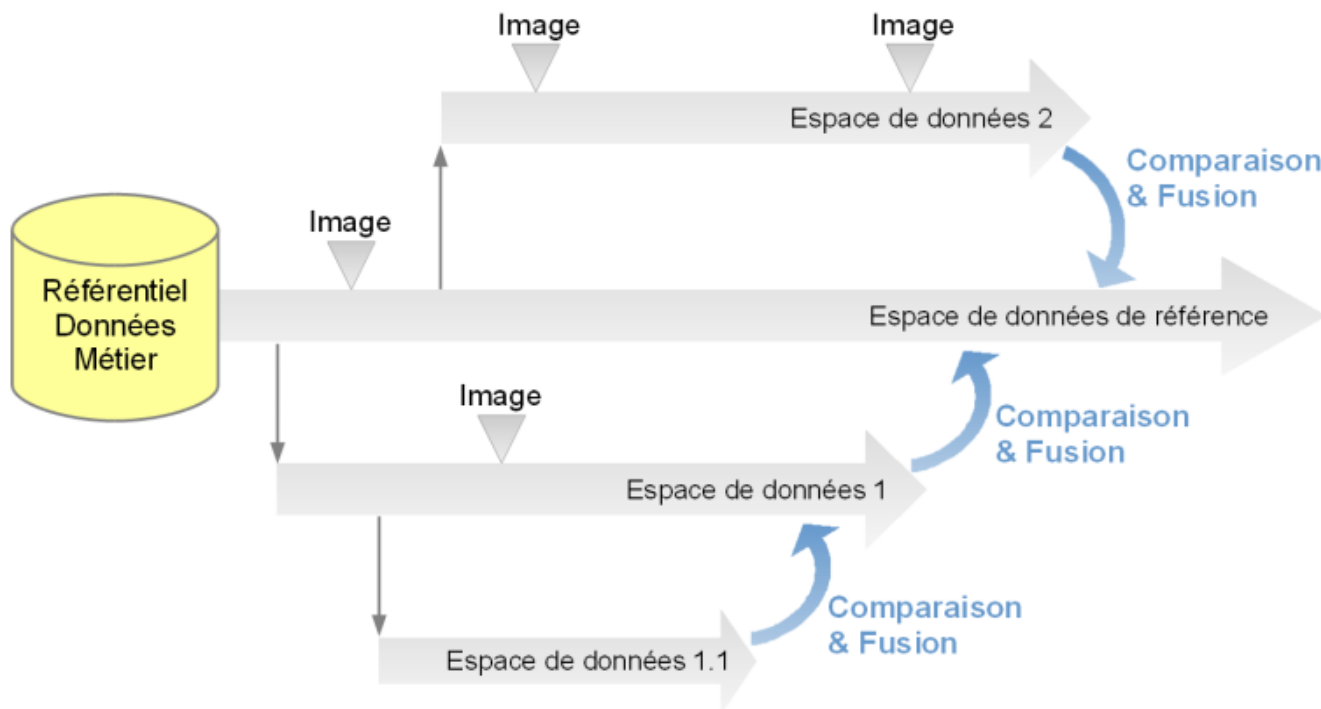
Pour une meilleure compréhension, n'hésitez pas à chercher un mot dans le [glossaire](#).

Concepts

Le cycle de vie des données est souvent complexe. Par exemple, une entreprise a besoin d'avoir une version courante de ses données tout en travaillant sur des évolutions futures. De plus, cette entreprise a besoin de conserver une trace de ses évolutions.

EBX5 permet de créer et gérer plusieurs espaces de données ainsi que des images. Il est possible, en utilisant des espaces de données, de faire des modifications simultanées dans un même référentiel, de les comparer et de les fusionner.

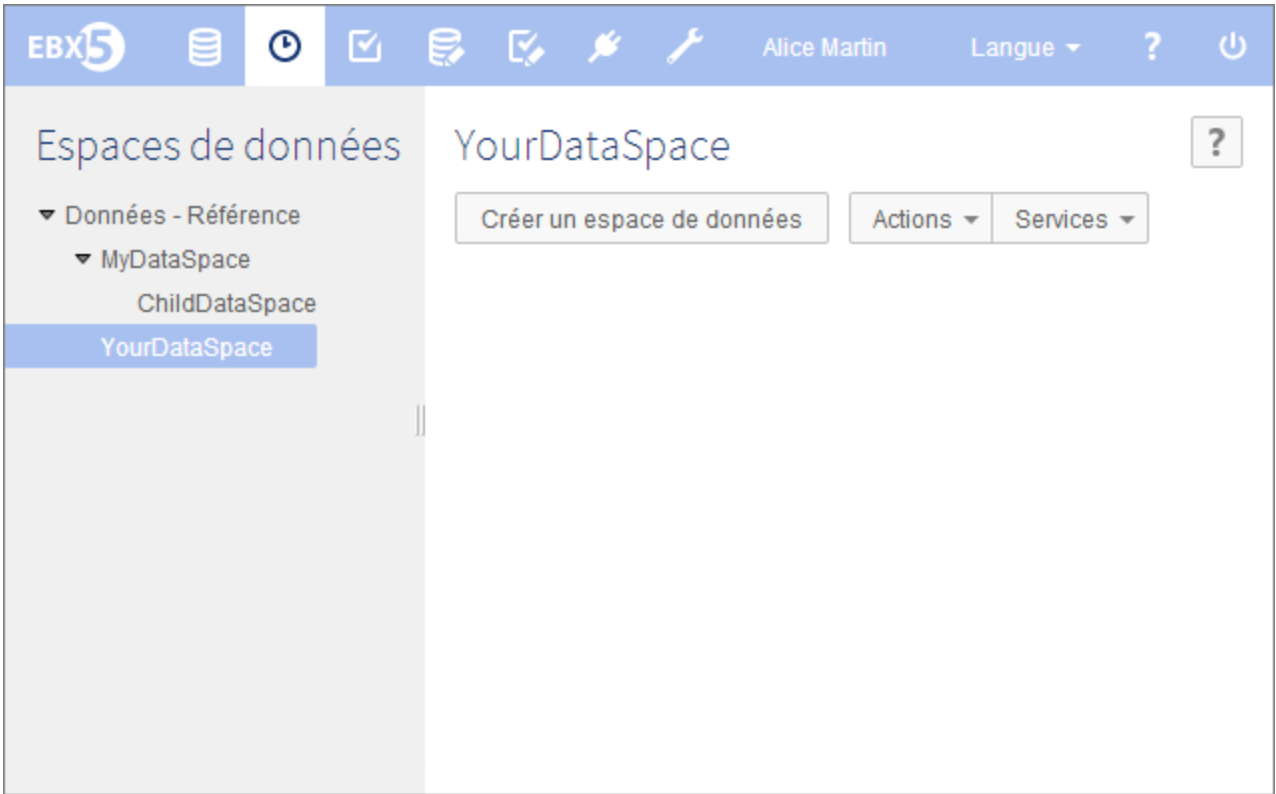
Une image permet de prendre un instantané d'un espace de données afin de conserver une version des données et de pouvoir détecter des modifications ultérieures.



Vue d'ensemble d'un espace de données

Modalités d'accès dans l'interface

Le panneau de navigation affiche l'organisation hiérarchique des espaces de données existants, tandis que l'espace de travail est utilisé pour interagir avec l'espace de données sélectionné, dont il liste les images prises. Pour gérer vos espace de données, sélectionner *Espace de données* dans la barre de menu.



Concepts et outils associés

Instantané	Photographie du contenu d'un espace de données à un instant t.
Espace de données Référence	Ancêtre de tous les autres espaces de données, n'ayant pas de parent et ne pouvant être fusionné.
Fusion	Copie de changements fait sur un espace de données fils vers son parent, qui requiert l'arbitrage de l'utilisateur en cas de conflits.
Espace de données relationnel	Un espace de données relationnel est réservé aux modèles de données en mode relationnel. Dans ce mode, la gestion des données est d'avantage déléguée à la base de données et certaines fonctionnalités ne sont pas disponibles : par exemple, il n'est pas possible de créer une image ou un espace de données fils.

CHAPITRE 15

Actions principales**Création d'un espace de données**

Vous pouvez créer un nouvel espace de données en utilisant le bouton *Créer un espace de données* situé dans l'espace de travail. L'espace de données nouvellement créé devient un sous-espace de données de celui qui était sélectionné et le contenu de son parent est virtuellement dupliqué dans le nouvel espace de données. Les informations suivantes sont requises :

Identifiant	Identifiant unique de l'espace de données.
Propriétaire	Utilisateur possédant l'espace de données et étant autorisé à en modifier les informations et les permissions, qui n'est pas obligatoirement son créateur.
Libellé	Libellé et description associés à l'espace de données en plusieurs langues.

Edition de l'espace de données

Informations

Vous pouvez modifier les informations associées à l'espace de données à l'aide du bouton *Actions*.

Propriétaire	Personne possédant l'espace de données et étant autorisée à en modifier les informations et les permissions, qui n'est pas obligatoirement son créateur.
Documentation	Libellé et description associés à l'espace de données en plusieurs langues.
Changement de propriétaire	Modification de l'attribut <i>Propriétaire</i> par le propriétaire d'un espace de données. En son absence, seul l'administrateur a le droit d'effectuer cette modification.
Changement des permissions	Modification des permissions par le propriétaire d'un espace de données. En son absence, seul l'administrateur a le droit d'effectuer cette modification.
Politique de fusion	<p>La politique de fusion des espaces de données enfants ne s'applique qu'à la fusion initiée par un utilisateur. Elle ne s'applique aux fusions programmatiques (ex: script 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 quelque soit la validité du résultat de cette fusion. • Fusion pré-validante : un espace de données enfant peut être fusionné si et seulement si le résultat de la fusion est valide.
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".

Validation

Le contenu d'un espace de données peut être validé d'un coup en utilisant le service validation au niveau espace de données. Ce service est accessible par le bouton *Actions* dans l'espace de travail.

Note : Pour utiliser ce service, l'utilisateur doit avoir la permission de valider chaque jeu de données contenus dans l'espace de données.

Exporter une archive

Le contenu d'un espace de données peut être exporté au sein d'une archive en utilisant le service accessible via le bouton *Actions*.

Pour réaliser un export, trois informations sont requises:

- **Nom du fichier archive** : champ texte permettant de renseigner le nom souhaité.
- **Type d'export** : Obligatoire.

Le type d'export par défaut est **Le contenu complet de l'espace de données**, celui ci exporte l'ensemble des données sélectionnées dans l'archive. Si vous travaillez dans un espace de données en fait, il peut être intéressant de comparer l'espace de données avec son père et d'inclure dans l'archive les différences, sous la forme d'un delta.

Deux types d'export permettent de le réaliser : **Mises à jour avec leur contenu complet** et **Mises à jour seules** . Le premier exporte tout les données et le delta tandis que le second n'exporte les données que si une différence est définie dans le delta. La granularité d'export est le niveau table. L'écran de comparaison permet de sélectionner les différences à inclure dans le delta.

- **Jeu de données à exporter** : la table permet de sélectionner les jeux de données qui doivent être exportés. Pour chaque jeu de donnée, il est possible de choisir si les données, les permissions et les informations doivent être exportés.

Importer une archive

Le contenu d'une archive peut être importé dans un espace de données en utilisant le service accessible via le bouton *Actions*.

Si l'archive sélectionnée contient un delta, vous pouvez choisir de l'utiliser. L'écran de comparaison vous permet de sélectionner les différences du delta qui doivent être importées.

Fusion

Des changements à un espace de données peuvent être appliqués à son parent en utilisant la fonction *fusionner*, grâce au bouton *Actions*. Le processus compare les différences entre les deux espace de données à fusionner, puis demande à l'utilisateur d'arbitrer les conflits éventuels.

Pour plus de détails, sur le déroulement des fusions, lisez la page [Fusion](#).

Image

Créer une image est un moyen de sauvegarder un état exact et en lecture seule de votre espace de données à un instant t , avant d'y appliquer tout changement majeur. L'image figée de votre espace de référence devient un point de repère pour vous, au cas où vous auriez besoin d'annuler des changements.

Pour trouver comment faire une image de votre espace de données, lire la page [Image](#).

Permissions

Il est possible pour un propriétaire ou un administrateur de définir des droits d'accès à un espace de données (lecture seule, écriture ou non visible) et de les préciser action par action de manière à protéger l'espace de données et son contenu de tout accès ou modification par des utilisateurs non autorisés. Plus de précisions sur les permissions et leur gestion, peuvent être trouvées à la page [Permissions](#).

Fermeture d'un espace de données

Si vous souhaitez abandonné un espace de données sans le fusionner, ce dernier peut tout simplement être fermé par son propriétaire ou tout utilisateur habilité. Une fois fermé, plus personne ne peut y accéder. L'espace de données peut toutefois être réouvert par un administrateur, sauf si celui-ci a déjà été purgé.

Cela peut se faire à l'aide du bouton *Actions*.

CHAPITRE 16

Image

Une image est une image exacte et en lecture seule d'un espace de données à un instant t.

Création d'une image

Une image peut être obtenue en utilisant le bouton *Créer une image* situé dans l'espace de travail. Les informations suivantes sont requises.

Identifiant	Identifiant unique pour l'image.
Libellé	Libellé et description associés à l'image en différentes langues.

Information

Il est possible de modifier les informations associées à une image grâce au bouton *Actions*.

Propriétaire	Utilisateur possédant l'image et étant autorisé à en modifier les informations et les permissions, qui n'est pas obligatoirement son créateur.
Documentation	Libellé et description associés à l'image en différentes langues
Changement de propriétaire	Modification de l'attribut <i>Propriétaire</i> par le propriétaire d'une image. En son absence, seul l'administrateur a le droit d'effectuer cette modification.

Visualiser le contenu

Il est possible de visualiser le contenu d'une image à l'aide du bouton *Actions*.

Services

Plusieurs autres services sont accessibles à l'aide du bouton *Actions*.

Validation

Le contenu d'une image peut être validé directement en utilisant le service validation au niveau de l'image.

Note : pour pouvoir utiliser ce service, l'utilisateur doit avoir la permission de valider chaque jeu de données contenus dans l'image.

Comparer

Ce service permet de sélectionner un autre espace de données ou une image et de comparer leurs contenus. Un écran de comparaison, similaire à celui qui s'affiche en cas de fusion d'espace de données, résume les différences.

Export

Ce service permet d'exporter le contenu d'une image vers une archive.

Quand on exporte une archive, les informations à renseigner sont les suivantes :

- **Nom de l'archive à créer**: nom du fichier à créer
- **Jeu de données à exporter**: sélectionner les jeux de données à exporter. Pour chaque jeu de données, il est possible de décider, ce qui doit être inclus dans l'archive (données, permissions de jeux de données et/ou informations sur les jeux de données).

Fermeture d'une image

Si vous souhaitez supprimer une image, cela peut être fait en la fermant. Le propriétaire de l'image, ou un utilisateur autorisé, peuvent fermer une image, à l'aide du bouton *Actions*.

Une fois fermée, l'image devient inaccessible. L'image peut toutefois être réouverte par un administrateur, sauf si celle-ci a déjà été purgée.

CHAPITRE 17

Fusion d'un espace de données

Quand le travail dans un espace de données est terminé, vous pouvez 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. Les verrous restent pendant la durée de la fusion. C'est à dire que les contenus des deux espaces de données peuvent être lus, mais ne peuvent pas être modifiés.

Note : cette restriction sur l'espace de données parent s'applique aussi aux fusions des autres espaces de données enfants et non seulement aux modifications directes.

2. les changements dans l'espace de données qui ont été faits 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 une fusion d'un espace de données dans 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, vous devez décider quels changements survenus dans l'espace de données enfant (source) doivent être fusionnés dans l'espace de données parent (cible).

Pendant la fusion, un écran de comparaison récapitule tous les changements qui doivent 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 fusion. Vous pouvez désélectionner les changements pour les exclure de la fusion. Vous pouvez voir les changements relatifs aux différentes étendues en sélectionnant des éléments dans le panneau de navigation.

La *fusion à trois points* implique l'espace de données courant, l'image initiale et l'espace de données parent. Elle permet de détecter les conflits. Parfois, les données ont 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 décrits dans une table. 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 intitulé *Marquer les différences comme revues* pour indiquer que vous avez vérifié les changements dans l'étendue actuelle. Tous les changements doivent être revus pour effectuer la fusion.

Types de modifications

Le processus de fusion considère comme 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 trouvés. Des conflits peuvent survenir quand une étendue contient des modifications dans l'espace de données source et l'espace de données cible.

Les conflits sont catégorisés comme suit :

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

Finalisation d'une fusion

Lorsque vous avez vérifié tous les changements et décidé lesquels sont à inclure dans le résultat de la fusion, 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. L'administrateur de l'espace de données parent peut configurer la politique de fusion pour que les fusions avec ses espaces de données enfants soient finalisées uniquement si le résultat n'a pas d'erreur de validation.

Si la politique de fusion par défaut est utilisée, la fusion est achevée quel que soit l'état de validation de l'espace de données résultant.

Si la politique de fusion pré-validante est utilisée, un espace de données dédié est d'abord créé pour contenir 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 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>" - seront accessibles. 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 récupère l'espace de données enfant dans l'état d'avant fusion.
- **Poursuivre la fusion** : cette action permet de retenter une 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, vous pouvez interrompre la finalisation des fusions avec ses espaces de données enfant à travers de l'interface utilisateur si le résultat contient des erreurs de validation. Pour cela, 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 enfant avec l'espace de données parent ainsi paramétré.

Note

En mode 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, donc ignorée pour les tâches automatiques des workflows de données.

Voir aussi : [Politique de fusion](#)

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, vous pouvez cliquer sur le bouton **Annuler** afin d'abandonner l'opération.

Si vous naviguez vers une autre page pendant une fusion, la fusion sera abandonnée, mais les verrous sur l'espace de données parent et l'espace de données enfant resteront. 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 ensuite 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, seulement l'espace de données parent sera déverrouillé, donc vous devrez également effectuer un déverrouillage de l'espace de données enfant.

CHAPITRE 18

Permissions

Les permissions de l'espace de données peuvent être accédées à l'aide du bouton *Actions* dans l'espace de travail.

Une permission est toujours attachée à un profil.

Permissions sur un espace de données

Permissions générales

- **Identifiant de l'espace de données** : Indique l'espace de données sur lequel les permissions vont être appliquées.
- **Sélection du profil** : indique le profil affecté par la permission définie.
- **Restriction d'accès** : indique si la permission définir ici restreint celles définies sur pour d'autres profils. Voir aussi la page [Restriction d'accès](#).
- **Permissions d'accès à l'espace de données** : Indique la permission d'accès globale sur l'espace de données (lecture seule, écriture ou non visible). Voir ci-dessous.

Lecture seule

- Permet de visualiser l'espace de données et son image et de voir les espaces de données fils, 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 vous en autorise l'accès.

Ecriture

- Permet de voir l'espace de données.
- Permet d'accéder aux jeux de données selon les droits que l'on possède sur eux.

Non visible

- Ni l'espace de données, ni ses images ne peuvent être vus.
 - A partir d'un espace de données fils, on peut voir l'espace de données courant sans toutefois pouvoir le sélectionner.
 - Absence d'accès au contenu de l'espace de données.
 - Aucune action ne peut être effectuée sur l'espace de données.
-

Actions permissibles à la carte

- **Créer un espace de données fils** : indique si le profil peut créer un espace de données fils.
- **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 accéder au service export.
- **Importer une archive** : indique si le profil peut accéder au service import.
- **Fermer un espace de données** : indique si le profil peut fermer un espace de données.
- **Fermer une image** : indique si le profil peut fermer l'image d'un espace de données.
- **Droits sur les services** : spécifie les permissions d'accès aux services. Un service n'est pas accessible à un profil s'il est barré.
- **Permissions d'un espace de données fils à la création** : spécifie les permissions d'accès, qui seront présentes dans un espace de données fils nouvellement créé.

Permissions sur les images

Permissions sur l'image initiale d'un espace de données

Sont appliquées les permissions définies pour l'espace de données.

Permissions sur une image d'un espace de données

Sont appliquées les permissions définies pour l'espace de données.

Jeux de données

CHAPITRE 19

Introduction aux jeux de données

Une fois que vos données de référence ont été modélisées, et ces modèles publiés, vous pouvez gérer vos données de référence au travers des actions suivantes :

- créer un nouveau jeu de données dans un data space (voir [création](#)),
- sélectionner un espace de données où travailler, un jeux de données dans celui-ci, puis consulter l'arborescence des groupes, tables et champs (voir [actions](#)),
- lire, créer, modifier ou supprimer des enregistrements dans une table d'un jeu de données (voir [tables](#)),
- définir des vues personnalisées et/ou des hiérarchies sur les tables (voir [modes de vue avancés](#)),
- commencer à travailler à partir d'un jeu de données existant en en copiant les données (voir [héritage](#)),
- définir qui peut, ou ne peut pas, accéder à certaines données (voir [permissions](#)).

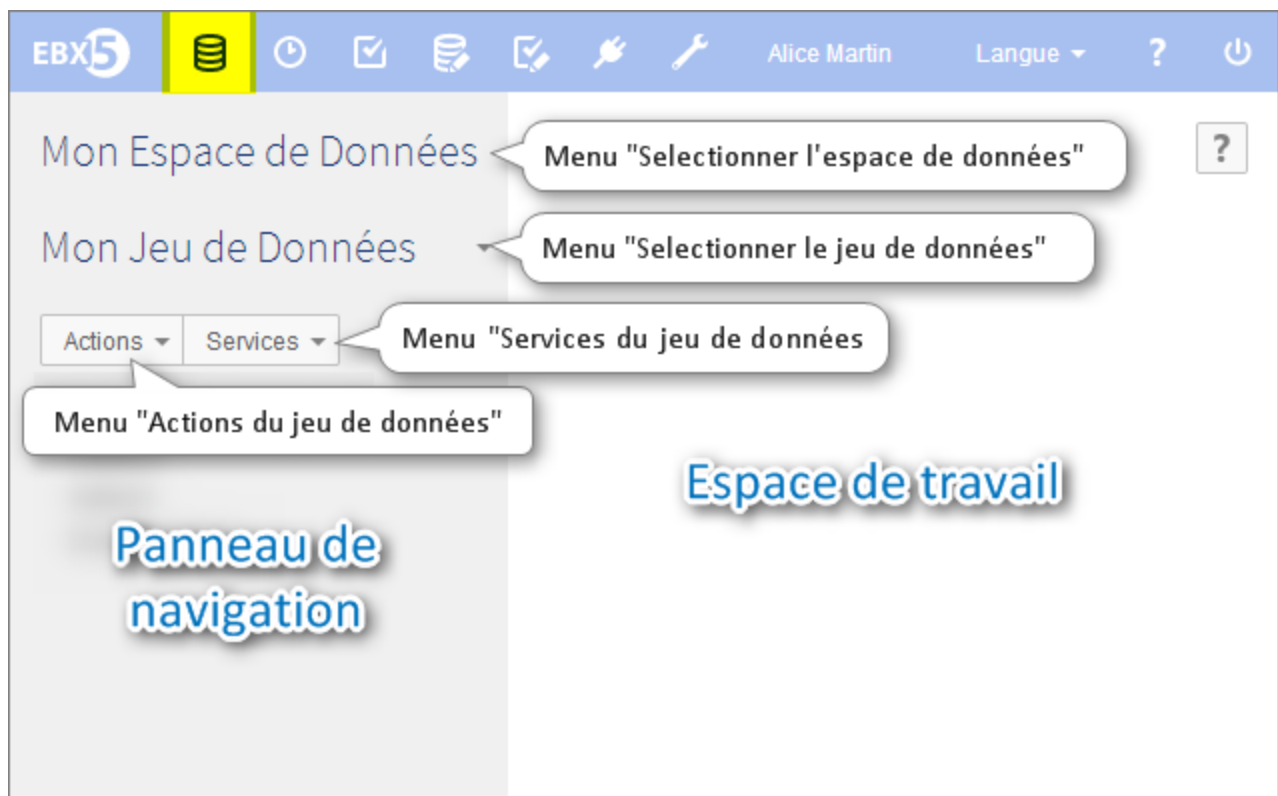
Un jeu de données correspond aux données présentées sous forme de tables ou hiérarchies, et pouvant être filtrées en vue multi-critères. L'accès au contenu d'un jeu de données peut être restreint à l'aide de règles de permissions.

Pour une meilleure compréhension de ces concepts, prenez la liberté de regarder notre [glossaire](#).

Vue d'ensemble d'un jeu de données

Modalités d'accès dans l'interface

Pour interagir avec vos données, sélectionnez *Données* dans la barre de menu. Puis ouvrez un espace de données dans la partie haute du panneau de navigation.



Les utilisateurs peuvent sélectionner un jeu de données en utilisant le panneau de navigation. Deux boutons s'affichent, le premier permet de sélectionner un espace de données, tandis que le second permet de sélectionner un jeu de données. Une fois le jeu de données sélectionné, sa structure apparaît dans le panneau de navigation, et peut être utilisée pour sélectionner un groupe ou une table. Les valeurs de ses éléments (tables, champs, enregistrements) s'affichent alors dans l'espace de travail.

Concepts et outils associés

Une fois dans la section *Données*, vous allez aborder les outils et les notions suivantes :

Espace de données	Conteneur de données, dont le contenu peut être mis à jour en toute isolation des autres éléments à proximité.
Jeu de données	Groupe de données prédéfini dans le modèle de données, et ayant une utilité ou un but commun
Arborescence	Moyen de visualiser le contenu d'un jeu de données dans le panneau de navigation
Table cible	Table étudiée, dont on souhaite visualiser les dépendances aux autres tables. C'est le dernier et le plus spécifique des niveaux de la hiérarchie, à l'image des feuilles d'un arbre.
Enregistrement	Groupe de champs formant une unité d'information entrée par un utilisateur dans un certain jeu de données, apparaissant comme le contenu d'un rang dans une table
Hiérarchie	Arborescence représentant l'enchaînement des dépendances entre les tables. Elle peut être équilibrée, déséquilibrée, irrégulière ou en réseau.
Relation récursive	Occurrence d'un lien de dépendance entre deux entités du même niveau de dimension
Dimension	Axe possible d'analyse d'une table cible, incluant différents niveaux de dimension (exemple : produits, familles, catégories, etc.)

Voir aussi :

- [Modèles de données](#)
- [Espaces de données](#)
- [Modèles de workflow](#)
- [Workflows de données](#)
- [Service de données](#)

CHAPITRE 20

Actions principales

Création d'un jeu de données

Un jeu de données peut être créé à l'aide du bouton *Créer un jeu de données*, accessible à partir du sélecteur de jeu de données. S'il n'existe aucun jeu de données dans l'espace de données sélectionné, le bouton s'affiche directement dans le panneau de navigation. Un assistant vous permet alors de créer le jeu de données.

Information

Les informations associées au jeu de données peuvent être éditées à l'aide du bouton *Actions* dans le panneau de navigation. Les informations éditables sont les suivantes :

- **Propriétaire** : Utilisateur pouvant éditer ses informations et définir des règles de permissions sur le jeu de données, qui n'est pas obligatoirement son créateur.
- **Documentation** : Libellé et description associés au jeu de données dans différentes langues.
- **Activé** : Activation du jeu de données nécessaire à la sélection de règles de validation.

Édition du jeu de données

Navigation dans le jeu de données

Dans un jeu de données, vous pouvez trouver des tables, des champs et des groupes de l'un et de l'autre. Pour y accéder, utilisez l'arborescence du panneau de navigation et cliquez sur le jeu de données qui vous intéresse. Son contenu s'affiche dans l'espace de travail.

Paramétrage des permissions

Pour savoir comment personnaliser les permissions pour chaque utilisateur potentiel, lisez attentivement la page [permissions](#).

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, vous pouvez cliquer sur le bouton **Revalider** pour mettre à jour ce rapport, ou vous pouvez cliquer sur le bouton **Réinitialiser le rapport de validation** pour supprimer tous les messages de validation actuellement associés au jeu de données afin de pouvoir relancer une validation complète.

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

Table de jeux de données

Le contenu d'une table s'affiche dans l'espace de travail, qui est aussi l'endroit où l'utilisateur peut interagir avec ce contenu.

Édition d'un enregistrement

Créer	Un nouvel enregistrement peut être créé à l'aide du bouton "+" situé en haut à gauche de la table. Un formulaire s'affiche, permettant d'entrer des données. Les données obligatoires sont repérées par une astérisque rouge.
Éditer	Un enregistrement peut être édité en double-cliquant dessus. Le formulaire s'affichant permet d'éditer l'enregistrement, tandis que le bouton <i>Rétablir</i> permet de recharger le formulaire sans soumettre aucun des changements effectués.
Dupliquer	<p>Un enregistrement qui a été sélectionné à l'aide d'une case à cocher peut être dupliqué grâce au bouton <i>Actions</i>.</p> <p>Un formulaire apparaît avec les valeurs pré-remplies à l'image de l'enregistrement copié. La clé primaire doit ensuite être modifiée pour pouvoir créer ce nouvel enregistrement, sauf si celle-ci est générée (par exemple une valeur auto-incrémentée).</p>
Supprimer	Un ou plusieurs enregistrements, ayant été sélectionnés à l'aide des cases à cocher, peuvent être supprimés grâce au bouton <i>Actions</i> .
Comparer	<p>Deux enregistrements, ayant été sélectionnés à l'aide des cases à cocher, peuvent être comparés à l'aide du bouton <i>Actions</i>.</p> <p>Note : Le contenu des noeuds terminaux complexes, comme les listes agrégées ou les attributs utilisateurs, ne sont pas comparés pendant ce processus. Le service de comparaison ignore toutes les différences entre les valeurs des noeuds terminaux complexes dans les enregistrements.</p>

Import/export d'enregistrement

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

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

Voir aussi :

- [Services CSV](#)
- [Services XML](#)


Tri des données

Un critère de tri contrôle l'ordre dans lequel les enregistrements sont présentés. Ils peuvent être définis grâce au bouton *Vue*. L'ordre par défaut est par clé primaire ascendante. L'option *Rétablir* permet d'annuler toute modification.

Il est défini par un nom de colonne et un ordre (ascendant, de la valeur la plus basse à la plus élevée, ou descendant, de la valeur la plus élevée à la plus faible). Un critère de tri peut être ajouté et retiré à l'aide des liens *Ajouter* et *Retirer*. L'ordre de classement d'un critère peut être modifié en cliquant sur les liens *ascendant* ou *descendant*.

L'ordre entre les critères peut être ajusté à l'aide du bouton situé à droite de la liste de critères.

Recherche et filtrage des données

Il existe des outils spécifiques dédiés à 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. Dans le reste de cette section, les différents types de filtres disponibles sont détaillés.

Note

Appliquer une vue personnalisée reinitialisera et retirera tous les filtres appliqués.

Recherche typée

En mode simple, la recherche typée permet d'ajouter des critères contextualisés sur un ou plusieurs champs. Lors de l'ajout d'un critère les opérateurs proposés seront pertinents vis-à-vis du 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.

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 * (n'importe quel texte) ou le point d'interrogation ? (n'importe quel caractère). Pour des raisons de performance, leur utilisation est restreinte à un seul de ces caractères 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 :

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

Option *Sensible à la casse* : la recherche tient compte de la casse (majuscules et minuscules sont distinctes).

Filtre par messages de validation

Le filtre par messages de validation permet de voir les enregistrements en fonction de leur statut dans la dernière validation effectuée. Vous pouvez voir les enregistrements avec les niveaux 'Erreur', 'Avertissement' et 'Information'.

Note

Cette recherche peut s'appliquer seulement sur les enregistrements de la table qui ont déjà été validés ; pour cela, il faut 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.

Vues personnalisées

Vous pouvez personnaliser la présentation d'une table en appliquant ou en définissant des vues. En utilisant le bouton *Vue*, il est possible de sélectionner une vue personnalisée existante ou d'en définir une nouvelle. Une fois une vue appliquée, elle peut être définie comme la *vue par défaut* pour la table. En cliquant sur *Enlever toutes les vues*, la vue par défaut est rétablie.

Voir aussi : [*Vues personnalisées*](#)

CHAPITRE 21

Vues personnalisées

Il y a deux modes avancés de visualisation :

- le premier est spécifique aux tables et vous permet de filtrer les informations selon des critères donnés ([vue taublaire simple](#)),
- le deuxième vous permet d'établir des liens et d'organiser des informations issues de différentes tables ([hiérarchies](#)).

Une vue personnalisée peut être créée via le bouton *Vue*.

Publication de vue

Une vue peut être publiée afin de la rendre accessible à tous les autres utilisateurs grâce aux *composants web*, tâches utilisateurs du workflow, ou data services. Il s'agit d'une publication technique, elle ne donnera pas accès à cette vue à tous les utilisateurs via l'interface graphique (voir *Profils autorisés* ci dessous).

Description d'une vue

Ce formulaire permet de spécifier les informations liées à la vue personnalisée communes à tous les modes.

Propriétaire	Nom du propriétaire de la vue personnalisée, qui peut à ce titre la gérer et la modifier.
Documentation	Libellé et description associés à la vue personnalisée dans différentes langues.
Profils autorisés	Détail des profils ayant le droit d'utiliser la vue personnalisée.
Mode de la vue	Mode de la vue personnalisée (voir ci dessus).

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.

Critères de recherche

Ce champs permet de définir des critères, qui vont être utilisés pour filtrer les enregistrements.

Voir aussi : [Documentation sur l'éditeur de critères](#)

Critères de tri

Ce champs permet de définir des critères, qui vont être utilisés pour trier les enregistrements.

Affichage des colonnes

Ce champs permet de sélectionner, à l'aide des cases à cocher, les colonnes de la table à afficher dans la vue personnalisée.

Hiérarchies

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

Dimension d'une hiérarchie

Une dimension est ce qui définit les différentes dépendances dans la hiérarchie (exemple : produits par catégories). En démarrant d'une table cible, sélectionner les liens de dépendances pas à pas jusqu'au dernier.

Chaque lien de dépendance établit devient un nouveau niveau de dimension.

Configuration d'une hiérarchie

Ce formulaire permet de configurer les niveaux de hiérarchie. Pour chaque niveau, vous pouvez configurer les libellés et spécifier les critères de filtrage des enregistrements.

Libellés

Dans l'arbre, les enregistrements sont nommés par des libellés, écrit dans différentes langues. Les champs d'enregistrement peuvent être aisément insérés dans un libellé en utilisant l'assistant (utiliser le bouton à droite du champ pour y accéder).

Filtre

Ce formulaire permet de définir des critères, qui pourront être utilisés pour filtrer les enregistrements.

Voir aussi : [*Documentation sur l'éditeur de critères*](#)

Ordonnancement

Il est possible de spécifier un champ d'ordonnancement qui permettra à l'utilisateur de définir l'ordre des noeuds enfants. Un champ d'ordonnancement doit être un entier et être en mode caché dans le modèle de données.

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

A défaut d'un noeud d'ordonnancement, les noeuds enfants sont triés selon l'ordre alphabétique des libellés des noeuds.

CHAPITRE 22

Héritage


Habituellement les gens créent un jeu de données à partir d'un modèle de données. EBX5 vous permet de créer des jeux de données additionnels, branchés à partir d'un jeu de données racine. Ces jeux de données enfants héritent de leur parents. Plusieurs niveaux d'héritage peuvent être créés.

Cela peut être utilisé pour adapter des données de référence à divers contextes, comme des zones géographiques et/ou des pays.

Note : Le comportement standard est d'interdire l'héritage de jeux de données. Il est nécessaire d'activer explicitement cette fonction quand vous créez votre modèle de données.

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

Arbre des jeux de données

Une fois le jeu de données racine créé, un jeu de données fils peut être créé à l'aide du bouton  situé dans l'écran de sélection des jeux de données du panneau de navigation. On demande ensuite à l'utilisateur de fournir un nom unique pour le jeu créé. Il peut aussi, s'il le souhaite, lui donner un libellé local optionnel et une description.


Note :

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

Héritage de valeurs

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


Les valeurs surchargées utilisent le style par défaut tandis que les valeurs héritées sont signalées par un repère dans le coin en haut à gauche de la cellule.

Le bouton  peut être utilisé pour indiquer, si une valeur est héritée ou surchargée.

Héritage d'enregistrement

Une table dans un jeu de données fils va hériter de l'enregistrement de la table située chez son ancêtre. Il est possible d'éditer ou de supprimer ces enregistrements, de nouveaux enregistrements peuvent également être créés et seront hérités par le jeu de données fils. Plusieurs états sont définis pour différencier ces enregistrements.

Racine	Un enregistrement racine est un enregistrement créé dans le jeu de données courant, qui n'existe pas chez ses jeux de données ancêtre. Il sera hérité par les jeux de données fils.
Hérité	Un enregistrement hérité est défini dans un des jeux de données ancêtre 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 fils.
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 fils.

Le bouton  indique, que l'enregistrement est hérité. Il peut également être utilisé pour changer son état en "surchargé" ou "hérité". Veuillez noter que c'est le même bouton, qui permet de changer le statut d'un enregistrement occulté en "hérité".

Les tables suivantes résume, ce qui se produit quand on crée, édite ou supprime un enregistrement dépendant de l'état initial.

Etat \ Opération	Créer	Editer la valeur	Supprimer
Racine	Création normale d'un enregistrement. L'enregistrement nouvellement créé sera hérité par ses jeux de données fils.	Edition normale d'un enregistrement. Les nouvelles valeurs seront héritées par ses jeux de données fils.	Suppression normale d'un enregistrement. L'enregistrement va disparaître du jeu de données hérité.
Hérité	Si un enregistrement est créé à l'aide de la clé primaire d'un enregistrement hérité existant, l'état de l'enregistrement sera changé à "surchargé" et sa valeur modifiée selon celle soumise à sa création.	Un enregistrement hérité doit être déclaré comme surchargé pour éditer cette valeur.	Supprimer un enregistrement hérité le fait passer à l'état "occulté".
Surchargé	Pas applicable. Il est impossible de créer un nouvel enregistrement si la clé primaire est déjà utilisée.	Un enregistrement surchargé peut être retourné à l'état hérité, mais sa valeur spécifique sera perdue. Les valeurs de l'enregistrement surchargé peuvent être changée à "hérité" ou peuvent être spécifiées.	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 sera changé à "surchargé" et sa valeur modifiée d'après celle soumise à la création.	Un enregistrement occulté ne peut plus être édité.	Pas applicable. Un enregistrement occulté est déjà considéré comme supprimé et à ce titre ne peut pas être supprimer une deuxième fois.

CHAPITRE 23

Permissions

Les permissions de jeux de données peuvent être accédées à l'aide du bouton *Actions* dans le panneau de navigation.

Les permissions sont toujours données via un profil.

Profil

Indique le profil affecté par une permission.

Restriction d'accès

Indique si les permissions définies ici restreignent celles définies pour d'autres profils. Pour plus de détails, voir [Restriction d'accès](#).

Actions sur les jeux de données

Cette section spécifie les permissions des actions sur les jeux de données.

Créer un jeu de données fils	Indique si un profil peut créer un jeu de données fils. L'héritage doit aussi être activé dans le modèle de données.
Dupliquer un jeu de données	Indique si un profil peut dupliquer un jeu de données.
Supprimer un jeu de données	Indique si le profil peut supprimer le jeu de données.
Activer/Inactiver un jeu de données	Indique si un profil peut modifier la propriété <i>Activé</i> dans les informations du jeu de données.
Créer une vue	Indique si un profil peut créer des vues personnalisées et des hiérarchies.

Droits sur tables

Cette section spécifie les permissions par défaut pour toutes les tables. Des permissions spécifiques peuvent aussi être définies pour une table en cliquant sur le bouton *Ajouter une occurrence* puis en sélectionnant une des options suivantes :

Créer un nouvel enregistrement	Indique si un profil peut créer des enregistrements dans une table.
Remplace un enregistrement hérité	Indique si un profil peut remplacer des enregistrements hérités dans une table. Cette permission est utile quand on utilise l'héritage de jeu de données.
Occulte un enregistrement hérité	Indique si un profil peut occulter des enregistrements hérités dans une table. Cette permission est utile quand on utilise l'héritage de jeu de données.
Supprimer un enregistrement	Indique si un profil peut supprimer des enregistrements dans une table.

Droits sur valeurs

Cette section 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 allouée à un élément.

Le selecteur de droits spécifique permet d'attribuer des permissions d'accès spécifiques pour un élément. Les boutons *Non visible*, *Lecture seule* et *Ecriture* règle les permissions d'accès correspondant à l'élément sélectionné.

Il est possible de retirer une permission spécifique d'accès en utilisant le bouton *(défaut)*.

Droits sur les services

Cette section spécifie les permissions d'accès sur les services. Un service n'est pas accessible à un profil s'il est barré.

Modèles de workflow

CHAPITRE 24

Introduction aux modèles de workflow

Le travail collaboratif est un moyen efficace de produire, mettre à jour, fusionner et valider des données dans une entreprise. Cependant il n'est pas toujours facile d'obtenir que des gens de différents endroits et ayant des compétences différentes, travaillent ensemble pour remplir un objectif à une échéance commune.

En cela, vous trouverez l'outil de modélisation de workflow bien utile, car il va vous permettre de définir des processus de gestion des données impliquant vos collaborateurs. Pour ce faire, vous allez avoir besoin de :

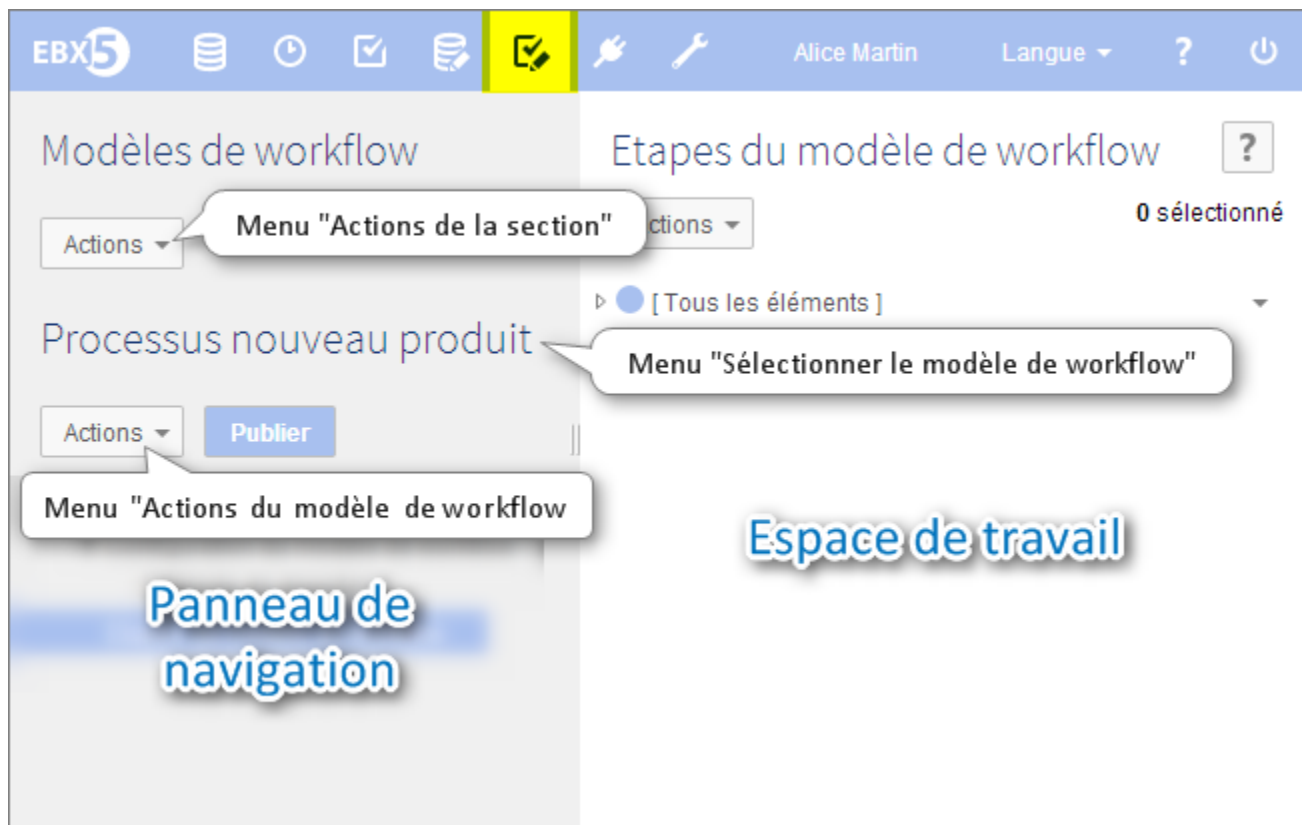
- définir des tâches réalisables par un utilisateur (voir [tâche utilisateur](#)) ou automatiquement par le système (voir [tâche autonome](#)),
- spécifier les responsabilités de chacun (voir [création](#)),
- d'envoyer des courriers électroniques de notification à des collègues, dont la participation est requise (voir [notification](#)),
- de connaître les limites de votre modèle; ce qui peut ou ne peut pas être fait (voir [limitations](#)),
- de publier votre modèle en tant que workflow pour pouvoir vous en servir (voir [publication](#)).

Un modèle de workflow définit les tâches à réaliser et les responsabilités impliquées. Il peut ensuite être publié en tant que publication de workflow. Il s'agit d'une succession de deux types de tâches : *tâches autonomes* et *tâches utilisateurs*, avec la possibilité d'ajouter des fourches conditionnelles entre les tâches.

Pour trouver le sens d'un mot, vous pouvez le rechercher dans notre [glossaire](#).

Vue d'ensemble d'un modèle de workflow

Modalités d'accès dans l'interface



Concepts et outils associés

Tâche autonome	Aucun utilisateur n'est impliqué dans ce type de tâche. Cela peut être, par exemple, une fusion automatique, la création d'une image d'un espace de données, etc.
Tâche utilisateur	Cela implique au moins un utilisateur et éventuellement plusieurs, qui doivent effectuer des bons de travail.
Bon de travail	Tâche unitaire réalisée par l'utilisateur, auquel elle a été allouée, et dont l'exécution permet au workflow de progresser.
Fourche conditionnelle/<i>Condition</i>	Elle décide à partir du résultat des tâches précédentes, quelle route doit être prise par un workflow. Cet embranchement comprends deux dérivations. Par exemple, quelqu'un peut continuer un workflow normalement, tandis que quelqu'un d'autre retourne à une tâche précédente; ou il peut y avoir deux routes différentes en parallèle.
Contexte des données	Il s'agit d'une variable hébergeant temporairement des données entrée/sortie liées à l'exécution d'un workflow. Son but est de faciliter le transfert d'informations clés d'une étape à l'autre (par exemple, le nom d'un espace de données créé à l'étape 1 et étant réutilisé à l'étape 2 dans un autre but).

CHAPITRE 25

Actions principales

Modélisation d'un workflow

Conception

Tout d'abord, vous devez penser à quel type d'évolution vos données vont régulièrement être soumises, et combien de personnes seront impliquées dans ces évolutions.

Ensuite, vous devrez schématiser tout cela et le traduire en un modèle de workflow dans EBX5.

Pour trouver comment procéder dans l'interface, lisez la section [création](#)).

Validation

Une fois le premier jet réalisé dans l'interface, votre modèle va devoir être validé, de manière à vérifier, qu'il n'y a pas d'erreurs dans le code associé. S'il y en a, elles devront être résolues avant que vous puissiez publier et utiliser votre modèle.

Publication

Une fois validé, votre modèle de workflow est prêt pour la publication. En le publiant, vous en générez une image figée, qui devient disponible à l'usage dans la section workflow.

Limitations

Les fonctionnalités suivantes ne sont pas supportées :

- **Tâches parallèles**, deux routes, ou plus, s'exécutent simultanément.
- **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.

Modèles de message génériques

Des courriers électroniques de notification peuvent être envoyés pour informer des utilisateurs spécifiques d'un événement donné pendant l'exécution d'un workflow.

Des modèles de message peuvent être définis et réutilisés dans chaque modèle de workflow en sélectionnant l'entrée "Modèles de message" dans le menu "Actions" global des Modèles de Workflow.

Ces modèles de message sont partagés par tous les modèles de workflow et 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.

Quand vous créez un nouveau message type, deux champs sont requis :

- **Libellé & Description:** spécifie le libellé et la description associée à ce message type en différentes langues.
- **Message:** spécifie l'objet de l'email et son corps de texte en différentes langues.

Le message peut être amélioré à l'aide de variables type contexte de données telles que : `${nom.variable}` . De plus, les variables système ci-dessous sont disponibles :

Syntaxe des variables	Signification
system.time	Heure système.
system.date	Date système.
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.currentStep.label	Libellé de l'étape courante (tâche autonome, condition ou tâche utilisateur).
workflow.currentStep.description	Description de l'étape courante (tâche autonome, condition ou tâche utilisateur).

Voici un exemple de message : " *Aujourd'hui à \${system.time}, un nouveau bon de travail vous a été proposé.*"

Cela pourra donner le courrier électronique suivant : " *Aujourd'hui à 15:19, un nouveau bon de travail vous a été proposé.* "

Evolution d'un modèle de workflow

Edition

Des améliorations peuvent être apportées au modèle existant. Toutefois il est conseillé, quand vous le publiez, de lui trouver un nom le différenciant de ses prédécesseurs.

Historique

L'historique des images d'un modèle de workflow peut être géré en utilisant la fonctionnalité d'historique accessible en utilisant le bouton *Actions*.

L'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*.

Suppression

Un modèle peut être supprimé. Cependant toute version publiée auparavant reste accessible dans la section workflow. De plus, si vous recréez un modèle de workflow avec un nom identique, un message vous demandera de confirmer si vous souhaitez remplacer le précédent modèle.

CHAPITRE 26

Modélisation du workflow

Création

Un modèle de workflow peut être créé à partir de la section *Modélisation / Modèle de workflow*. L'assistant de création requiert seulement un nom. Ce nom doit être unique, de manière à identifier chaque modèle de workflow spécifiquement.

Dès lors, le modèle de workflow est créé et les étapes du modèle de workflow sont initialisées par la présence d'une transition initiale. L'objet du modèle de workflow est de décrire l'enchaînement des étapes au delà de cette transition initiale.

Héritage

Il est possible d'utiliser les mécanismes d'héritage avec les limitations suivantes :

- impossibilité d'ajouter des étapes en début ou fin ;
- impossibilité d'insérer des étapes ;
- impossibilité de modifier des liens entre étapes.

Étapes

Un modèle de workflow définit des étapes correspondant à des opérations et des conditions. Les sections suivantes introduisent les différents types d'étape.

Conditions

Condition de la bibliothèque

En outre, un libellé et une description peuvent être spécifiés en différentes langues.

EBX5 met à disposition des conditions prédéfinies, grâce auxquelles vous pouvez vérifier :

- si des données sont valides (niveau espace de données, jeu de données ou table) ;
- si un espace de données a été modifié ;
- quelle était la dernière tâche utilisateur acceptée ;
- si deux valeurs sont égales ;
- si une valeur est nulle ou vide ;

Certaines conditions sont marquées comme "obsolètes" car elles ne sont pas compatibles avec I18N. Il est recommandé d'utiliser les nouvelles conditions compatibles avec I18N.

Tâche utilisateur

La modélisation d'une tâche utilisateur fait l'objet d'un [chapitre](#) particulier.

Tâche autonome

Tâche autonome de la bibliothèque

Par ailleurs, un libellé et une description peuvent être spécifiés en différentes langues.

EBX5 inclut des tâches autonomes prédéfinies, comme :

- créer un espace de données,
- fermer un espace de données,
- créer une image,
- fusionner un espace de données,
- importer une archive,
- envoyer un courrier électronique.

Certaines tâches autonomes sont marquées comme "obsolètes" car elles ne sont pas compatibles avec I18N. Il est recommandé d'utiliser les nouvelles tâches compatibles avec I18N.

Contexte des données

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

Quelques paramètres ajustables

Informations

Les informations d'un modèle de données peuvent être éditées à l'aide du bouton *Actions* dans le panneau de navigation. Les informations éditables sont les suivantes :

Champs éditables	Contenu requis
Propriétaire	Personne pouvant, en tant que propriétaire, éditer les informations du modèle de workflow et y définir des règles de permissions.
Documentation locale	Libellé et description associés au modèle de workflow en différentes langues.
Activation	Un modèle de workflow doit être activé pour pouvoir être publié.

Paramétrage spécifique au modèle

La configuration d'un modèle de workflow est accessible depuis le panneau de navigation. Les propriétés principales sont les suivantes :

Options de configuration	Contenu requis
Notification de démarrage	Liste des profils qui doivent recevoir une notification (choisie dans la liste des modèles de messages) quand un workflow démarre.
Notification de fin	Liste des profils qui doivent recevoir une notification (choisie dans la liste des modèles de messages), quand un workflow se termine.
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. Voir Priorité de bons de travail 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>
Permissions	Définit les permissions pour différentes tâches du workflow (voir section ci-dessous).
Permissions programmatiques	Définit le composant gérant les permissions. S'il est défini, il remplace l'ensemble des permissions ci-dessus.

Permission sur une publication de workflow

Pour trouver comment définir des permissions sur une publication de workflow, voir ci-dessous :

Permissions	Définition des profils autorisés
Administration de workflow	Permet de réaliser des tâches administratives sur une publication de workflow basée sur un modèle de workflow. L'autorisation de tâches spécifiques peut être définie à l'aide du lien <i>Voir la configuration avancée</i> .
Gestion des allocations	Permet d'allouer, désallouer ou réallouer des bons de travail à des utilisateurs. Une autorisation pour une opération spécifique peut être définie à l'aide du lien <i>Voir la configuration avancée</i> .
Lancement de workflow	Permet de lancer un workflow.
Monitoring de workflow	Permet de voir un workflow en cours, même s'il n'est pas directement concerné par les bons de travail courants du workflow. Un tel utilisateur peut également voir l'historique d'un workflow terminé.

Note : Un utilisateur, qui n'a aucun privilège particulier attribué, pourra voir un bon de travail uniquement, si celui-ci lui est proposé ou personnellement alloué.

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

CHAPITRE 27

Tâche utilisateur

Libellé et description

Le libellé et la description de la tâche utilisateur peuvent être spécifiés en différentes langues, afin de renseigner l'utilisateur sur la nature et le contenu de celle-ci

Définition

Profils


Les profils designent les rôles et les utilisateurs auxquels la tâche utilisateur est destinée. Un bon de travail est créé pour chaque profil spécifié.

Si un profil se rapporte à un utilisateur au lieu d'un rôle, le bon de travail lui est directement alloué.

Service à appeler

EBX5 propose nativement les services suivants :

- Accès à un contenu particulier (data space, data set, table, record, etc.).
- Créer un nouvel enregistrement.
- Valider un espace de données, une image ou un jeu de données.
- Fusionner un espace de données.
- Comparer les contenus.
- Exporter les données d'une table vers des fichiers XML.
- Exporter les données d'une table vers des fichiers CSV.
- Importer des données d'un fichier XML vers une table.
- Importer des données d'un fichier CSV vers une table.

Chaque service requiert des paramètres spécifiques ; Un assistant permet de sélectionner des espaces de données, des images ou des valeurs du contexte des données : .

Voir aussi : [Services prédéfinis de EBX5](#)

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 d'un 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".

Tolérance au 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 suivant tolèrent automatiquement tous les rejets :

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

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 certaine tâche 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.

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 faux.

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](#)

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 passée et si les bons de travail associés n'ont pas été complétés, un courrier électronique spécifique est envoyé aux utilisateurs concernés. Ce courrier électronique va alors être réenvoyé 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.

CHAPITRE 28

Publication d'un modèle de workflow

Les modèles de workflow deviennent exécutables uniquement après publication.

Cela est réalisé en créant une image, où la publication de workflow sera stockée. Ce mécanisme assure la stabilité d'une publication de workflow pendant l'exécution de workflow associés.

Une publication de modèle de workflow peut être accédé à l'aide du bouton *Publier* dans le panneau de navigation.

Etape de publication

Sélection d'un modèle de workflow

Il est possible de publier plusieurs modèles de workflow en une fois. Quand le service de publication est appelé, les modèles de workflow existant s'affichent. L'utilisateur doit sélectionner le ou les modèles de workflow, qu'il souhaite publier.

Note : un modèle de workflow doit être activé pour pouvoir être publié.

Informations sur les images et les publications de workflow

Un libellé et une description peuvent être spécifiés pour l'image à créer. Le libellé par défaut est la date et l'heure de publication et la description par défaut indique, qui a publié le modèle de workflow.

Pour chaque modèle de workflow sélectionné, le nom de publication doit être renseigné et unique. quand un modèle de workflow a déjà été publié, il est possible de le mettre à jour. Les noms des publication de workflow disponibles associées au même modèle de workflow s'affiche. Il est possible d'en sélectionner une pour réutiliser une publication de workflow existante. Auquel cas, l'ancienne publication de workflow devient inaccessible.

Workflows de données

CHAPITRE 29

Introduction aux workflows de données

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*](#)
- [*Lancement et monitoring de workflows de données*](#)
- [*Administration de workflows de données*](#)
- [*Permission sur une publication de workflow*](#)

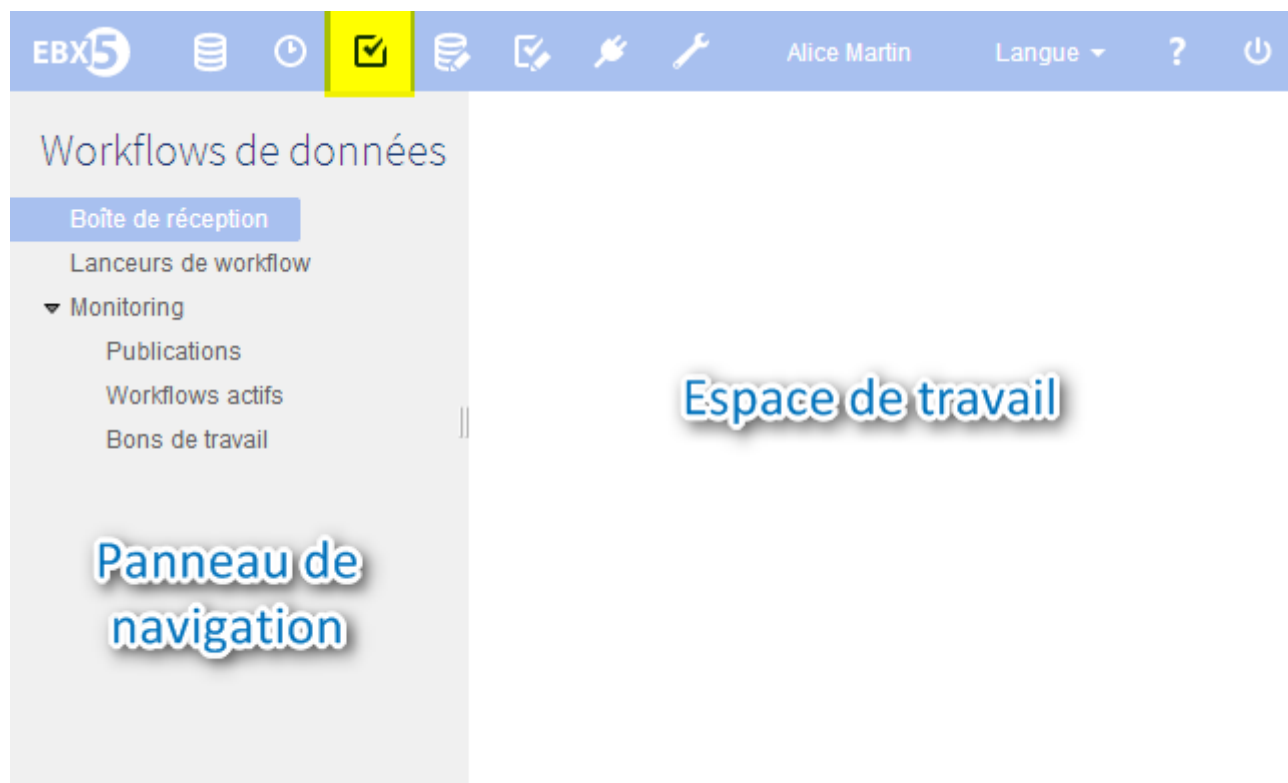
Concepts apparentés : [*Modèles de workflow*](#)

CHAPITRE 30

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

Navigation dans l'interface utilisateur

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

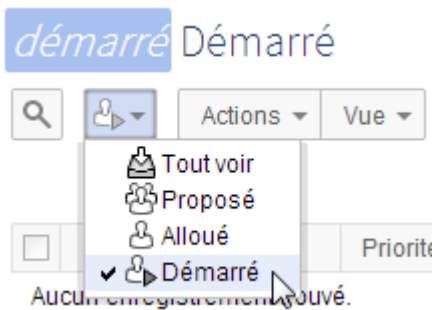


Le panneau de navigation est composé des sections suivantes :


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 travaux aux utilisateurs ou rôles.
Workflows terminés	Workflows de données qui ont fini de s'exécuter, 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.

Filtrage d'items dans les vues

Dans certaines vues, comme "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.



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 en cliquant 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 31

Bons de travail

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 qui sont 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

Quand un workflow de données émet un bon de travail pendant son exécution pour une tâche utilisateur définie dans le modèle de workflow, 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', et le bon de travail passe alors à l'état *démarré*.

Par défaut, pour chaque rôle défini comme participant de la tâche utilisateur, le workflow 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 passant 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, plaçant ainsi le bon de travail dans 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*.

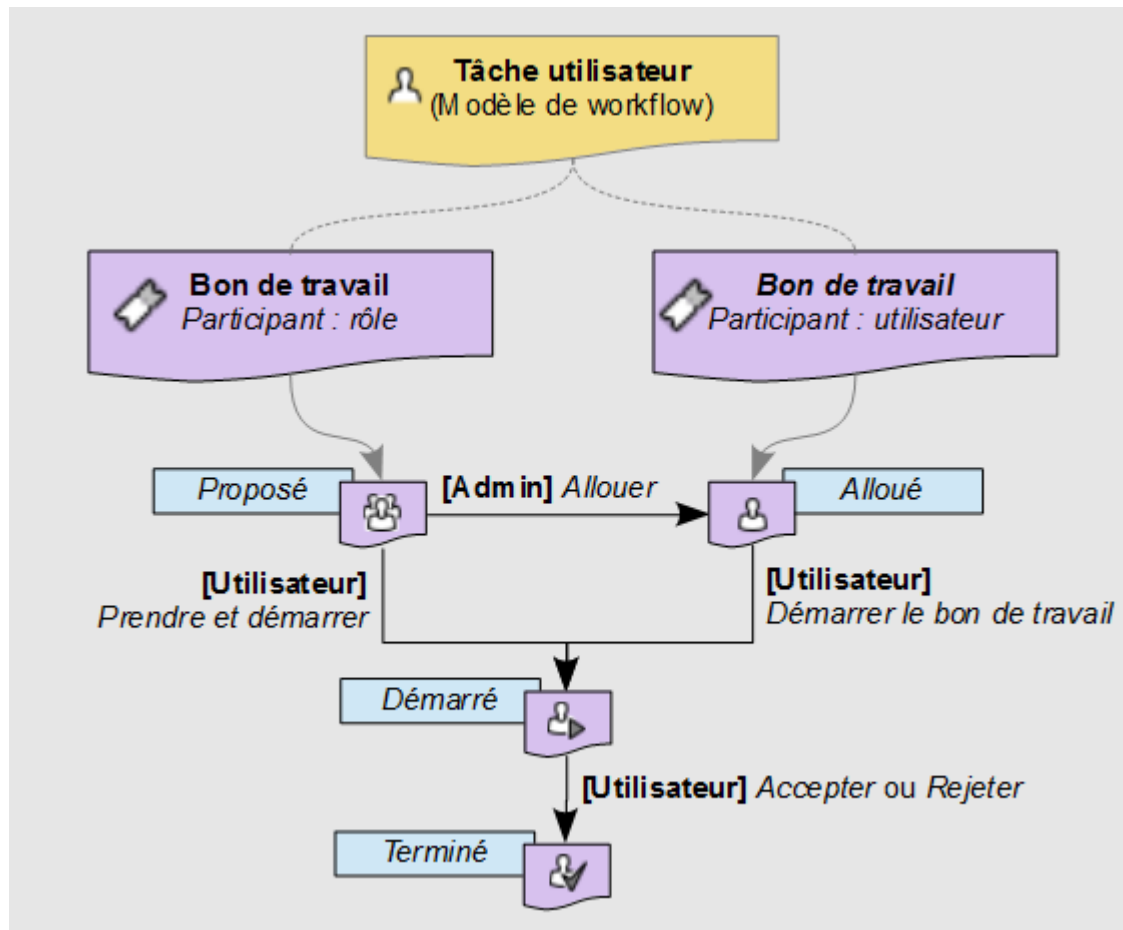
Note

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 finit 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



Travail sur un bon de travail en tant que participant

Tous les bons de travail qui vous sont destinés en tant qu'utilisateur (qui vous sont soit proposé, soit alloué), 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 autres participants du workflow de données, et les administrateurs et superviseurs du workflow. 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 continues à partir du même point.

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 \(glossaire\)](#)

Concepts apparentés : [Tâche utilisateur](#)

CHAPITRE 32

Lancement et monitoring de workflows de données

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 de données en utilisant la vue "Lanceurs de workflow" dans le panneau de navigation. Pour créer un nouveau workflow de données depuis une publication d'un modèle de workflow, cliquez 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.

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.

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

Gestion de l'allocation des bons de travail

Quand un modèle de workflow vous donne les permissions de gestion de l'allocation des bons de travail, vous pouvez intervenir pour allouer manuellement des bons de travail durant l'exécution des workflows de données associés au modèle de workflow. Dans ce cas, vous pouvez effectuer une ou plusieurs des actions ci-dessous sur les bons de travail.

Note

Ces actions ne sont pas disponibles pour les bons de travail qui ne définissent pas un rôle pour proposer le bon de travail (uniquement un utilisateur spécifique).

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> .
Réalouer	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](#)
- [Permission sur une publication de workflow](#)

Concepts apparentés : [Modèles de workflow](#)

CHAPITRE 33

Administration de workflows de données

Si vous disposez de permissions pour administrer des workflows de données, toutes les publications, workflows actifs, et bons de travail associés seront affichés dans les différentes vues de la section "Monitoring" dans le panneau de navigation. Dans ces vues de monitoring, vous pouvez accéder directement aux menus **Actions** sur les lignes des tables pour effectuer les 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.

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.

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.
- **Terminé** : Le jeton a atteint la fin du workflow de données.
- **Erreur** : Une erreur est survenue.

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, en suivant le processus décrit dans la section [Désactivation d'une publication de workflow](#).
2. Dépublier la publication de workflow en sélectionnant **Actions > Dépublier** de la ligne de la publication cible.

Note

Quand vous décidez de dépublier 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 tous bons de travail associés. Toute perte de données résultante d'une fin prématurée est alors définitive.

Actions sur workflows 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 liés, et le jeton est repositionné au début de l'étape courante.

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

Forcer un workflow de données actif à se finir

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 de données et tous les bons de travail associés.

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'.

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 : [Permission sur une publication de workflow](#)

Services de données

CHAPITRE 34

Introduction aux services de données

Les services de données offrent la possibilité d'accéder et d'interagir avec EBX5 pour :

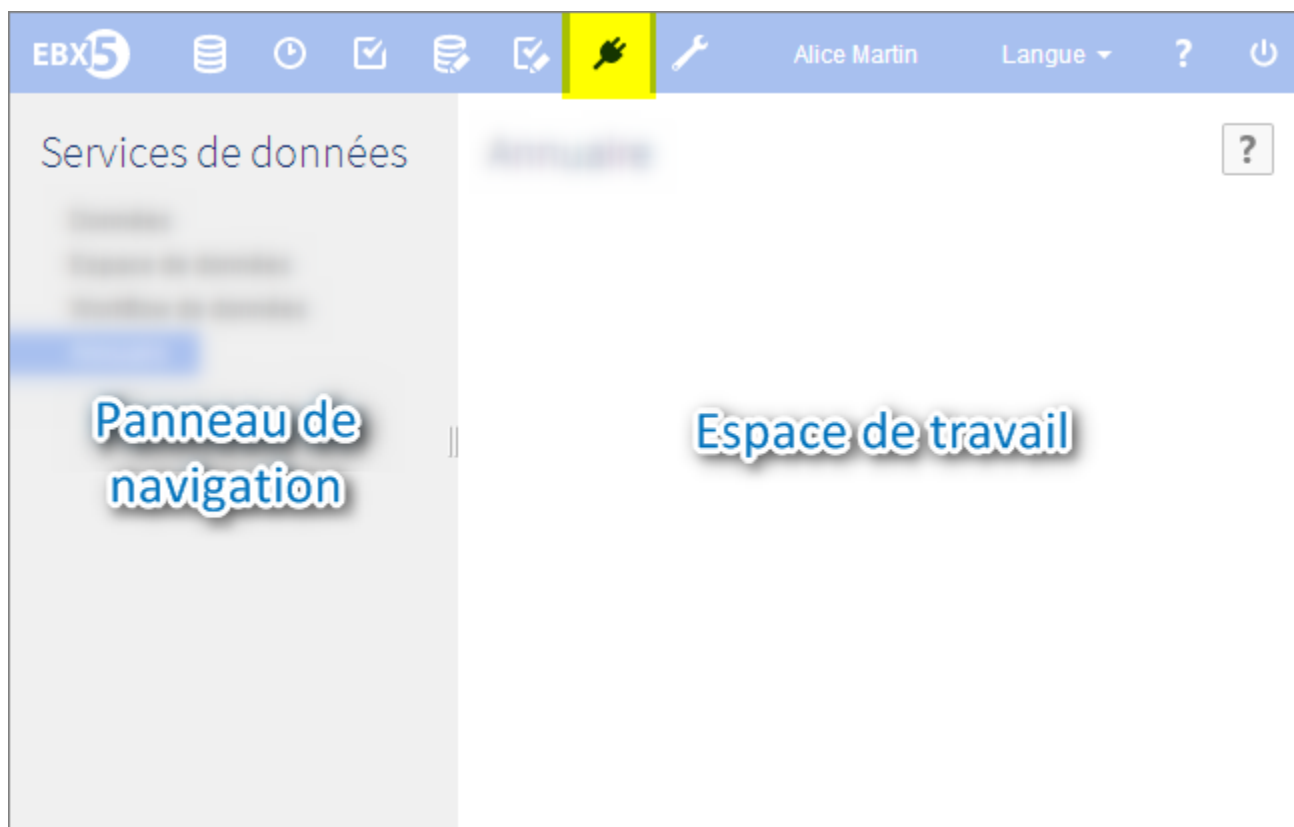
- accéder aux données stockées dans les tables (voir [données](#)),
- interagir avec les espaces de données (voir [espace de données](#)),
- contrôler les workflows (voir [workflow](#)),
- définir un lignage vers une logiciel tiers (voir [lignage](#)).

Un service de données est un service web standard, qui donne accès à toutes les fonctionnalités et données (créer un espace de données, mettre à jour les enregistrements d'un jeu de données, etc.).

Si vous vous demandez toujours à quoi ces termes techniques se réfèrent, n'hésitez pas à aller à la page [glossaire](#).

Vue d'ensemble des services de données

Modalités d'accès dans l'interface



Concepts et outils associés

Services de données standard	Services de données permettant d'accéder à un jeu de données, un espace de données ou un workflow en particulier
Lignage	Services de données prenant en considération les permissions utilisateur et évitant de possibles erreurs, en cas d'accès à des données non visibles de l'utilisateur actuellement identifié.

Pour en savoir plus sur la norme WSDL (Web Services Description Language), veuillez consulter le site du [World Wide Web Consortium \(W3C\)](http://www.w3.org/).

Voir aussi :

- [Modèles de données](#)
- [Espaces de données](#)
- [Jeux de données](#)

- [*Modèles de workflow*](#)
- [*Workflows de données*](#)

CHAPITRE 35

Actions principales

Générer un WSDL pour accéder aux données

La génération de WSDL pour l'accès de données est possible par sélection de *Données* dans la section *Services de données*.

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

1. Sélectionnez un espace de données
2. Entrez l'identifiant d'un jeu de données (ou nom unique) et cliquez sur suivant
3. Sélectionnez les opérations autorisées sur chaque table
4. Téléchargez le WSDL généré

Opérations disponibles sur la table d'un jeu de données

- *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* : vous donne un résumé des différences entre une table et son équivalent, s'il existe, dans une image ou un autre espace de données
- *Réaliser plusieurs opérations sur les tables d'un jeu de données.*

Générer un WSDL pour accéder à un espace de données

La génération de WSDL pour la manipulation d'un espace de données est accessible par sélection de *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 un WSDL* situé dans l'espace de travail.

Opérations disponibles

- *Créer un espace de données.*
- *Fermer un espace de données.*
- *Créer l'image d'un espace de données.*
- *Fermer l'image d'un espace de données.*
- *Fusionner un espace de données.*
- *Valider un espace de données ou une image d'espace de données.*
- *Valider un jeu de données.*

Générer un WSDL pour contrôler un workflow

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

Opérations disponibles

- *Démarrer un workflow.*
- *Finir un workflow.*

Générer un WSDL pour un lignage

La génération d'un WSDL pour un lignage est accessible par sélection de *Lignage* dans la section *Services de données*, sous réserves que des profils aient été autorisés par un profil administrateur dans la section *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](#).

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

1. Sélectionnez un rôle ou un utilisateur, dont les permissions seront appliquées. Veuillez noter qu'un rôle ou un utilisateur doit être autorisé à être utilisé pour le lignage par un administrateur.
2. Sélectionnez un espace de données.
3. Entrez l'identifiant du jeu de données (ou nom unique) et cliquez sur suivant.
4. Sélectionnez les tables et les opérations autorisées.
5. Téléchargez le WSDL généré.

Manuel de référence (en anglais)

Intégration

CHAPITRE 36

Using EBX5 as a web component

Overview

EBX5 can be used as a user interface web component, called through the HTTP protocol. An EBX5 web component can be integrated into any application that is accessible through a supported web browser. This method of access offers the major benefits of EBX5, such as user authentication, data validation, and automatic user interface generation, while additionally providing the ability to focus user navigation on specific elements of the repository.

Typical uses of EBX5 web components include integrating them into the intranet frameworks of organizations or into applications that manage the assignment of specific tasks to users.

Repository element and scope selection

When an EBX5 web component is called, the user must first be authenticated in the newly instantiated HTTP session. The web component then selects a repository element and displays it according to the *scope* layout parameter defined in the request.

The repository elements that can be selected are as follows:

- Data space or snapshot
- Data set
- Node
- Table or a published custom view
- Table record

The scope determines how much of the user interface is displayed to the user, thus defining where the user is able to navigate in the session. The default scope that the web component uses depends on the entity or service being selected or invoked by the request.

Request specifications

Base URL

In a default deployment, the base URL must be of the following form:

```
http://<host>[:<port>]/ebx/
```

User authentication and session information parameters

Parameter	Description	Required
login and password, or a <i>user directory-specific token</i>	Specifies user authentication properties. If neither a login and password pair nor a user directory-specific token is provided, user will be required to authenticate using through the repository login page.	No
trackingInfo	Specifies the tracking information of the new session. Tracking information is logged in history tables. Additionally, it can be used to programmatically restrict access permissions.	No
redirect	The URL to which the user will be redirected at the end of the component session, when they click on the button 'Close'. The close button is always displayed for record selections, but whether or not it is displayed for all other cases must be specified using the parameter <code>closeButton</code> .	No
locale	Specifies the locale to use. Value is either <code>en-US</code> or <code>fr-FR</code> .	No, default is the locale registered for the user.

Entity and service selection parameters

Parameter	Description	Required
branch	Selects the specified data space.	No
version	Selects the specified snapshot.	No
instance	Selects the specified data set. The value must be the reference of a data set that exists in the selected data space or snapshot.	Only if <code>xpath</code> or <code>viewPublication</code> is specified.
viewPublication	<p>Specifies the publication name of the tabular view to apply to the selected content.</p> <p>This publication name is the one declared during the publication of the view. It can be found in the 'Administration' area under Views configuration > User views.</p> <p>All settings of the view, that is, its filters, sort order, and displayed columns, are applied to the result. A data space and a data set must be selected in order for this view to be applied. The target table selection is not necessary, as it can be automatically determined based on the definition of the view. This parameter can be combined with the predicate specified in the <code>xpath</code> parameter as a logical 'AND' operation.</p>	No
xpath	<p>Specifies a node selection in the data set. Value must be a valid absolute path located in the selected data set. The notation must conform to a simplified XPath, with abbreviated syntax.</p> <p>If the parameter <code>viewPublication</code> is also specified, the table path is not necessary. The parameter can directly contain the predicate, surrounded by "[" and "]".</p> <p>For XPath syntax, see XPath supported syntax</p>	No
service	<p>Specifies the service to access.</p> <p>For more information on built-in UI services, see Built-in services.</p>	No

Example calls to an EBX5 web component

Minimal URI:

```
http://localhost:8080/ebx/
```

Logs in as the user 'admin' and selects the 'Reference' data space:

```
http://localhost:8080/ebx/?login=admin&password=admin&branch=Reference
```

Selects the 'Reference' data space and accesses the built-in validation service:

```
http://localhost:8080/ebx/?
login=admin&password=admin&branch=Reference&service=@validation
```

Selects the roles table in the default directory:

```
http://localhost:8080/ebx/?login=admin&password=admin&branch=ebx-
directory&instance=ebx-directory&xpath=/directory/roles
```

Selects the record 'admin' in the default directory:

```
http://localhost:8080/ebx/?login=admin&password=admin&branch=ebx-  
directory&instance=ebx-directory&xpath=/directory/user[./login="admin"]
```

Accesses the interface for creating a new user in the default directory:

```
http://localhost:8080/ebx/?login=admin&password=admin&branch=ebx-  
directory&instance=ebx-directory&xpath=/directory/user&service=@creation
```

Compares the record 'admin' in the default directory with the record 'jSmith':

Compares the record 'R1' in the data set 'instanceld' in the data space 'Reference' with the record 'R0':

```
http://localhost:8080/ebx/?login=admin&password=admin&branch=ebx-  
directory&instance=ebx-directory&xpath=/directory/user[./  
login="admin"]&service=@compare&compare.branch=ebx-directory&compare.instance=ebx-  
directory&compare.xpath=/directory/user[./login="jSmith"]
```

CHAPITRE 37

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

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

Access a data space

The data space selection service is automatically considered to be 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.
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.

Access data (default service)

The default service. By default, this service is automatically considered 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'.
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.
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.

Compare contents

The compare service is automatically considered 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.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.
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 conform to a simplified XPath, in its abbreviated syntax.

Create a new record

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.
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.

Duplicate a record

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.
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.

Export data from a table in CSV files

The exportToCSV service is considered 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.
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.

Export data from a table in XML files

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.
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.

Import data into a table from an CSV file

The importFromCSV service is considered 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.
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.

Import data into a table from an XML file

The importFromXML service is considered 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.
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.

Merge a data space

The merge service is considered 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.
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.

Validate a data space, a snapshot or a data set

The validation service is automatically considered 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.
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 38

Data services

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 or ending a data workflow

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.XXX.YYY.ZZZ.AAA.BBB(AAAA.java:44) at
        com.orchestranetworks.XXX.YYY.ZZZ.CCC.DDD(CCC.java:40) ... </description>
    </m:StandardException>
  </detail>
</soapenv:Fault>
```

Data services security

Authentication

Authentication is mandatory. By default, several methods of authentication exist. They are described below, in the look-up order attempted by the Web Service connector.

Note

When using JMS, only the standard HTTP 'Base64 Basic Authentication' is attempted.

- Standard HTTP 'Base64 Basic Authentication' encoded using HTTP-Header Authorization, as described in [RFC 2324](#).

If the user agent wishes to send the userid "Aladdin" and password "open sesame", it will use the following header field: Authorization: Basic QWxhZGRpbjpvYVUyIHNlc2FtZQ==

- 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>user</wsse:Username>
      <wsse:Password Type="wsse:PasswordText">password</wsse:Password>
    </wsse:UsernameToken>
  </wsse:Security>
</SOAP-ENV:Header>
```

Overriding the default security header

It is possible to override the 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:input>
  ...
</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>

```

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).

Data service operations generated from a data model

These operations are generated using a given 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">`.

Common data model operation 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.	One of either this parameter or the 'version' parameter must be defined
version	The identifier of the snapshot to which the data set belongs.	One of 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 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 (used by the insert and update operations)	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 .	Yes
viewPublication	This parameter can be combined with the predicate parameter as a logical AND operation. The behavior of this parameter is described in the section EBX5 as a web component . 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

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 .	
version	See the description under Common parameters .	
instance	See the description under Common parameters .	
predicate	See the description under Common parameters . This parameter can be combined with the viewPublication parameter as a logical AND operation.	
viewPublication	See the description under Common parameters .	
includesTechnicalData	The response will contain technical data if 'true'. See also the optimistic locking 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 be fetched, this value is provided by the previous response, as the element <code>lastRecordPredicate</code> . 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>
  <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](#) 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>
</m:delete_{TableName}>

```

with:

Element	Description	Required
branch	See the description under Common parameters .	
version	See the description under Common parameters .	
instance	See the description under Common parameters .	
predicate	See the description under Common parameters .	
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 section.	No

Delete response

```

<ns1:delete_{TableName}Response xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <status>00</status>
</ns1:delete_{TableName}Response>

```

with:

Element	Description
status	'00' indicates that the operation has been executed successfully.

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 .
version	See the description under Common parameters .
instance	See the description under Common parameters .
predicate	See the description under Common parameters .

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>true</updateOrInsert>
  <byDelta>true</byDelta>
  <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 .	
version	See the description under Common parameters .	
instance	See the description under Common parameters .	
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. The complete behavior is described in the sections Insert and update operations .	No
data	See the description under Common parameters .	

Update response

```

<ns1:update_{TableName}Response xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <status>00</status>
</ns1:update_{TableName}Response>

```

with:

Element	Description
status	'00' indicates that the operation has been executed successfully.

Insert operation

Insert request

```

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

```

```

    ...
    </TableName>
  </XX>
</data>
</m:insert_{TableName}>

```

with:

Element	Description	Required
branch	See the description under Common parameters .	
version	See the description under Common parameters .	
instance	See the description under Common parameters .	
data	See the description under Common parameters .	
byDelta	<p>If 'true' and an element does not currently exist in the incoming message, the target value is not changed.</p> <p>The complete behavior is described in the sections Insert and update operations.</p>	No

Insert response

```

<ns1:insert_{TableName}Response xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <status>00</status>
  <inserted>
    <predicate>./a='String'</predicate>
  </inserted>
</ns1:insert_{TableName}Response>

```

with:

Element	Description
status	'00' indicates that the operation has been executed successfully.
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 .	
version	See the description under Common parameters .	
instance	See the description under Common parameters .	
compareWithBranch	The identifier of the data space with which to compare. You must use only one parameter between this one and compareWithVersion .	No
compareWithVersion	The identifier of the snapshot with which to compare. You must use only one parameter between this one and compareWithBranch .	No
resolvedMode	Defines whether or not the difference is calculated in resolved mode. Default is 'true'.	No

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>
```

```

</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>

```

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 .	
version	See the description under Common parameters .	
instance	See the description under Common parameters .	
viewPublication	See the description under Common parameters .	

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. 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>
  <request id="id1">
    <m:{operation}_{TableName}>
      ...
    </m:{operation}_{TableName}>
  </request>
  <request id="id2">
```

```

<m:{operation}_{TableName}>
...
</m:{operation}_{TableName}>
</request>
</m:multi>

```

with:

Element	Description	Required
branch	See the description under Common parameters .	
version	See the description under Common parameters .	
instance	See the description under Common parameters .	
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 <code>count</code>, <code>select</code>, <code>getChanges</code>, <code>getCredentials</code>, <code>insert</code>, <code>delete</code> or <code>update</code>.</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:multiResponse xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <response id="id1">
    <ns1:{operation}_{TableName}Response>
      ...
    </ns1:{operation}_{TableName}Response>
  </response>
  <response id="id2">
    <ns1:{operation}_{TableName}Response>
      ...
    </ns1:{operation}_{TableName}Response>
  </response>
</ns1:multiResponse>

```

with:

Element	Description
response	<p>This element contains the response of one operation. It is be 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>true</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.

Data service 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 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	Documentation for the data space or the snapshot to create. Multiple documentation elements may be used.	No
locale (nested under the documentation element)	Locale of the data space or snapshot documentation.	Only required when the <code>documentation</code> element is used
label (nested under the documentation element)	Label of the data space or the snapshot to create.	No

<i>Element</i>	<i>Description</i>	<i>Required</i>
description (nested under the documentation element)	Description of the data space or the snapshot to create.	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>true</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>
      </reportItem>
    </details>
  </validationReport>
</ns1:validateInstance_Response>
```

```
</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>00</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>00</status>
  <versionName>String</versionName>
</ns1:createVersion_Response>
```

with:

Element	Description
status	'00' indicates that the operation has been executed successfully.

Merge a data space

Merge data space request

```
<m:mergeBranch xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <branch>String</branch>
  <deleteDataOnMerge>false</deleteDataOnMerge>
  <deleteHistoryOnMerge>false</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>00</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>false</deleteDataOnClose>
  <deleteHistoryOnClose>false</deleteHistoryOnClose>
</m:closeBranch>
```

Close snapshot request:

```
<m:closeVersion xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <version>String</version>
  <deleteDataOnClose>false</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>00</status>
</ns1:closeBranch_Response>
```

Close snapshot request:

```
<ns1:closeVersion_Response xmlns:ns1="urn:ebx-schemas:dataservices_1.0">
  <status>00</status>
</ns1:closeVersion_Response>
```

Data service operations on data workflows

Parameters for operations on data workflows are as follows:

Element	Description	Required
publishedProcessKey	Identifier of the workflow to start.	Yes
documentation	Documentation for the process instance to be created.	No
parameters	Input parameters for the process instance to be created.	No
processInstanceId	Identifier of the process instance, as returned by the <code>workflowProcessInstanceStart</code> operation.	Yes

Start a workflow

```
<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>
```

End a workflow

```
<m:workflowProcessInstanceEnd xmlns:m="urn:ebx-schemas:dataservices_1.0">
  <publishedProcessKey>String</publishedProcessKey>
  <processInstanceId>String</processInstanceId>
</m:workflowProcessInstanceEnd>
```

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 39

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.

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 <code>null</code>. • 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 <code>null</code> from an empty string, the target field value is set to <code>null</code>. If it is a string that does not distinguish between the two, an empty string. <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 <code>update</code> operation, the field value remains unchanged. • For the <code>insert</code> operation, the behavior is the same as when <code>byDelta</code> mode is disabled.
Element exists but is empty (for example, <code><fieldA/></code>)	<ul style="list-style-type: none"> • For nodes of type <code>xs:string</code> (or one of its sub-types), the target field's value is set to <code>null</code> if it distinguishes <code>null</code> from an empty string. Otherwise, the value is set to empty string. • For non-<code>xs:string</code> type nodes, an exception is thrown in conformance with XML Schema.
Element is present and <code>null</code> (for example, <code><fieldA xsi:nil="true"/></code>)	<p>The target field is always set to <code>null</code> except for lists, for which it is not supported.</p> <p>In order to use the <code>xsi:nil="true"</code> attribute, you must import the namespace <code>xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"</code>.</p>

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.

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.

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

CHAPITRE 40

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.

Imports

When importing a CSV 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.

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.
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.
Separator	Specifies the field separator to use in the exported file. The default is to use commas.

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 be 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

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 errors, however, no such values are 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](#).

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.

CHAPITRE 41

Supported XPath syntax

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.

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,'yy') and osd:is-not-null(./col3))
```

```
contains(col3 , 'xxx') and ( not(coll=100) and date-greater-than(col2,'2007-12-30') )
```

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	../.. /books

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>	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'
<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)]`.

Contextual values

For predicates that are relative to a selected node, the criterion value right-hand side) 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)]`.

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).

Divers

CHAPITRE 42

Permissions

Permissions specify and regulate the access of each user to data and the actions he can execute.

Main principles

Permissions are related to actions (action authorized or not) and to access rights (hidden, read, read-write). The main entities controlled by permissions are:

- Data space
- Data set
- Table
- Group
- Field

Users, roles and profiles

The definition and the resolution of permissions make extensive use of the *profile* notion, a generic term that references either a user or a role.

Each user can participate in several roles and a role can be shared by several users.

Particular definitions:

- A user is an *administrator* when he participates in the built-in role ADMINISTRATOR.
- A user is an *owner of a data set* when he participates in the *owner* attribute specified on the root data set ("Infos" tab). In this case, the built-in role OWNER is activated when permissions are resolved in the context of the instance.
- A user is an *owner of a data space* when he participates in the *owner* attribute specified for the data space. In this case, the built-in role OWNER is activated when permissions are resolved in the context of the data space.

Permission rules

A permission rule defines the authorization granted to a profile on an entity.

User-defined permission rules are created through the use of EBX5 (see section [Definition of permissions with EBX5](#)).

Resolution of permissions

Permissions are always resolved in the context of an authenticated user session. Hence, the defined permission rules will be able to take into account the information associated with the current user, more particularly the roles in which he participates.

The resolution process is further detailed in section [Resolution of permissions](#) .

Particular privileges on data set

For a specific data set, the following tasks are accessible only if the user is an administrator or an owner of this data set:

- Manage permissions of data set.
- Change the owner of the root data set.
- Change the data set documentation (label and description).

Notes.

- An administrator or an owner of a data set can have a restricted access to a data set through permissions definitions, but in any case, he will keep the possibility to perform the tasks defined above.

Particular privileges on data space

A user is a "*super owner*" of a data space when:

- He is the owner of the data space and he is allowed to manage its permissions;
- or he is the owner of an ancestor data space and he is allowed to manage the permissions of this ancestor.

For a specific data space, the general rule is that only an administrator or a "super owner" may perform the following administration tasks:

- Manage permissions of data space.
- Change owner of data space.
- Lock and unlock data space.
- Change the data space documentation (label and description).

Notes. An administrator or an owner of a data space can have a restricted access to it through permissions definitions, but in any case, he will keep the possibility to perform administration tasks defined above. This implies that an administrator always sees all open data spaces and that any user sees all the data spaces that he owns.

Merge and permissions

When a data space is merged, the permissions of its parent data space are not impacted but the permissions of the data set of the parent data space are merged if and only if the user specifies it during the merge process. However when some elements are hidden for a profile, this profile will not be able to merge since he would not be aware of the actual impacts on data.

Definition of permissions with EBX5

As described below, each level has a similar schema that allows the definition of several permission rules.

Permissions on data space

For a given data space, several permission rules can be specified. For each defined profile, the allowed permissions are the following:

1. Data space access: defines access rights (read-write, read-only or hidden, see below for definition).
2. Restriction: indicates if "profile (role or user) - permission (right or action)" associations, for the current data space, should have priority.
3. Create a child data space: indicates if it is possible to create a child data space from the current data space.
4. Create a child snapshot: indicates if it is possible to create a snapshot of the current data space.
5. Initiate merge: indicates if a profile has the right to merge a data space with its parent data space.
6. Export archive: indicates if a profile has the right to export the current data space as an archive.
7. Import archive: indicates if a profile has the right to import an archive into the current data space.
8. Close a data space: indicates if a profile has the right to close the current data space.
9. Close a snapshot: indicates if a profile has the right to close a snapshot of the current data space.
10. Permissions of child data space when created: defined the same permissions as above but on current data space children.

Mode	Authorization
Write	<ul style="list-style-type: none"> The user can see the data space. The user has the right to access data set according to his rights on these.
Read-only	<ul style="list-style-type: none"> The user can visualize the data space and its snapshot. He can see the child data space if he has the right to do so. The user can at most visualize the contents of the data space. He cannot modify the data space contents.
Hiddens	<ul style="list-style-type: none"> The user can neither see the data space nor its snapshots. If the user has access to a child data space, then he can see the current data space but he cannot select it. The user cannot access the data space contents (data set contents). The user cannot perform any action on the data space.

Permissions on a new data space

When a user creates a child data space (if he is allowed to do so), the permissions of this new data space are automatically assigned to the profile "owner" and are determined from the *Permissions of child data space when created* section defined on the parent data space. If several permissions are defined through different roles, the [resolved permissions](#) are applied.

Note

Only the administrator and the owner of a data space can define a new owner for this data space or modify the associated permissions. They can also modify general information on the data space ("owner role" for data set) and also permissions of the different users.

Furthermore, if using the "Create a data space" or "Create a snapshot" built-in script tasks and since the current session is related to a system user, the resolved permissions is computed using the owner defined in the script task's configuration and is not based on the user associated with the session.

Permissions on data set

For a given data set, several permission rules can be specified. For each defined profile, permissions are the following:

Actions on data set

- Restricted mode: indicates if "profile (role or user) - permission (right or action)" associations, for the current data set, should have priority.
- Create a child data set: indicates if a given profile has the right to create a child data set.
- Duplicate data set: indicates if a given profile has the right to duplicate a data set.
- Change the data set parent: indicates if a given profile has the right to change the parent data set of a given child data set.

Default actions on tables

For a given table, several "profile-permissions" associations can be specified. For each defined profile, possible actions are the following:

- Create a record: indicates if a profile has the right to create records in a table.
- Modify a record: indicates if a profile has the right to modify a record (in case of inheritance of data in a data set tree).
- Hide a record: indicates if a profile has the right to hide a record in a table.
- Delete a record: indicates if a profile has the right to delete a record in a table.

Permissions on tables

The actions specified on tables modify the default actions (*cf.* above) on these tables in the data set.

Default access right on nodes values

- Read-write: nodes can be consulted and modified (modification of values).
- Read: nodes can only be consulted, not modified.
- Hidden: nodes are hidden.

Permissions on terminal nodes

Rights specified on terminal nodes modify the default rights (*cf.* above) on these terminal nodes.

Permissions on services

Permissions on data set services can also be defined. By default, all data set services are enabled. The administrator or the owner of the data set can modify these services permissions for a given profile.

Resolution of permissions

The resolution of permissions is always done in the context of a user's session and his associated roles. In general, a restrictive resolution is performed between a given level and its parent level. Hence at a given level, a user cannot have a permission higher than the one resolved at a parent level.

Restriction policy notion

Permissions defined with EBX5 can be restricted. Given a set of profiles to which a user belongs to, restricting some of them means to consider their permissions more important than those defined for non restricted profiles. Hence, generally:

- if restrictions are defined, the minimum permission over the restricted profiles is considered
- if no restriction is defined, the maximum permission over profiles is then taken

Example

Given two profiles P1 and P2, the following table lists the possibilities when resolving the access to a service.

P1 authorization	P2 authorization	Permission resolution
Allowed.	Allowed.	Allowed. Restrictions don't interfere.
Forbidden.	Forbidden.	Forbidden. Restrictions don't interfere.
Allowed.	Forbidden.	Allowed, unless P2 has a restriction.
Forbidden.	Allowed.	Allowed, unless P1 has a restriction.

Another example would be to hide a data space from all users, the administrator or the owner of this data space would define a restriction on the association between the built-in profile "Profile.EVERYONE" and the access right "hidden".

Solving access rights

Solving access rights defined with EBX5

For access rights resolution defined with EBX5, there are three levels of resolution: data space, data set and node.

If a user is associated with different access rights at a given level and if restrictions have been defined for these "user-rights" associations, then the minimum of the restricted rights is applied. If there is no restriction, then the maximum access right is applied for the given user. The following tables illustrate the resolution mechanism.

Three users exist and each one of them participates in different roles or profiles:

- User 1: user1 - role A - role B
- User 2: role A - role B - role C
- User 3: user3 - role A - role C

This table indicates the rights associated with those different profiles on a specific object:

Role/Profile	Rights	Restriction policy
user1	Hidden	Yes
user3	Read	No
Role A	Read/Write	No
Role B	Read	Yes
Role C	Hidden	No

This table lists the right which will be applied for each user after rights resolution:

User	Right applied
User 1	<i>hidden</i>
User 2	<i>read</i>
User 3	<i>read-write</i>

At a given level, the most restrictive right of this level and of the higher levels is applied. For instance, if a data set is in read-write access and the container data space allows only reading, then the user will have a read-only access right on this data set.

Note. Rights defined on a data set will be applied on its child data set. It is possible to modify these rights on the child data set. *The inheritance mechanism is therefore applied for either values or access rights.*

Data space/snapshot access right resolution

At data space level, the access right will be the following: if a user has several rights defined (one for each of his roles) and if restrictions have been defined, then the minimum of the restricted "user-rights" associations is applied. Otherwise, the maximum of the "user-rights" associations of the given user is applied.

On the other hand, if the user has no right defined, then if he is administrator or owner of the data space, he will have a read-write access to this data space, otherwise the data space will be hidden from him.

Data set access right resolution

At data set level, the same principle as the one applied to data space is used. Moreover, access right on data set is defined by the minimum between the right on the data space and the right on the data set (identified by using the "solving rights" principle similar to the one applied at the data space level). For instance, if a data space is in read-only access for the user U and a data set of the data space is in read-write access for the same user, then U will have a read-only access on the data set.

Node access right resolution

At node level, the same principle as the one applied to data space and to data set is used. Moreover, the right on the node is defined by the minimum between the right on the data set and the right on the node (identified by using the "solving rights" principle similar to the one applied at data space level and at data set level).

Note. At node level, one can define a specific access right for the target node. If no specific access right is defined, the default access right is then considered for the resolution process. However, the procedure is slightly different for table and table child node (see next section).

Specific case of table and table child node

This section describes the resolution process applied for a given table node or table record *N*.

For each user-defined permission rules that matches one of the user's profiles, the access right for *N* is either:

1. The locally defined access right for *N*;
2. Inherited from the access right defined on the table node;
3. Inherited from the default access right for instance's values.

Then, all matching user-defined permission rules will provide the resolved access right for *N*. Resolution is done according to [restriction policy](#).

The final resolved access right will be the minimum between data space, data set and the resolved access right for *N*.

Solving actions and services

Solving which actions and which services are accessible to a given user is done using the same process.

When several actions lists are defined for a given profile on a data set (respectively table), the actions list to consider is dynamically generated after an evaluation of each action type among the different lists (of actions) associated with this user. If some "user-(list of) actions" associations are restricted, then for each action type we verify (among the restricted associations) whether it is forbidden at least once to forbid it at all. If there is no restriction, then if at least one action type is authorized then the action is finally allowed (*cf.* tables below for actions on tables).

Two users exist and each one of them participates in different roles or profiles:

- User 1: user1 - role A - role B
- User 2: role C - role D

This table indicates the rights associated with those different profiles on a table:

Role/Profile	Create a record	Modify a record	Hide a record	Duplicate a record	Delete a record	Restriction policy
user1	Allowed	Forbidden	Allowed	Forbidden	Allowed	No
Role A	Allowed	Allowed	Forbidden	Allowed	Forbidden	Yes
Role B	Allowed	Forbidden	Allowed	Allowed	Forbidden	Yes
Role C	Allowed	Allowed	Forbidden	Forbidden	Forbidden	No
Role D	Allowed	Forbidden	Forbidden	Allowed	Forbidden	No

This table lists the actions that will be allowed for each user after rights resolution:

Users	Actions list applied
User 1	Create a record
	Duplicate a record
User 2	Create a record
	Modify a record
	Duplicate a record

CHAPITRE 43

Inheritance and value resolution

Overview

The main idea behind inheritance is to mutualize resources that are shared by multiple contexts or entities. EBX5 offers several mechanisms for defining, factorizing and resolving data values: *data set inheritance* and *inherited fields*

Data set inheritance

Data set inheritance is particularly useful when data has to be specialized to various global enterprise contexts, like subsidiaries or business partners.

Based on a hierarchy of data sets, it allows you to factorize common data on the root data set (or on intermediate ones) and to specialize specific data to specific contexts.

Data set inheritance mechanisms are detailed in the [section below](#) .

Inherited fields

As opposed to data set inheritance (which exploits a global built-in relationship between data sets), inherited fields are able to exploit finer-grained dependencies that are specific to the data structure. It allows you to factorize and specialize data at the business entities-level.

For example, if the model specifies that a Product is associated with a FamilyOfProducts, it is possible that some attributes of Product inherit their value from attributes defined in its associated family.

Note: It is not possible to use both attribute inheritance and data set inheritance for a same data set.

Data set inheritance

As described above, data set inheritance is based on a hierarchy of data set, the *data set tree*. This section's purpose is to precisely define data set inheritance mechanisms.

Lookup mechanism for values

Formally speaking, the data set inheritance lookup mechanism for values is as follows:

1. if the value is locally defined, returns it
2. otherwise, looks up the first locally defined value according to the built-in child-to-parent relationship of data set in the hierarchy tree
3. if no locally defined value is found, the default value is returned

`null` is returned if no default value is defined

Note: default values cannot be defined on:

- a single primary key node
- auto-incremented nodes
- nodes defining a computed value.

Lookup mechanism for records

Like values, table records can also be inherited as a whole by multiple contexts, but they can also be partially redefined (*overwritten*), be specific to a context (*root mode*) or even be occulted. More formally speaking, a table record has one of four distinct definition modes:

- A *root record* is locally defined in the table and has no parent. This means that no record with the same primary key exists in the parent table, or that this parent is an occulting record.
- An *overwriting record* is locally defined in the table and has a parent record. This means that a record with the same primary key exists in the parent table, and that this parent is not an occulting record. The overwriting record inherits its values from its parent, except for the values that it explicitly redefines.
- An *inherited record* has a parent record but is not locally defined in the current table. All values are inherited.
- An *occulting record* specifies that if a parent with same primary key is defined, this parent will not be seen in table descendants.

Inherited fields

The specific inheritance mechanism allows the fetching of the value of a node according to its relationship to other tables.

Lookup mechanism for value

The lookup mechanism for inherited fields values is the following:

1. If the value is locally defined, returns it.
2. Otherwise, looks up the source record and value to inherit from, according to the properties that are defined in the data model.
3. The process is recursive, if the source node does not locally define a value, it is further looked up, according to the inheritance behavior of the source node.

Optimize & Refactor

EBX5 provides a built-in UI service for optimizing data set inheritance in your data set tree. This service performs the following functions:

- *Handle duplicated values*: this procedure detects and removes all parameter values that are duplicates of the inherited value.
- *Mutualize common values*: this procedure detects and mutualizes the common values among the descendants of a common ancestor.

Procedure details

Data sets are processed from the bottom up, which means that if the service is run on the data set at level N , with $N+1$ being the level of its children and $N+2$ being the level of its children's children, the service will first process the data sets at level $N+2$ to determine if they can be optimized with respect to the data sets at level $N+1$. Next, it would proceed with an optimization of level $N+1$ against level N .

Note

- These optimization and refactoring functions do not handle default values that are declared in the data model.
- The highest level considered during the optimization procedure is always the data set on which the service is run. This means that optimization and refactoring is not performed between the target data set and its own parent.
- Table optimization is done for records with the same primary key.
- Inherited fields are not optimized.
- *Optimization and refactoring functions do not modify the resolved view of a data set if it is activated.*

Service availability

The 'Optimize & Refactor' service is available on data sets that have child data sets and also have the property 'Activated' set to 'No' in its data set information.

The service is available to any profile with write access on current data set values. It can be disabled by setting a restrictive access right on a profile.

Note

For performance reasons, access rights are not verified on every node and table record.

CHAPITRE 44

Criteria Editor

EBX5 integrates a specific criteria editor. It is used in order to define filters on table, validation and computation rules on data. This editor is based on XPath 1.0 Recommendation.

Two kinds of criteria exist: atomic criteria and conditional blocks.

Voir aussi : [Supported XPath syntax](#)

Conditional Blocks

Conditional blocks, or conditional criteria, are made up of atomic criteria and conditional blocs. They express a condition over the criteria. Four kinds of block are defined:

- *Don't match any criteria*: no criteria must be matched.
- *Don't match one of criteria*: one of the criteria must not be matched.
- *Match all criteria* all criteria must be matched.
- *Match one of criteria* one of the criteria must be matched.

Atomic Criteria

An atomic predicate. It is defined by a field, an operator and an expression (a value or a XPath formula).

Field	Specify the field of the table on which the criterion applies.
Operator	Specify the operator used. Available operators depend on the field type.
Expression	Specify the value or expression (see section below).

Expression

The expression can either be a fixed value or a formula. While creating a filter, only fixed value is authorized. A formula can be created using the assistant interface while creating validation or computation rules.

Known limitation: field formula does not check input values, only syntax and path are checked.