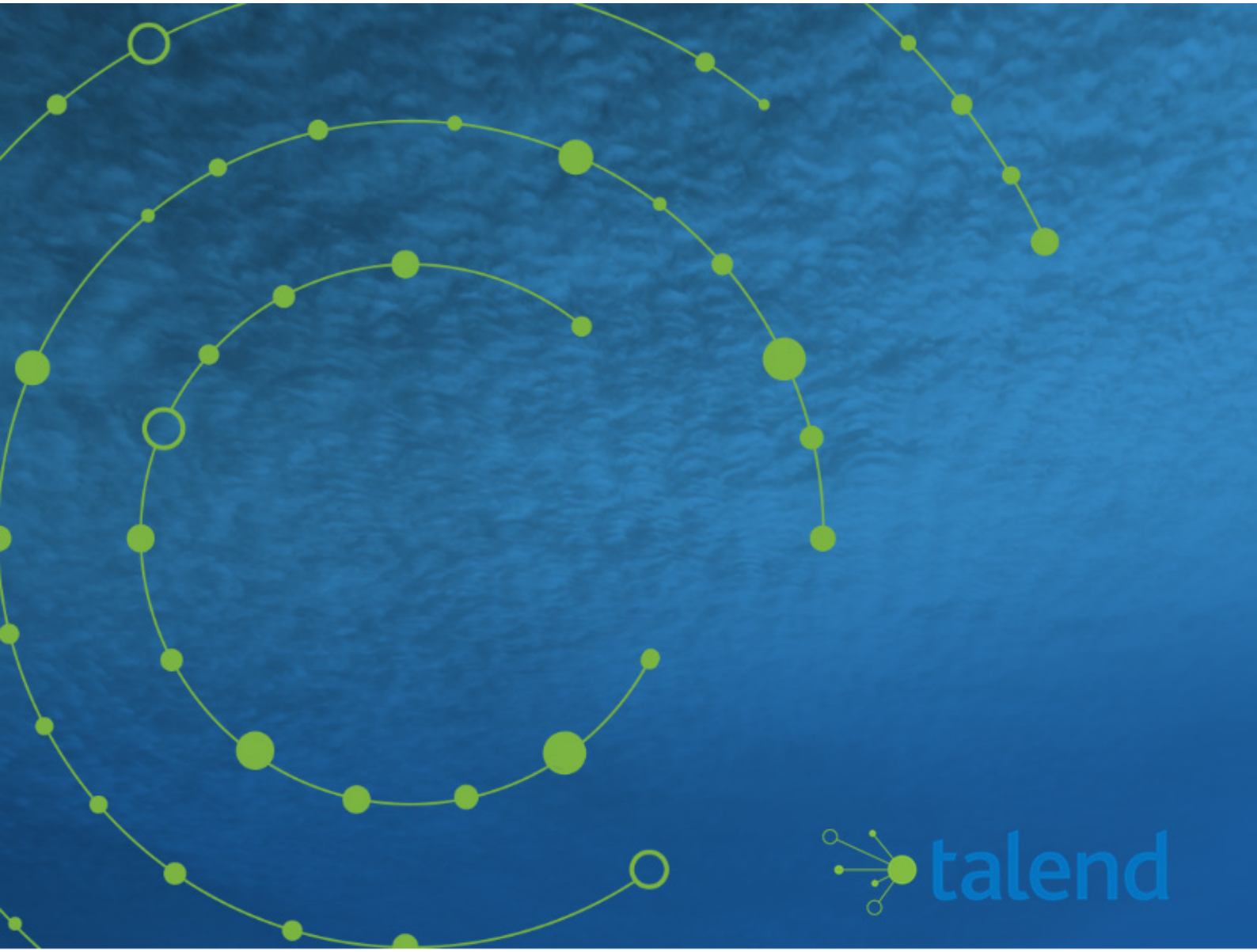


Exemples de Jobs Big Data

7.0.1

Table des matières

Copyleft

Convient à la version 7.0.1. Annule et remplace toute version antérieure de ce guide.

Date de publication : 13 avril 2018

Ce produit comprend les logiciels développés par AOP Alliance (standards Java/J2EE AOP), ASM, Amazon, AntlR, Apache ActiveMQ, Apache Ant, Apache Avro, Apache Axiom, Apache Axis, Apache Axis 2, ApacheBatik, Apache CXF, Apache Cassandra, Apache Chemistry, Apache Common Http Client, Apache Common HttpCore, Apache Commons, Apache Commons Bcel, Apache Commons JxPath, Apache Commons Lang, Apache Datafu, Apache Derby Database Engine and Embedded JDBC Driver, Apache Geronimo, Apache HCatalog, Apache Hadoop, Apache Hbase, Apache Hive, Apache HttpClient, Apache HttpComponents Client, ApacheJAMES, Apache Log4j, Apache Lucene Core, Apache Neethi, Apache Oozie, Apache POI, Apache Parquet, Apache Pig, Apache PiggyBank, Apache ServiceMix, Apache Sqoop, Apache Thrift, Apache Tomcat, ApacheVelocity, Apache WSS4J, Apache WebServices Common Utilities, Apache Xml-RPC, Apache Zookeeper, Box Java SDK (V2), CSV Tools, Cloudera HTrace, ConcurrentLinkedHashMap for Java, Couchbase Client, DataNucleus, DataStax Java Driver for Apache Cassandra, Ehcache, Ezmorph, Ganymed SSH-2 for Java, GoogleAPIs Client Library for Java, Google Gson, Groovy, Guava : Google Core Libraries for Java, H2 EmbeddedDatabase and JDBC Driver, Hector : client Java haut niveau pour Apache Cassandra, Hibernate BeanValidationAPI, Hibernate Validator, HighScale Lib, HsqlDB, Ini4j, JClouds, JDO-API, JLine, JSON, JSR 305: Annotations for Software Defect Detection in Java, JUnit, Jackson Java JSON-processor, Java API for RESTful Services, Java Agent for Memory Measurements, Jaxb, Jaxen, JetS3T, Jettison, Jetty, Joda-Time, Json Simple, LZ4 :Extremely Fast Compression algorithm, LightCouch, MetaStuff, Metrics API, Metrics Reporter Config, MicrosoftAzure SDK pour Java, Mondrian, MongoDB Java Driver, Netty, Ning Compression codec for LZF encoding, OpenSAML, Paraccel JDBC Driver, Parboiled, PostgreSQL JDBC Driver, Protocol Buffers - Google's datainterchange format, Resty : client simple HTTP REST pour Java, Riak Client, Rocoto, SDSU Java Library, SL4J :Simple Logging Facade for Java, SQLite JDBC Driver, Scala Lang, Simple API for CSS, Snappy for Java a fastcompressor/decompresser, SpyMemCached, SshJ, StAX API, StAXON - JSON via StAX, Super SCV, The CastorProject, The Legion of the Bouncy Castle, Twitter4J, Uuid, W3C, bibliothèques Windows Azure Storage pourJava, Woden, Woodstox : High-performance

XML processor, Xalan-J, Xerces2, XmlBeans, XmlSchema Core, Xmlsec - Apache Santuario, YAML parser et emitter pour Java, Zip4J, atinject, dropbox-sdk-java : bibliothèque Java pour l'API Dropbox Core, google-guice. Fournis sous leur licence respective.

Rassembler des informations concernant le trafic Web à l'aide d'Hadoop

Pour conduire une campagne marketing concernant les habitudes et les profils de vos clients ou utilisateurs, vous devez pouvoir récupérer des données selon leurs habitudes ou leur comportement sur votre site Web afin de créer des profils utilisateur et de leur envoyer les publicités adéquates, par exemple.

Le dossier `ApacheWebLog` du projet démo Big Data inclus dans le Studio Talend fournit un exemple permettant de retrouver les utilisateurs ayant le plus souvent visité un site Web, en triant les adresses IP à partir d'un grand nombre d'enregistrements dans le fichier de registre d'accès pour un serveur Apache HTTP, afin de faire d'autres analyses sur le comportement des utilisateurs sur le site Web. Cette section décrit les procédures pour créer et configurer des Jobs implémentant cet exemple. Pour plus d'informations concernant le projet démo Big Data, consultez le Guide de prise en main de votre Studio.

Avant de découvrir cet exemple et de créer les Jobs, vous devez :

- Avoir importé le projet démo et obtenu le fichier de registre d'accès utilisé dans cet exemple en exécutant le Job `GenerateWebLogFile` inclus dans le projet démo.
- Avoir installé et démarré l'appliance virtuelle Hortonworks Sandbox pour laquelle le projet démo est fait pour fonctionner tel que décrit dans le Guide de prise en main de votre Studio.
- Avoir ajouté une IP vers l'entrée de mapping du nom d'hôte dans le fichier `hosts` afin de résoudre le nom d'hôte `sandbox`.

Dans cet exemple, certains composants Big Data Talend sont utilisés pour tirer parti de la plateforme Open source Hadoop, dans le domaine de la gestion des Big Data. Dans ce scénario, vous utilisez six Jobs :

- le premier Job configure une base de données et une table HCatalog comprenant une partition, dans HDFS
- le deuxième Job charge le registre d'accès à analyser dans le système de fichiers HDFS.
- le troisième Job se connecte à la base de données HCatalog et affiche le contenu du fichier chargé dans la console.
- le quatrième Job analyse le fichier chargé. Il supprime notamment tout enregistrement contenant une erreur "404", compte les occurrences de code dans les appels de services vers le site Web exécutés avec succès, trie les données de résultats et les sauvegarde dans le système de fichiers HDFS.
- le cinquième Job analyse le fichier chargé. Il supprime notamment tout enregistrement contenant une erreur "404", compte les occurrences d'adresses IP dans les appels de services vers le site Web exécutés avec succès, trie les données de résultats et les sauvegarde dans le système de fichiers HDFS.
- le dernier Job lit les résultats depuis HDFS et affiche les adresses IP ainsi que les appels de services réussis et le nombre de visites du site Web dans la console standard du système.

Transformer le scénario en Jobs

Cette section décrit comment configurer les métadonnées de connexion utilisées dans les Jobs d'exemple et comment créer, configurer et exécuter les Jobs afin d'obtenir le résultat attendu pour ce scénario.

Configurer les métadonnées de connexion utilisées dans les Jobs

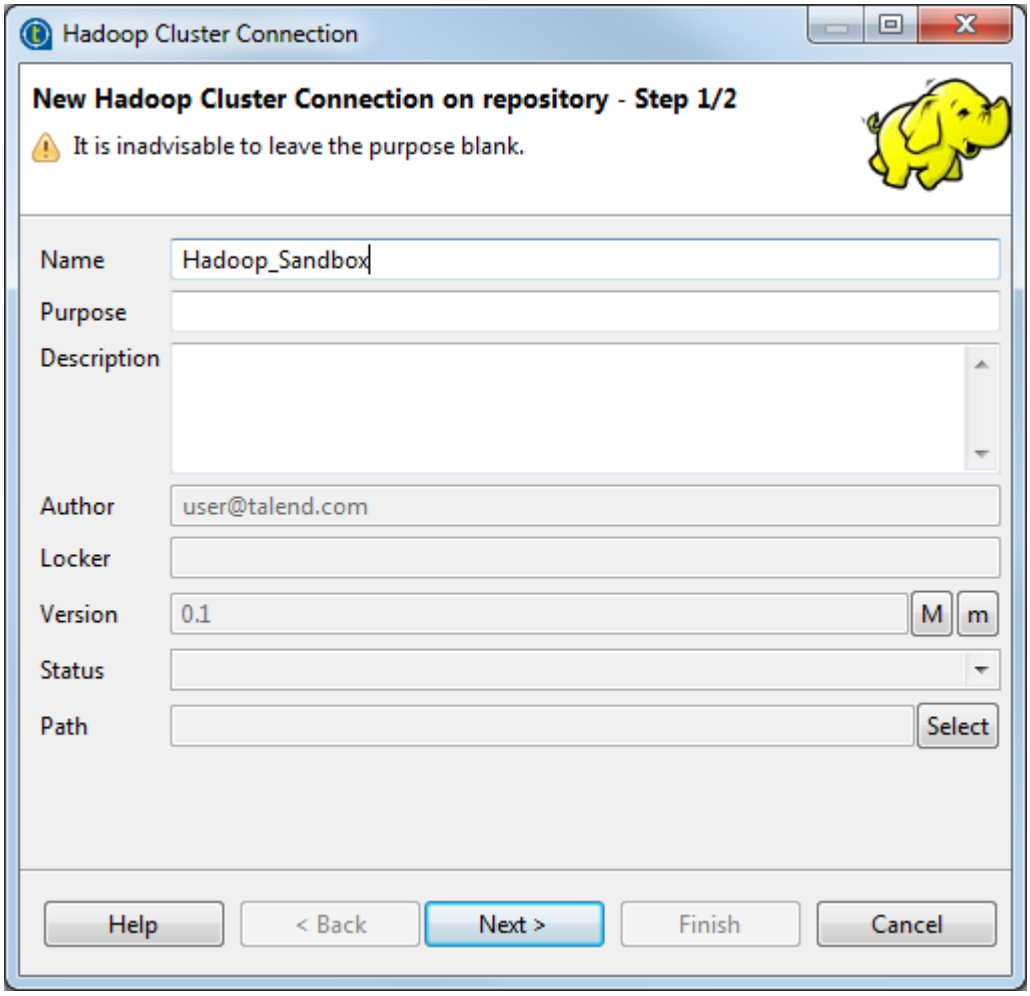
Dans ce scénario, une connexion HDFS et une connexion HCatalog sont utilisées plusieurs fois dans différents Jobs. Afin de simplifier la configuration des composants, il est possible de centraliser ces connexions sous le nœud **Hadoop Cluster** du **Repository**. Cela permet de réutiliser facilement ces connexions.

Ces éléments de métadonnées centralisés peuvent désormais être utilisés afin de configurer les détails de connexion de différents composants et dans différents Jobs. Notez que ces connexions n'ont pas de schémas de table définis. Ces schémas sont créés séparément lors de la configuration des Jobs d'exemple.

Configurer une connexion au cluster Hadoop

Procédure

1. Cliquez-droit sur **Hadoop cluster** sous le nœud **Metadata** du **Repository** puis sélectionnez **Create Hadoop** depuis le menu contextuel afin d'ouvrir l'assistant de configuration de la connexion. Donnez un nom à la connexion, `Hadoop_Sandbox` dans cet exemple. Enfin, cliquez sur **Next**.

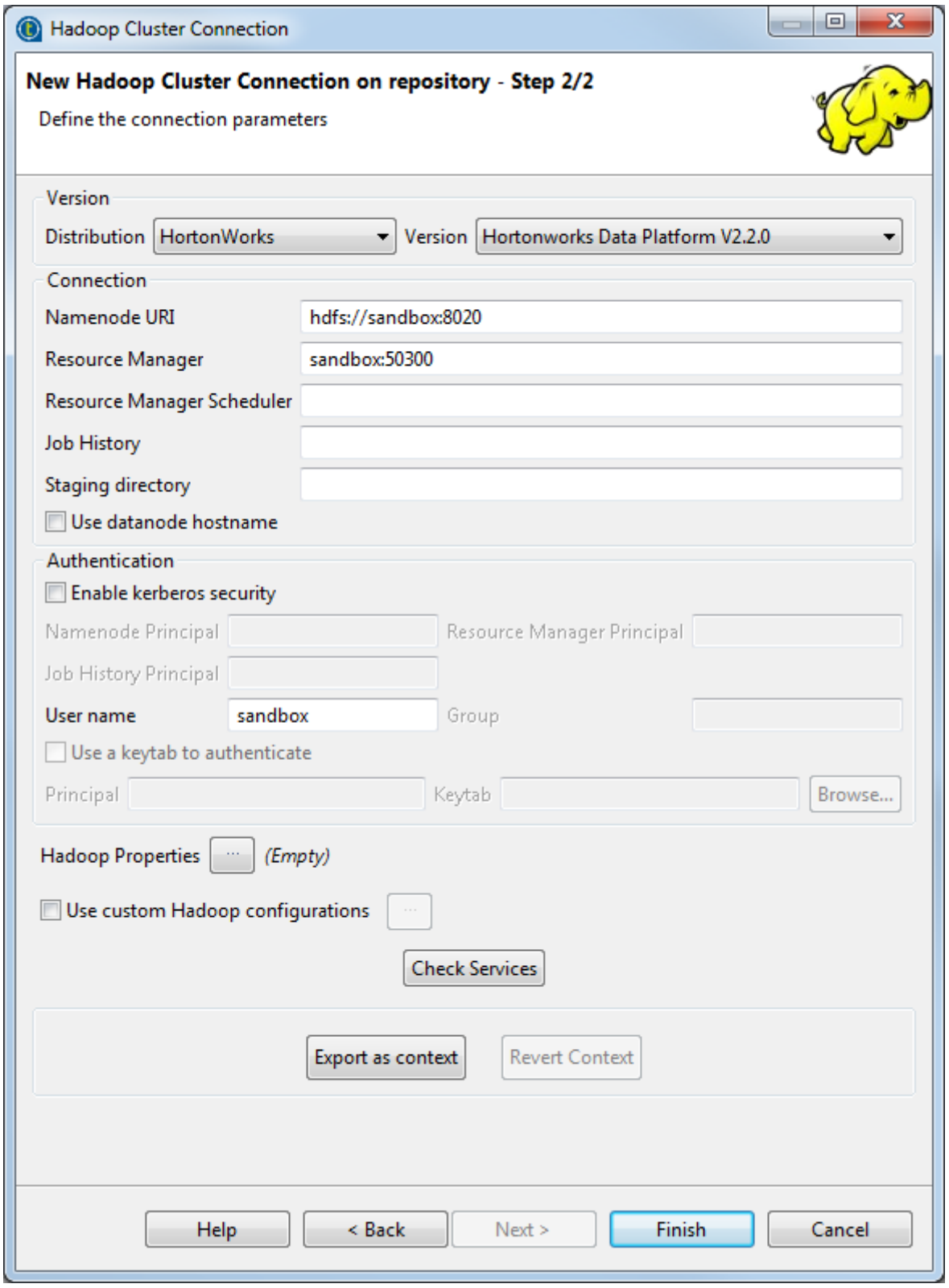


2. Configurez la connexion au cluster Hadoop :

a) Sélectionnez la distribution Hadoop et sa version.

b) Spécifiez l'URI du NameNode et le Resource Manager. Dans cet exemple, le nom d'hôte `sandbox` est utilisé pour le NameNode et le Resource Manager. Le nom d'hôte doit avoir été mappé vers l'adresse IP assignée à la machine virtuelle Sandbox. Les ports par défaut sont utilisés, `8020` pour le NameNode et `50300` pour le Resource Manager.

c) Spécifiez le nom d'utilisateur pour l'authentification Hadoop, `sandbox` dans cet exemple.



3. Cliquez sur **Finish**.

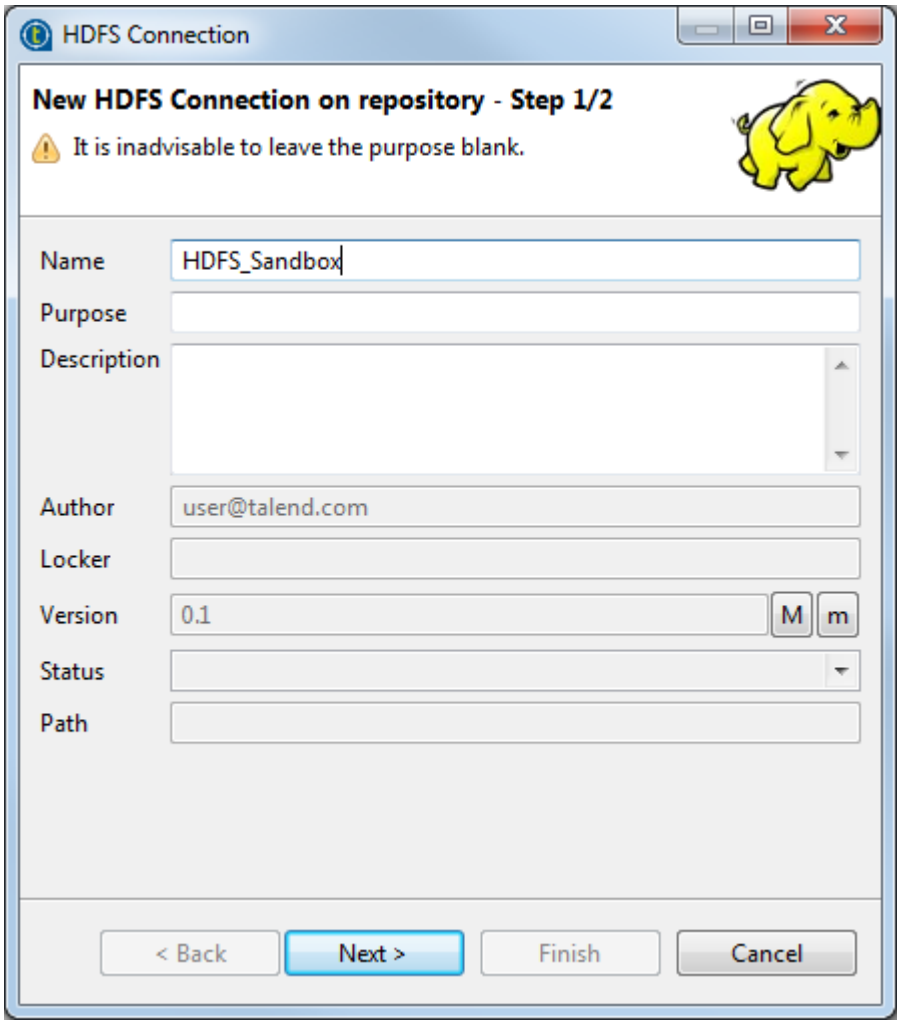
Résultats

La connexion au cluster Hadoop apparaît sous le nœud **Hadoop Cluster** du **Repository**.

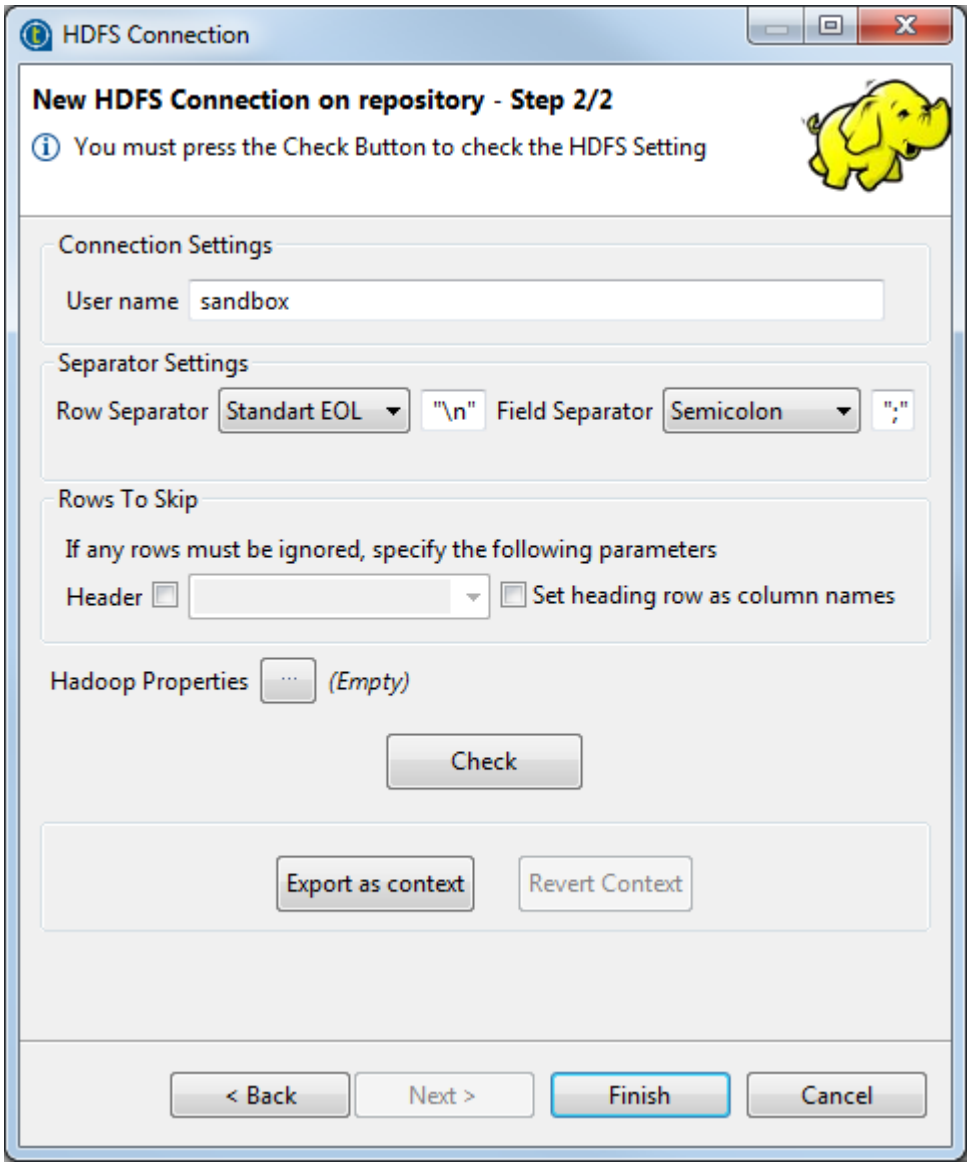
Configurer une connexion HDFS

Procédure

1. Cliquez-droit sur la connexion au cluster Hadoop que vous venez de créer et, dans le menu contextuel, cliquez sur **Create HDFS** afin d'ouvrir l'assistant de configuration de connexion. Donnez un nom à la connexion HDFS, `HDFS_Sandbox` dans cet exemple, puis cliquez sur **Next**.



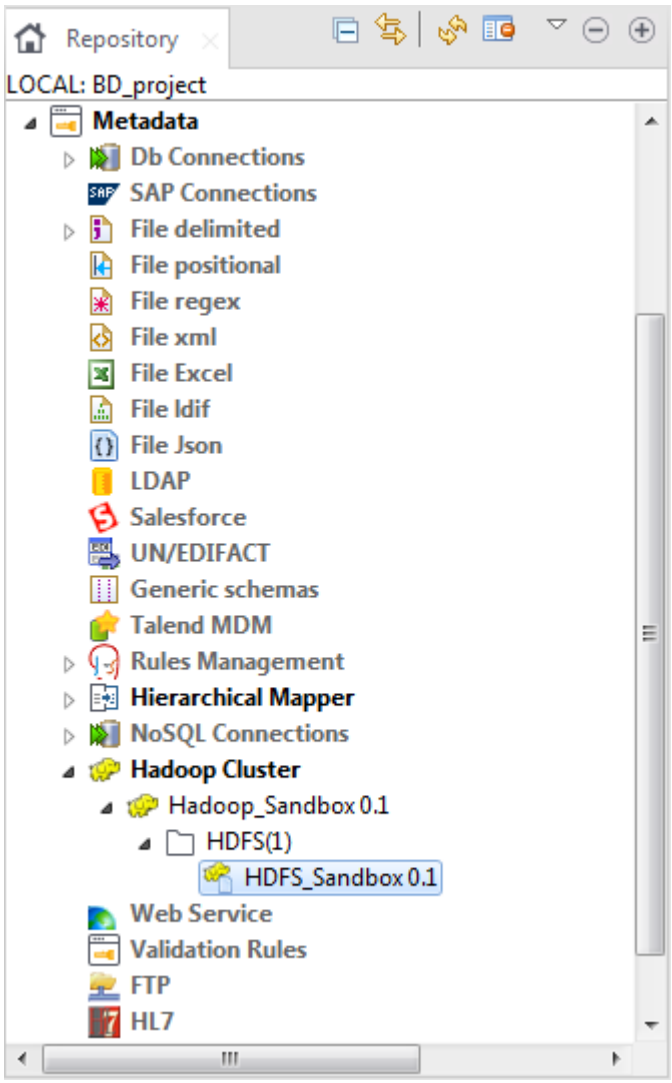
2. Si besoin, personnalisez les paramètres de la connexion HDFS et vérifiez la connexion. Comme les Jobs d'exemple fonctionnent avec les paramètres par défaut, cliquez simplement sur **Check** afin de vérifier la connexion.



3. Cliquez sur **Finish**.

Résultats

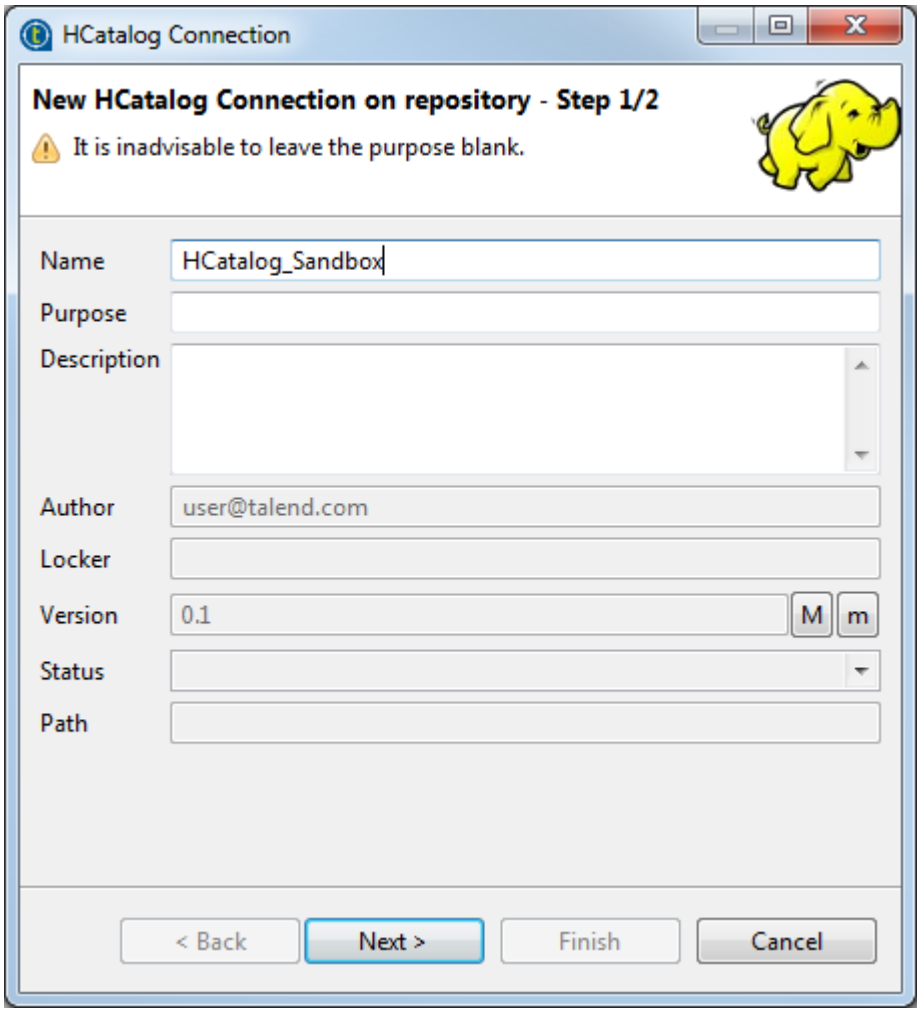
La connexion HDFS apparaît sous votre connexion au cluster Hadoop.



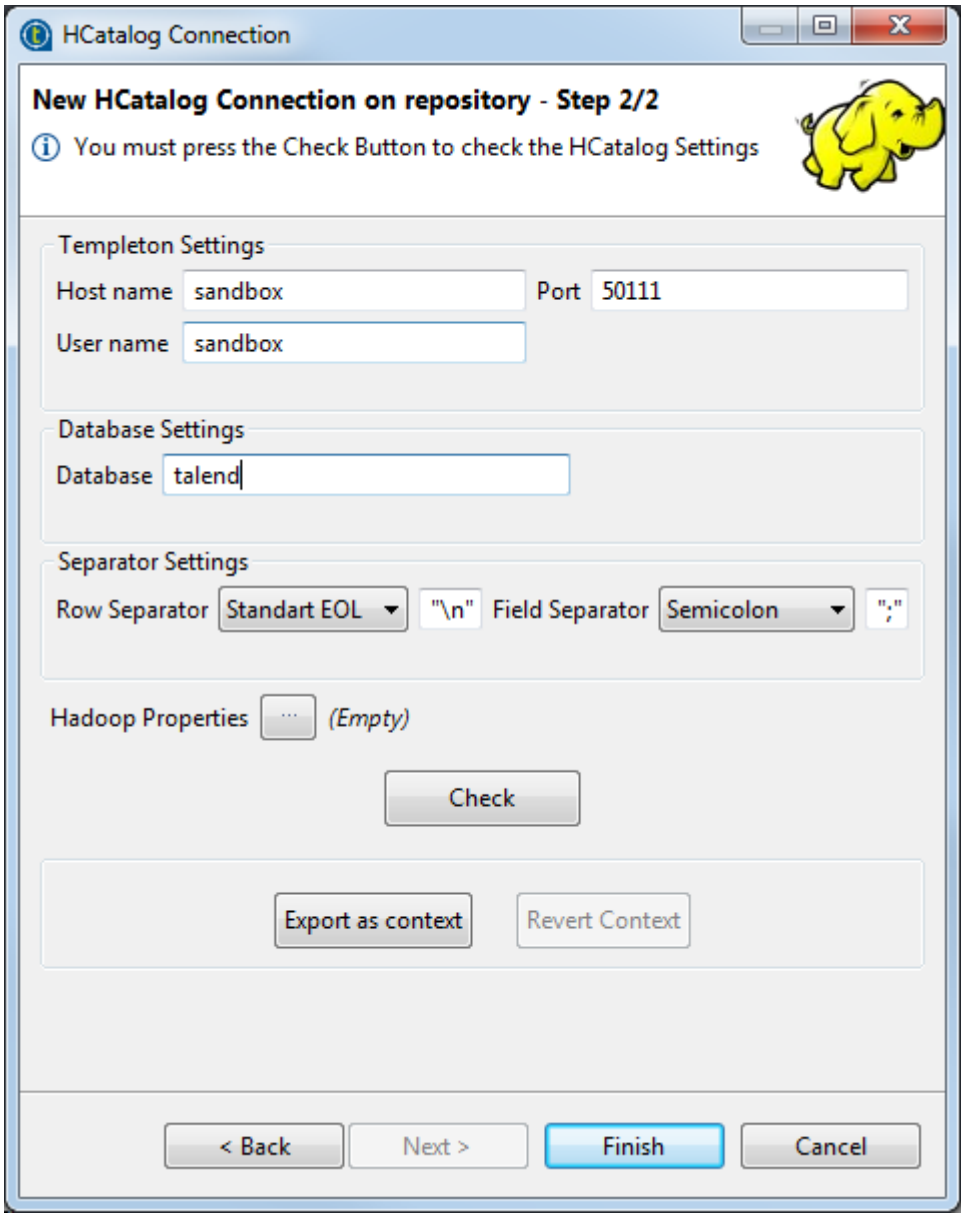
Configurer une connexion HCatalog

Procédure

1. Cliquez-droit sur la connexion au cluster Hadoop que vous venez de créer et, dans le menu contextuel, cliquez sur **Create HCatalog** afin d'ouvrir l'assistant de configuration de connexion. Donnez un nom à la connexion HCatalog, `HCatalog_Sandbox` dans cet exemple, puis cliquez sur **Next**.



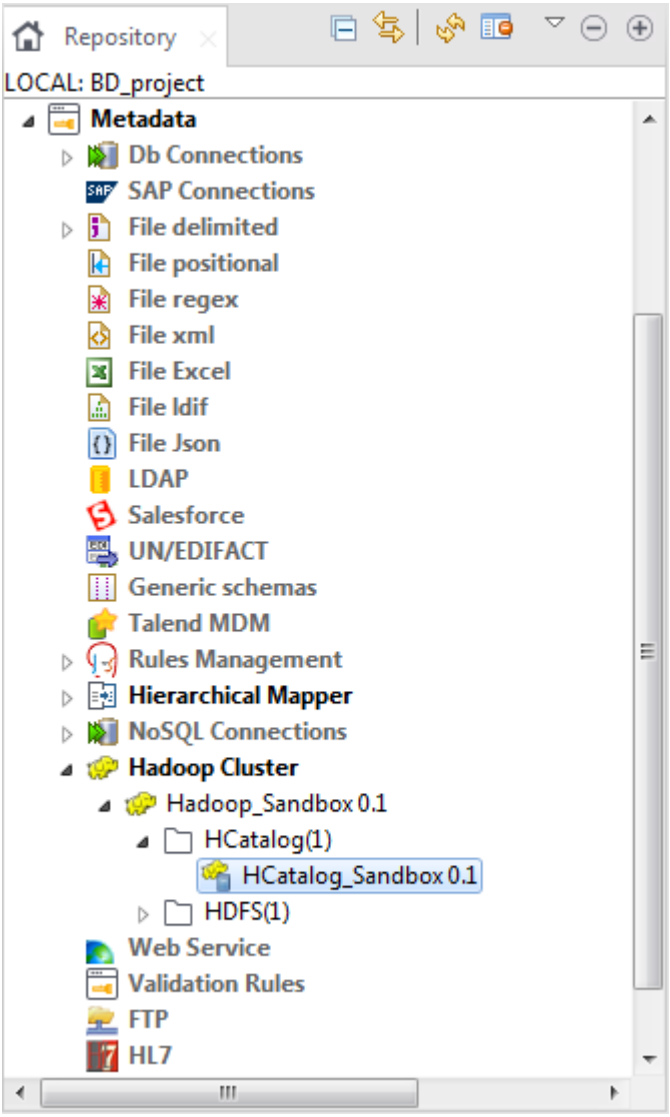
2. Dans le champ **Database**, saisissez le nom de la base de données utilisée, `talend` dans cet exemple. Cliquez sur **Check** afin de vérifier la connexion.



3. Cliquez sur **Finish**.

Résultats

La connexion HCatalog apparaît sous votre connexion au cluster Hadoop.



Créer les Jobs d'exemple

Dans cette section, vous créez six Jobs permettant d'implémenter l'exemple `ApacheWebLog` du Job de démo.

Créer le premier Job

Afin de créer le premier Job, permettant de configurer une base de données HCatalog afin de gérer le registre d'accès à analyser, procédez comme suit :

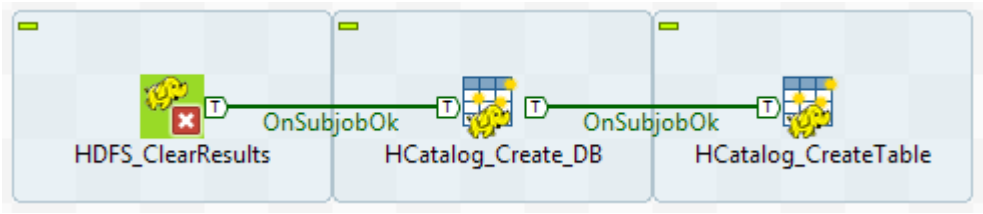
Procédure

1. Dans le **Repository**, cliquez-droit sur **Job Designs**. Cliquez ensuite sur **Create folder** afin de créer un nouveau dossier pour grouper les Jobs créés.
2. Cliquez-droit sur le dossier que vous venez de créer et cliquez sur **Create job** afin de créer votre premier Job. Nommez-le `A_HCatalog_Create` afin d'identifier son rôle et son ordre d'exécution parmi les Jobs d'exemple.

   Vous pouvez également fournir une courte description pour votre Job. Cette description apparaît en tant qu'info-bulle lorsque vous placez le pointeur de votre souris sur le Job.

3. Déposez un composant **tHDFSDelete** et deux composants **tHCatalogOperation** de la **Palette** dans l'espace de modélisation graphique.
4. Reliez les trois composants à l'aide de liens **Trigger** > **OnSubjobOk**.

   Le sous-job HDFS est utilisé pour supprimer, s'il y en a, tous les résultats précédents de cet exemple afin d'éviter toute erreur lors de l'exécution du Job. Les deux sous-jobs HCatalog créent une base de données HCatalog ainsi qu'une table HCatalog et une partition dans la table HCatalog créée, respectivement.
5. Renommez les composants afin de mieux identifier leur rôle au sein du Job.



Créer le deuxième Job

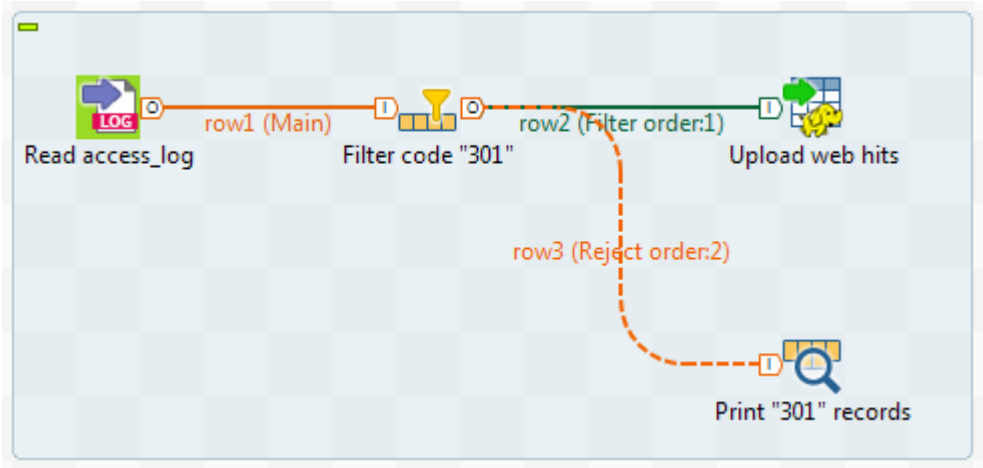
Afin de créer le deuxième Job, permettant de charger le registre d'accès dans HCatalog, procédez comme suit :

Procédure

1. Créez le nouveau Job et nommez-le `B_HCatalog_Load` afin d'identifier son rôle et son ordre d'exécution parmi les autres Jobs d'exemple.
2. Déposez un **tApacheLogInput**, un **tFilterRow**, un **tHCatalogOutput** et un **tLogRow** de la **Palette** dans l'espace de modélisation graphique.
3. Reliez le **tApacheLogInput** au **tFliterRow** à l'aide d'un lien **Row** > **Main**. Reliez ensuite le **tFilterRow** au **tHCatalogOutput** à l'aide d'un lien **Row** > **Filter**.

   Ce flux de données charge le fichier de log à analyser dans la base de données HCatalog tout en supprimant tous les enregistrements ayant le code d'erreur "301".
4. Reliez le **tFilterRow** au **tLogRow** à l'aide d'un lien **Row** > **Reject**.

   Ce flux affiche les enregistrements ayant le code d'erreur "301" dans la console.
5. Afin de mieux identifier le rôle de chaque composant, renommez-les comme suit :

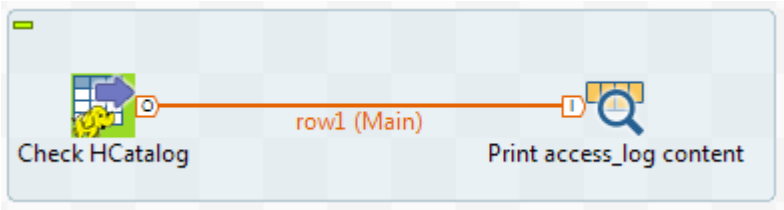


Créer le troisième Job

Afin de créer le troisième Job, permettant d'afficher le contenu du fichier chargé, procédez comme suit :

Procédure

1. Créez un nouveau Job et nommez-le `C_HCatalog_Read` afin d'identifier son rôle et son ordre d'exécution parmi les autres Jobs.
2. Déposez un **tHCatalogInput** et un **tLogRow** de la **Palette** dans l'espace de modélisation graphique. Reliez-les à l'aide d'un lien **Row** > **Main**.
3. Afin de mieux identifier le rôle de chaque composant, renommez-les comme suit :

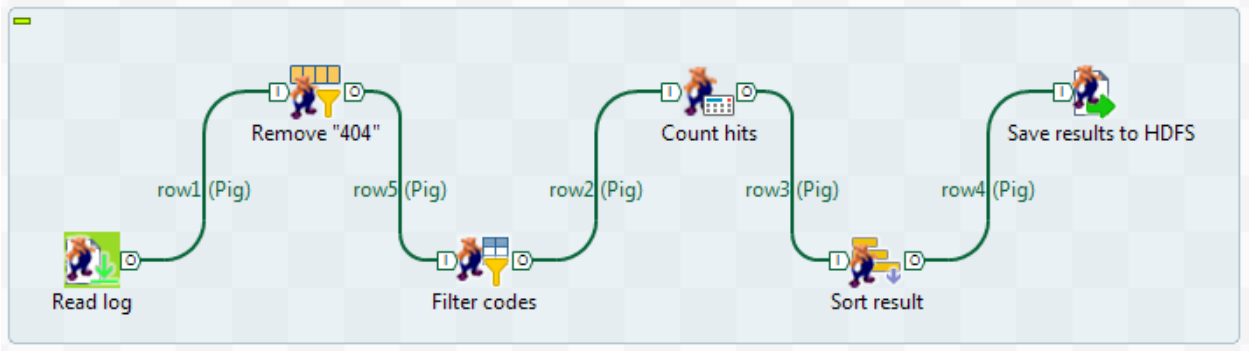


Créer le quatrième Job

Afin de créer le quatrième Job, permettant d'analyser le fichier chargé afin d'obtenir les occurrences de code dans les appels de services vers le site Web exécutés avec succès, procédez comme suit :

Procédure

1. Créez un nouveau Job et nommez-le `D_Pig_Count_Codes` afin d'identifier son rôle et son ordre d'exécution dans les Jobs d'exemple.
2. Déposez les composants suivants de la **Palette** dans l'espace de modélisation graphique :
   - un **tPigLoad**, afin de charger les données à analyser,
   - un **tPigFilterRow**, afin de supprimer du flux d'entrée les enregistrements ayant l'erreur "404",
   - un **tPigFilterColumns**, afin de sélectionner les colonnes que vous souhaitez inclure dans les résultats,
   - un **tPigAggregate**, afin de compter le nombre de visites sur le site web,
   - un **tPigSort**, afin de trier les résultats et
   - un **tPigStoreResult**, afin de sauvegarder le résultat dans HDFS.
3. Reliez ces composants à l'aide de liens **Row** > **Pig Combine** afin de former une chaîne Pig et, afin de mieux identifier leur rôle, renommez-les comme suit :

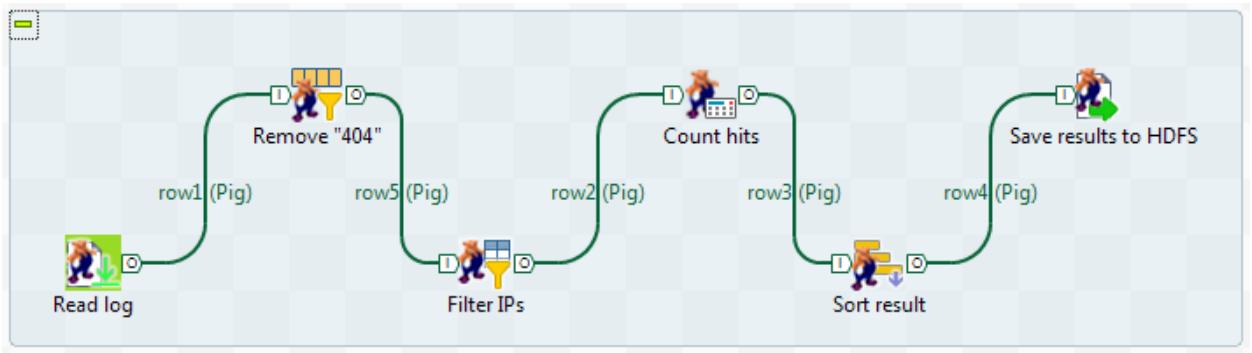


Créer le cinquième Job

Afin de créer le cinquième Job, permettant d'analyser le fichier chargé afin d'obtenir les occurrences d'adresses IP dans les appels de services vers le site Web exécutés avec succès, procédez comme suit :

Procédure

1. Cliquez-droit sur le Job précédent dans le **Repository** et cliquez sur **Duplicate**.
2. Dans la boîte de dialogue qui s'affiche, renommez le Job afin d'identifier son rôle et son ordre d'exécution parmi les Jobs d'exemple.
3. Afin d'identifier son rôle dans le Job, renommez le composant **tPigFilterColumns** comme suit :

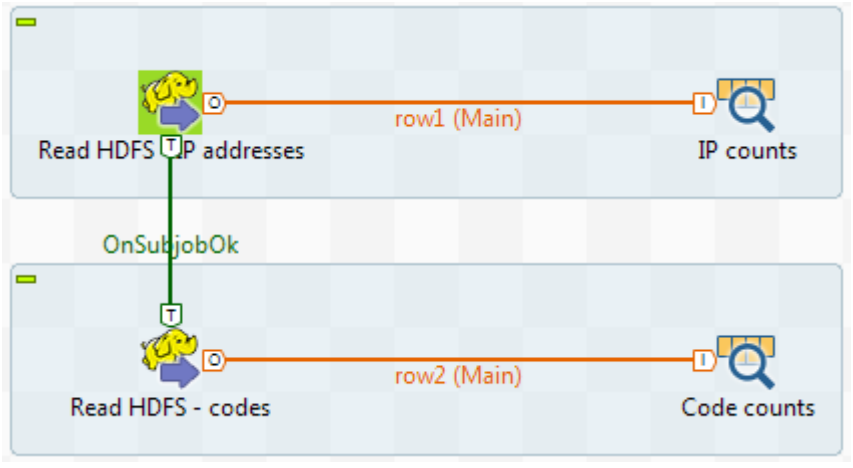


Créer le sixième Job

Afin de créer le sixième Job, permettant d'afficher les résultats de l'analyse du fichier, procédez comme suit :

Procédure

1. Créez un nouveau Job et nommez-le `F_Read_Results` afin d'identifier son rôle et son ordre d'exécution parmi les Jobs d'exemple.
2. Déposez deux **tHDFSInput** et deux **tLogRow** depuis la **Palette** dans l'espace de modélisation graphique.
3. Reliez le premier **tHDFSInput** au premier **tLogRow** et le second **tHDFSInput** au second **tLogRow** à l'aide de liens **Row** > **Main**.
4. Reliez le premier **tHDFSInput** au second **tHDFSInput** à l'aide d'un lien **Trigger** > **OnSubjobOk**.
5. Afin de mieux identifier leur rôle, renommez les composants comme suit :



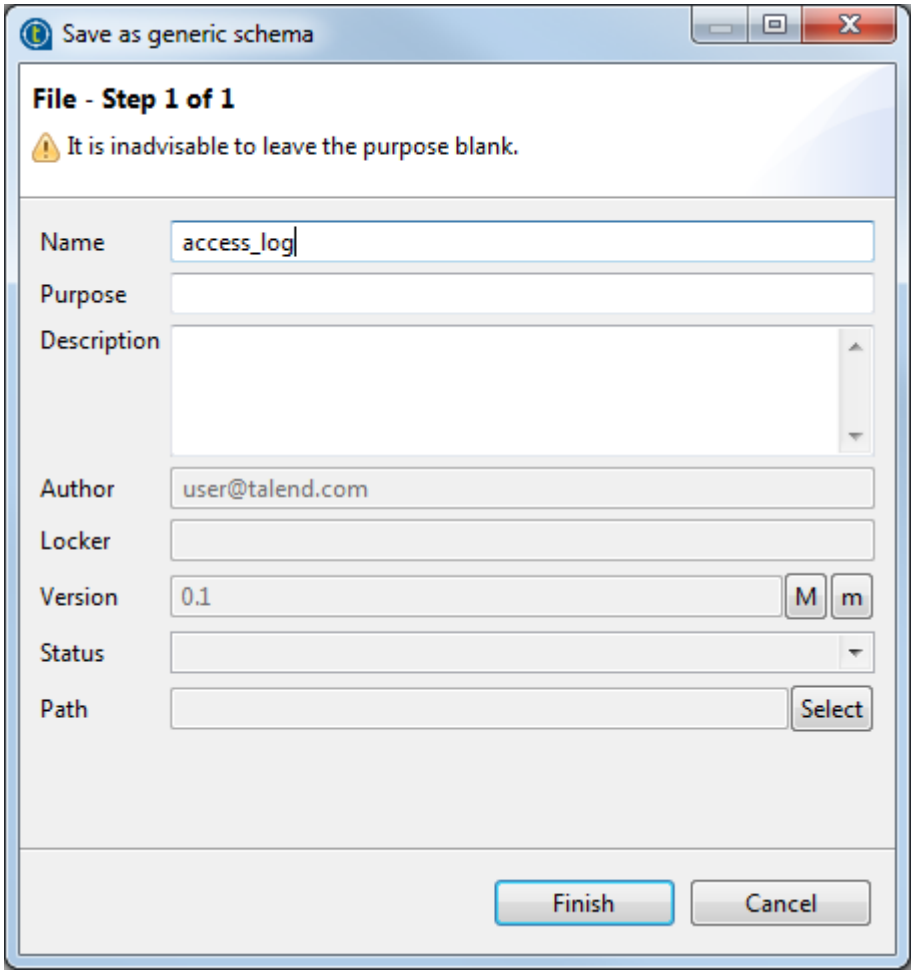
Centraliser le schéma du registre d'accès afin de le réutiliser dans la configu ration des Jobs

Afin de prendre en charge le registre d'accès à analyser dans le système Hadoop, vous devez définir le schéma approprié dans les composants concernés.

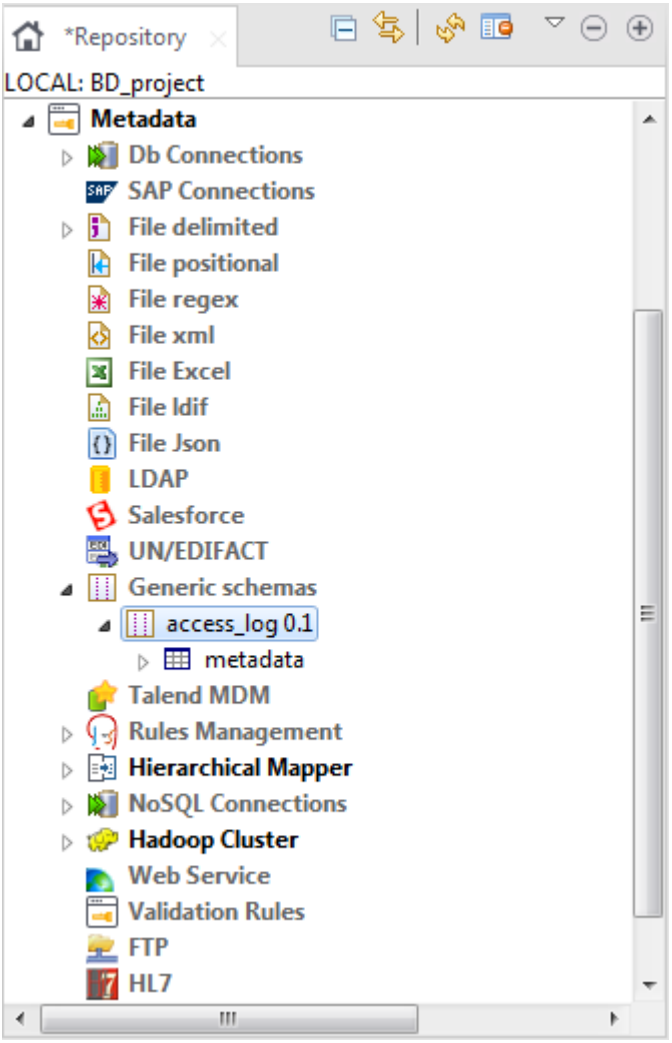
Afin de simplifier la configuration, avant de commencer à configurer les Jobs, vous pouvez sauvegarder le schéma du composant **tApacheLogInput** en lecture seule en tant que schéma générique qui peut être réutilisé dans plusieurs Jobs.

Procédure

1. Dans le Job `B_HCatalog_Read`, double-cliquez sur le **tApacheLogInput** afin d'ouvrir sa vue **Basic settings**.
2. Cliquez sur le bouton **[...]** à côté du champ **Edit schema** afin d'ouvrir la boîte de dialogue **[Schema]**.
3. Cliquez sur le bouton  afin d'ouvrir la boîte de dialogue **[Select folder]**.
4. Dans cet exemple, aucun dossier n'ayant été créé sous le nœud **Generic schemas**, cliquez simplement sur **OK** afin de fermer la boîte de dialogue et ouvrir l'assistant de configuration du schéma générique.
5. Nommez le schéma générique, `access_log` dans cet exemple, puis cliquez sur **Finish** afin de fermer l'assistant et sauvegarder votre schéma.



6. Cliquez sur **OK** afin de fermer la boîte de dialogue **[Schema]**. Le schéma générique apparaît sous le nœud **Generic schemas** du **Repository** et est prêt à être utilisé lorsque nécessaire dans la configuration de vos Jobs.



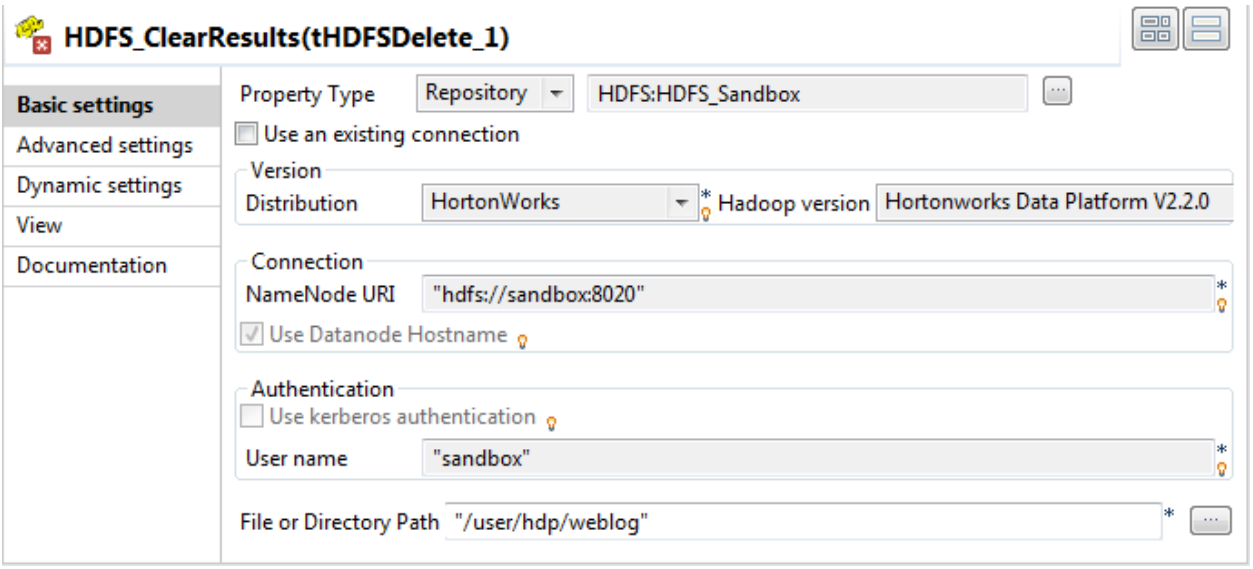
Configurer le premier Job

Cette étape détaille comment configurer le premier Job, `A_HCatalog_Create`, afin de configurer le système HCatalog pour traiter le registre d'accès.

Configurer une base de données HCatalog

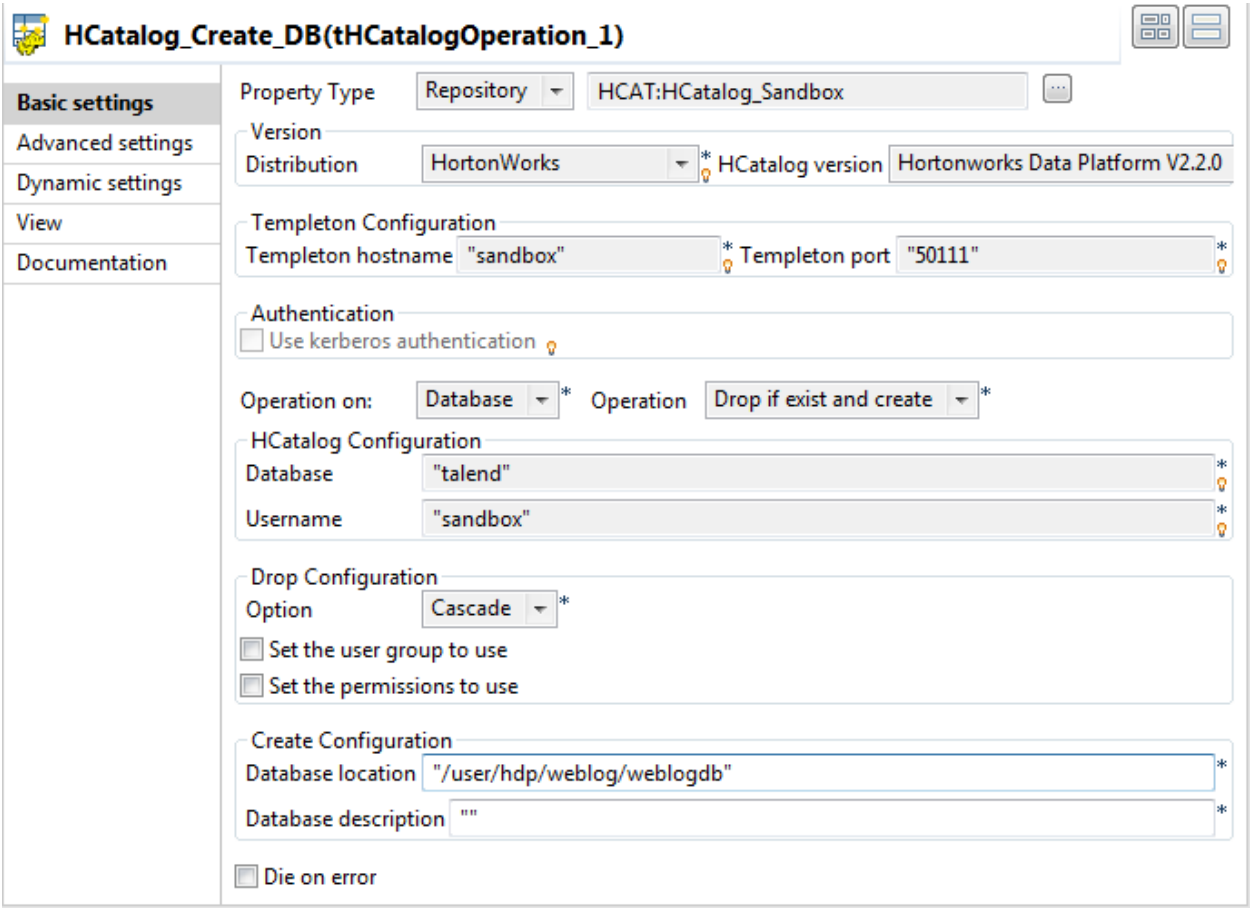
Procédure

1. Double-cliquez sur le composant **tHDFSDelete**, nommé `HDFS_ClearResults` dans cet exemple, afin d'ouvrir sa vue **Basic settings** dans l'onglet **Component**.



2. Afin d'utiliser une connexion HDFS centralisée, ouvrez la liste **Property type** et sélectionnez **Repository**. Cliquez ensuite sur le bouton **[...]** afin d'ouvrir la boîte de dialogue **[Repository Content]**.
3. Sélectionnez la connexion HDFS définie pour vous connecter au système HDFS et cliquez sur **OK**.

   Tous les détails de connexion sont automatiquement remplis dans les champs appropriés.
4. Dans le champ **File or Directory Path**, spécifiez le dossier dans lequel le registre d'accès est stocké dans HDFS, `/user/hdp/weblog` dans cet exemple.
5. Double-cliquez sur le premier composant **tHCatalogOperation**, nommé `HCatalog_Create_DB` dans cet exemple, pour ouvrir sa vue **Basic settings**.

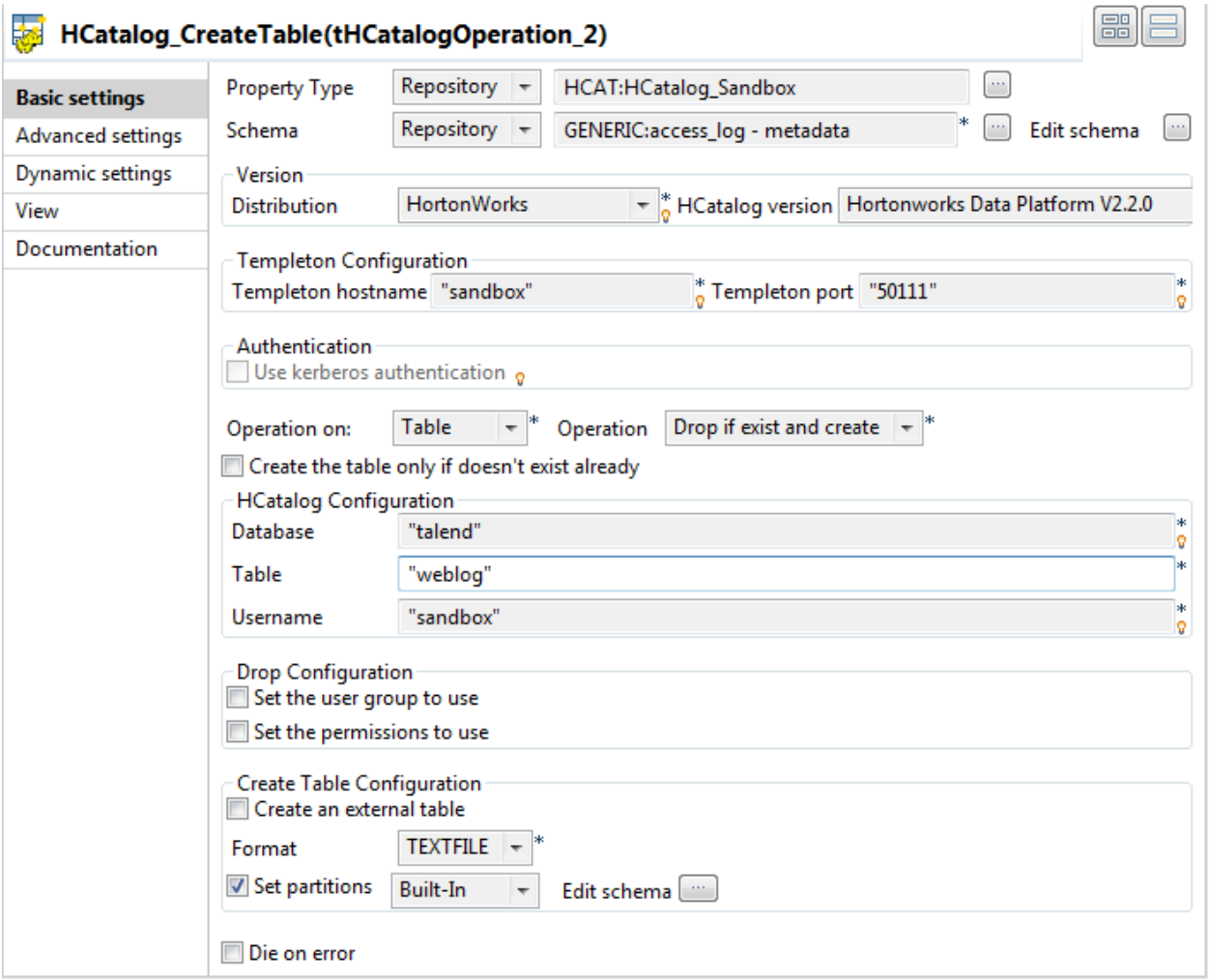


6. Afin d'utiliser une connexion HCatalog centralisée, ouvrez la liste **Property Type** et sélectionnez **Repository**. Cliquez ensuite sur le bouton **[...]** afin d'ouvrir la boîte de dialogue **[Repository Content]**.
7. Sélectionnez la connexion HCatalog définie pour vous connecter à la base de données HCatalog et cliquez sur **OK**. Tous les détails de connexion sont automatiquement remplis dans les champs appropriés.
8. Dans la liste **Operation on**, sélectionnez **Database**. Dans la liste **Operation**, sélectionnez **Drop if exist and create**.
9. Dans la liste **Option** de la zone **Drop configuration**, sélectionnez **Cascade**.
10. Dans le champ **Database location**, saisissez l'emplacement du fichier de base de données à créer dans HDFS, `/user/hdp/weblog/weblogdb` dans cet exemple.

Configurer la table HCatalog et sa partition

Procédure

1. Double-cliquez sur le second **tHCatalogOperation**, nommé `HCatalog_CreateTable` dans cet exemple, pour ouvrir sa vue **Basic settings**.



2. Définissez les même détails de connexion HCatalog que pour le premier composant **tHCatalogOperation** en utilisant la même procédure.
3. Ouvrez la liste **Schema** et sélectionnez **Repository**. Cliquez ensuite sur le bouton **[...]** à côté du champ qui apparaît dans la boîte de dialogue **[Repository Content]**. Développez les nœuds **Metadata** > **Generic schemas** > **access_log** et sélectionnez le schéma. Cliquez sur **OK** afin de

confirmer votre choix et fermer la boîte de dialogue. Le schéma générique de `access_log` est automatiquement appliqué au composant.

Vous pouvez également sélectionner directement le schéma générique de `access_log` depuis le **Repository** et le déposer sur le composant afin d'appliquer son schéma.

4. Dans la liste **Operation on**, sélectionnez **Table**. Dans la liste **Operation**, sélectionnez **Drop if exist and create**.
5. Dans le champ **Table**, saisissez un nom pour la table à créer, `weblog` dans cet exemple.
6. Cochez la case **Set partitions** et cliquez sur le bouton **[...]** à côté du champ **Edit schema** afin de configurer une partition et un schéma de partition.

   Notez que le schéma de partition ne doit contenir aucun nom de colonne défini dans le schéma de la table. Dans cet exemple, la colonne du schéma de partition se nomme `ipaddresses`.
7. Une fois les paramètre des composants définis, appuyez sur **Ctrl+S** afin de sauvegarder la configuration du Job.

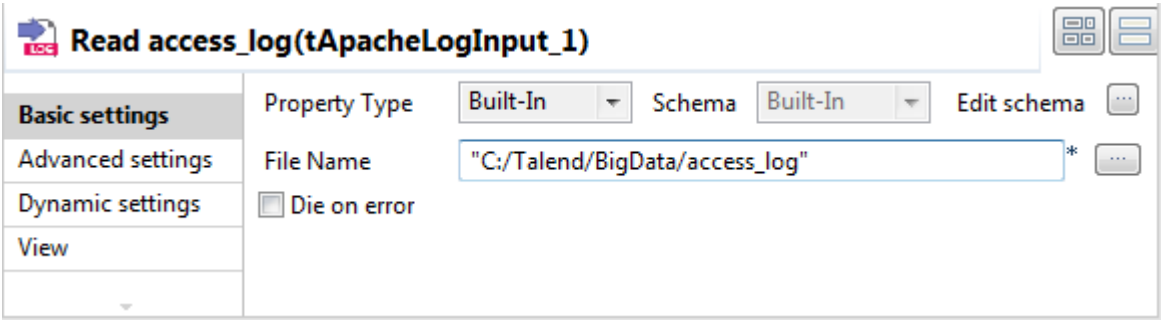
Charger le fichier de log dans HCatalog

Cette étape détaille comment configurer le deuxième Job,`B_HCatalog_Load`, afin de charger le fichier de registre dans le système Hadoop.

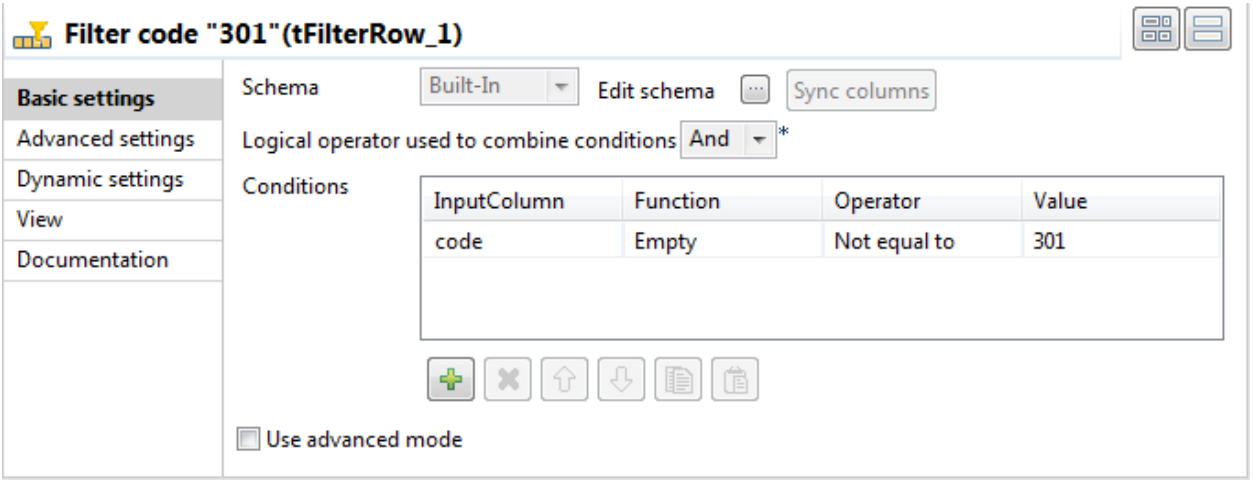
Procédure

1. Double-cliquez sur le composant **tApacheLogInput** pour ouvrir sa vue **Basic settings** et spécifiez le chemin d'accès au fichier de log à charger, dans le champ **File Name**.

   Dans cette exemple, le fichier de log `access_log` est stocké dans le dossier `C:/Talend/BigData`.

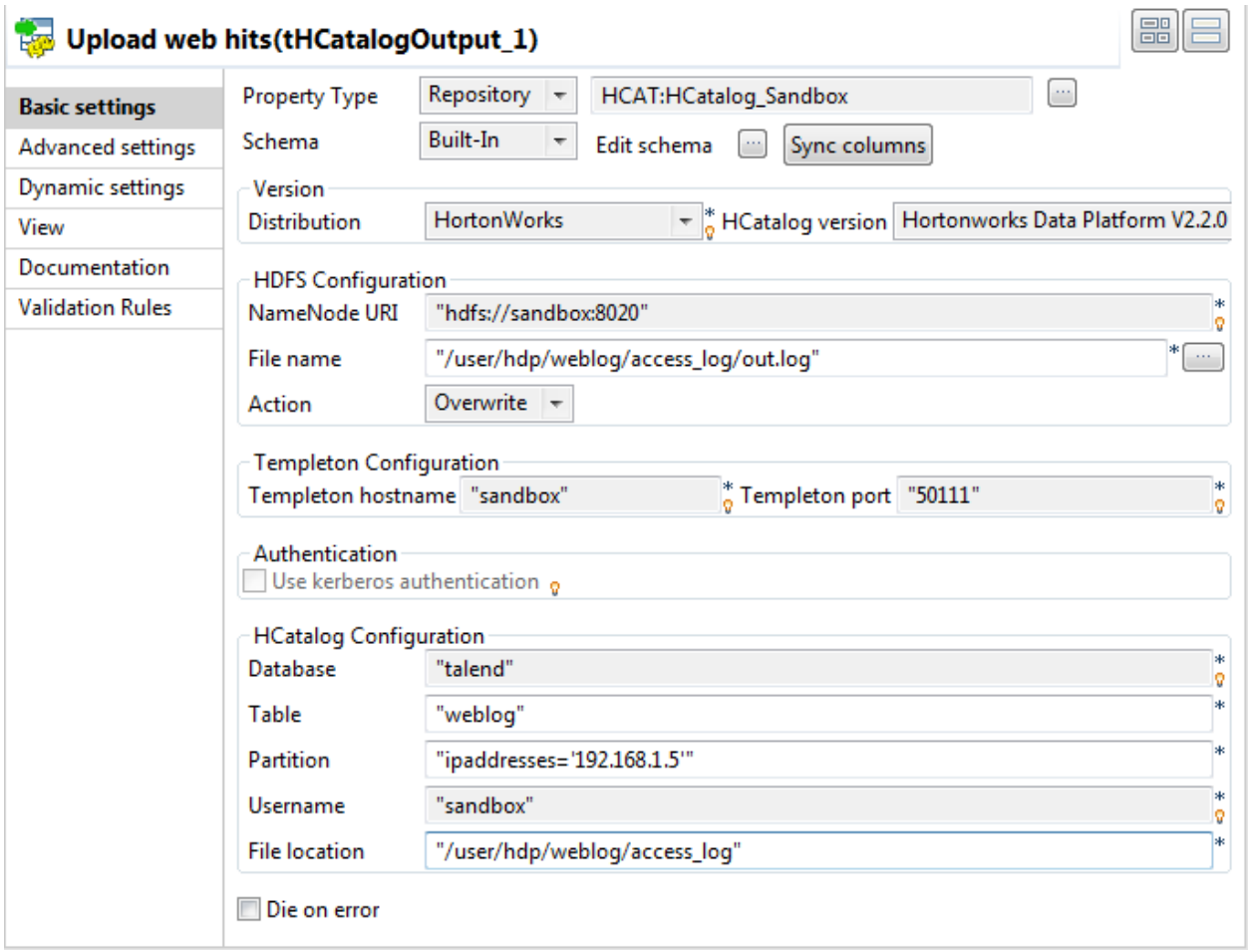


2. Double-cliquez sur le **tFilterRow** afin d'ouvrir sa vue **Basic settings**.



3. Dans la liste **Logical operator used to combine conditions**, sélectionnez **AND**.

4. Cliquez sur le bouton **[+]** afin d'ajouter une ligne dans la table **Filter configuration**. Définissez les paramètres de filtre afin d'envoyer les enregistrements contenant le code "301" vers le flux **Reject** et passer le reste des enregistrements au flux **Filter** :
   a) Dans le champ **InputColumn**, sélectionnez la colonne `code` du schéma.
   b) Dans le champ **Operator**, sélectionnez **Not equal to**.
   c) Dans le champ **Value**, saisissez `301`.
5. Double-cliquez sur le **tHCatalogOutput** pour ouvrir la vue **Basic settings**.



6. Afin d'utiliser une connexion à HCatalog centralisée, ouvrez la liste **Property Type** et sélectionnez **Repository**. Cliquez ensuite **[...]** afin d'ouvrir la boîte de dialogue **[Repository Content]**.
7. Sélectionnez la connexion à HCatalog définie pour la connexion à la base de données HCatalog puis cliquez sur **OK**.

   Tous les détails de connexion sont automatiquement saisis dans les champs appropriés.
8. Cliquez sur le bouton **[...]** pour vérifier que le schéma a bien été propagé depuis le composant précédent. Si nécessaire, cliquez sur le bouton **Sync columns** afin de récupérer le schéma.
9. Dans la liste **Action**, sélectionnez **Create** pour créer le fichier, ou **Overwrite** si le fichier existe déjà.
10. Dans le champ **Partition**, saisissez, entre guillemets doubles, la paire de partition nom-valeur, `ipaddresses='192.168.1.15'` dans cet exemple.
11. Dans le champ **File location**, saisissez l'emplacement où sauvegarder les données, `/user/hdp/weblog/access_log` dans cet exemple.
12. Double-cliquez sur le **tLogRow** afin d'ouvrir sa vue **Basic settings**. Sélectionnez l'option **Vertical** afin d'afficher chaque ligne du contenu de sortie sous forme de liste pour une meilleure lisibilité.

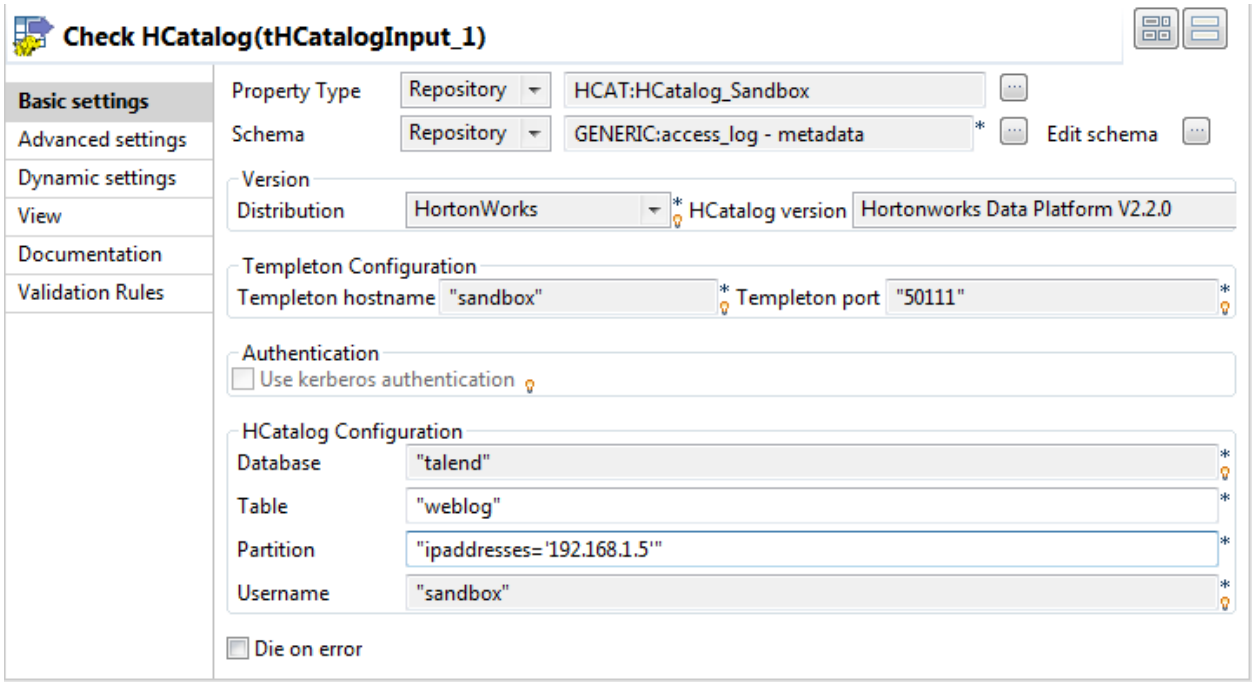
13. Une fois les paramètres des composants définis, appuyez sur **Ctrl+S** pour sauvegarder la configuration du Job.

Configurer le troisième Job

Cette étape détaille comment configurer le troisième Job, `C_HCatalog_Read`, afin de vérifier le contenu du fichier chargé dans HCatalog.

Procédure

1. Double-cliquez sur le composant **tHCatalogInput** pour ouvrir sa vue **Basic settings** dans l'onglet **Component**.



2. Afin d'utiliser une connexion à HCatalog centralisée, ouvrez la liste **Property Type** et sélectionnez **Repository**. Cliquez ensuite **[...]** afin d'ouvrir la boîte de dialogue **[Repository Content]**.
3. Sélectionnez la connexion à HCatalog définie pour la connexion à la base de données HCatalog puis cliquez sur **OK**.

   Tous les détails de connexion sont automatiquement saisis dans les champs appropriés.
4. Ouvrez la liste **Schema** et sélectionnez **Repository**. Cliquez ensuite sur le bouton **[...]** à côté du champ qui s'affiche afin d'ouvrir la boîte de dialogue **[Repository Content]**. Développez le nœud **Metadata** > **Generic schemas** > **access_log** et sélectionnez le schéma. Cliquez sur **OK** afin de confirmer votre sélection et fermer la boîte de dialogue. Le schéma générique de `access_log` est automatiquement appliqué au composant.

   Vous pouvez également sélectionner directement `access_log` dans le **Repository** et le déposer sur ce composant afin d'appliquer le schéma.
5. Dans la vue **Basic settings** du **tLogRow**, sélectionnez l'option **Vertical** afin d'afficher pour chaque ligne la clé et la valeur, à l'exécution du Job.
6. Une fois les paramètres des composants définis, appuyez sur **Ctrl+S** pour sauvegarder la configuration du Job.

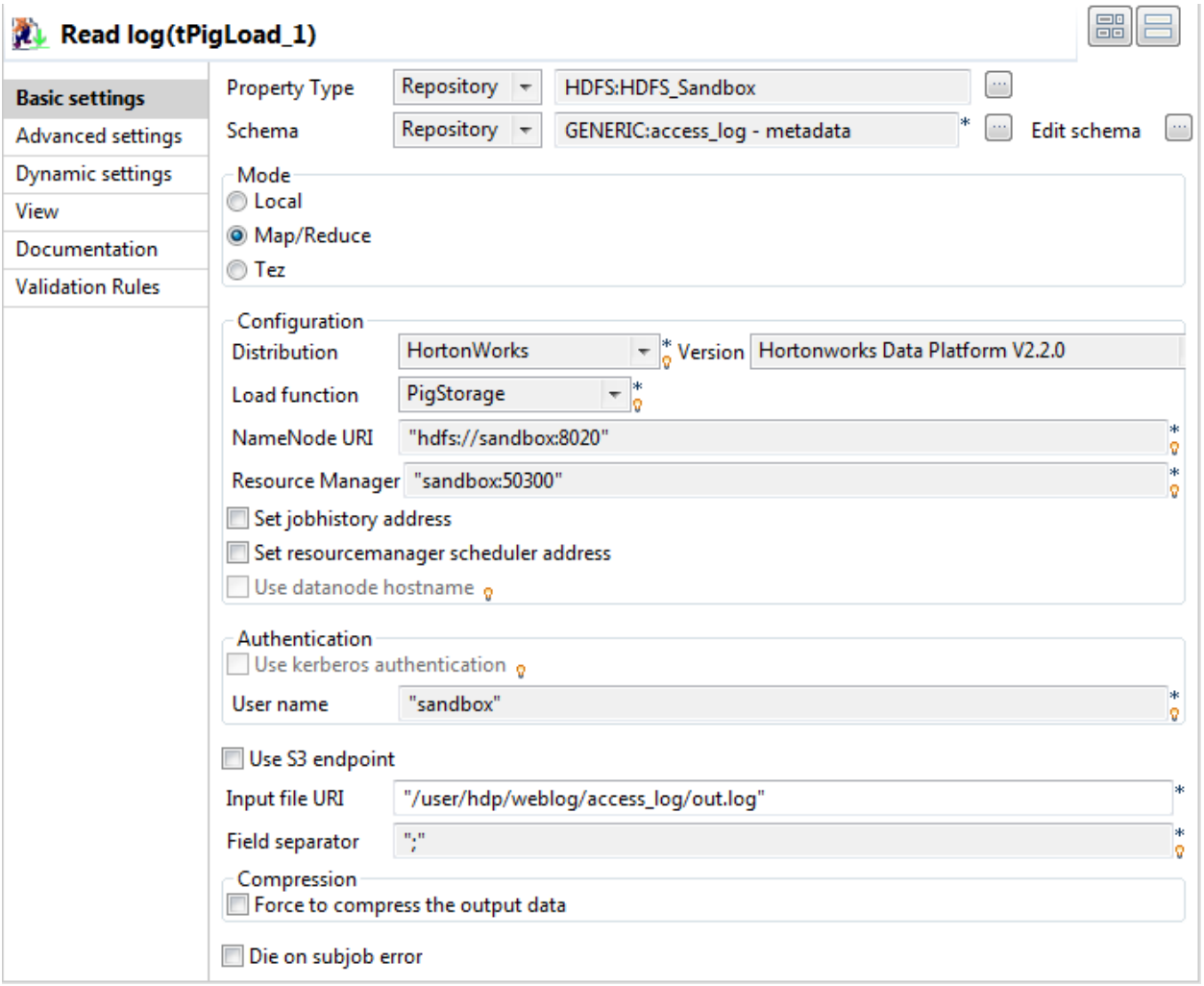
Configurer le quatrième Job

Dans cette étape, vous allez configurer le quatrième Job, `D_Pig_Count_Codes`, afin d'analyser le fichier de registre chargé utilisant une chaîne Pig pour d'obtenir les codes des appels de services exécutés avec succès et le nombre de visites sur le site Web.

Lire le fichier de registre à analyser dans la chaîne Pig

Procédure

1. Double-cliquez sur le **tPigLoad** afin d'ouvrir sa vue **Basic settings**.



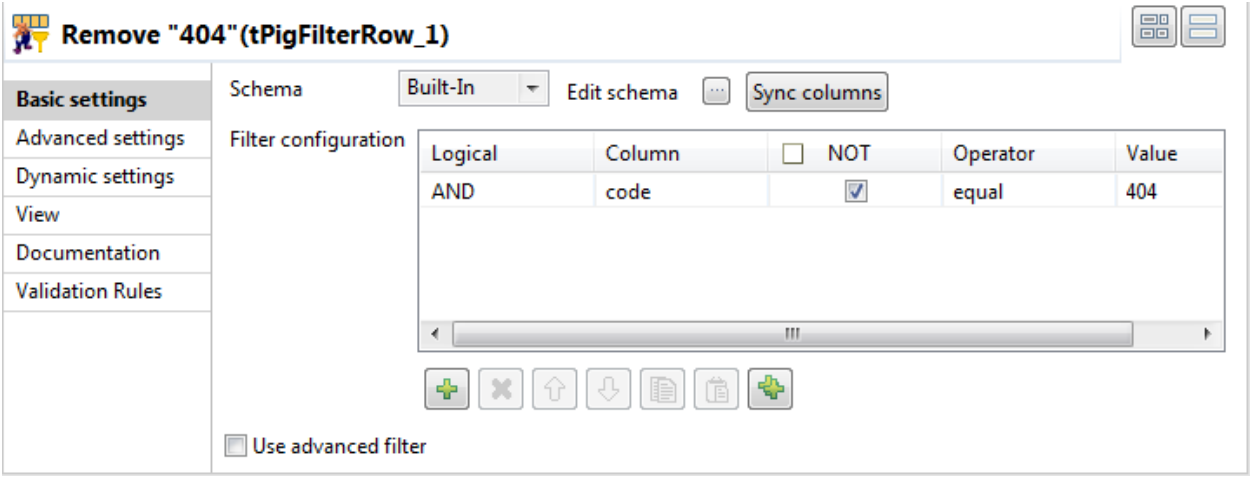
2. Afin d'utiliser une connexion à HCatalog centralisée, ouvrez la liste **Property Type** et sélectionnez **Repository**. Cliquez ensuite **[...]** afin d'ouvrir la boîte de dialogue **[Repository Content]**.
3. Sélectionnez la connexion à HCatalog définie pour la connexion à la base de données HCatalog puis cliquez sur **OK**.

   Tous les détails de connexion sont automatiquement saisis dans les champs appropriés.
4. Sélectionnez le schéma générique de `access_log` depuis le **Repository** et déposez-le sur le composant afin d'appliquer le schéma.
5. Dans la liste **Load function**, sélectionnez **PigStorage** et saisissez, dans le champ **Input file URI**, le chemin défini dans le Job précédent, `/user/hdp/weblog/access_log/out.log` dans cet exemple.

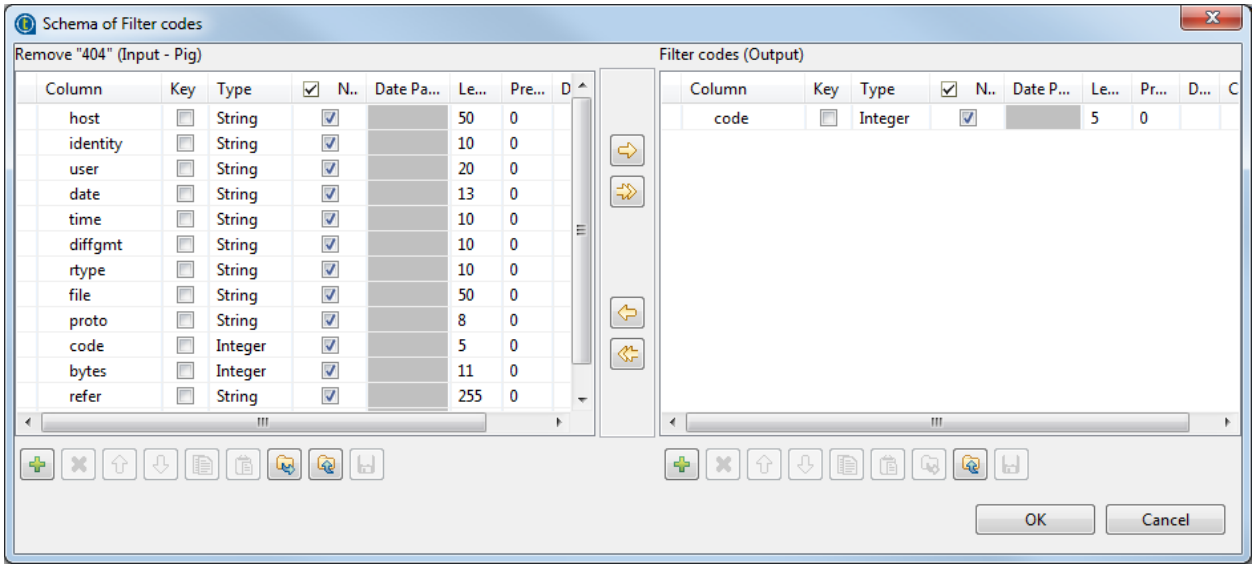
Analyser le fichier chargé et sauvegarder le résultat

Procédure

1. Dans la vue **Basic settings** du composant **tPigFilterRow**, cliquez sur le bouton **[+]** pour ajouter une ligne à la table **Filter configuration** et configurez les paramètres de filtre, afin de supprimer les enr egistrements contenant le code 404 et de passer les autres enregistrements dans le flux de sortie :
   a) Dans le champ **Logical**, sélectionnez **AND**.
   b) Dans le champ **Column**, sélectionnez la colonne `code` du schéma.
   c) Cochez la case **NOT**
   d) Dans le champ **Operator**, sélectionnez **equal**.
   e) Dans le champ **Value**, saisissez `404`.



2. Dans la vue **Basic settings** du **tPigFilterColumns**, cliquez sur le bouton **[...]** pour ouvrir la boîte de dialogue **[Schema]**. Sélectionnez la colonne `code` du panneau **Input** et cliquez sur la flèche simple pour copier la colonne dans le panneau **Output**. Cela permet de passer les informations de la colo nne `code` dans le flux de sortie. Cliquez sur **OK** afin de confirmer les paramètres du schéma de sortie et fermer la boîte de dialogue.



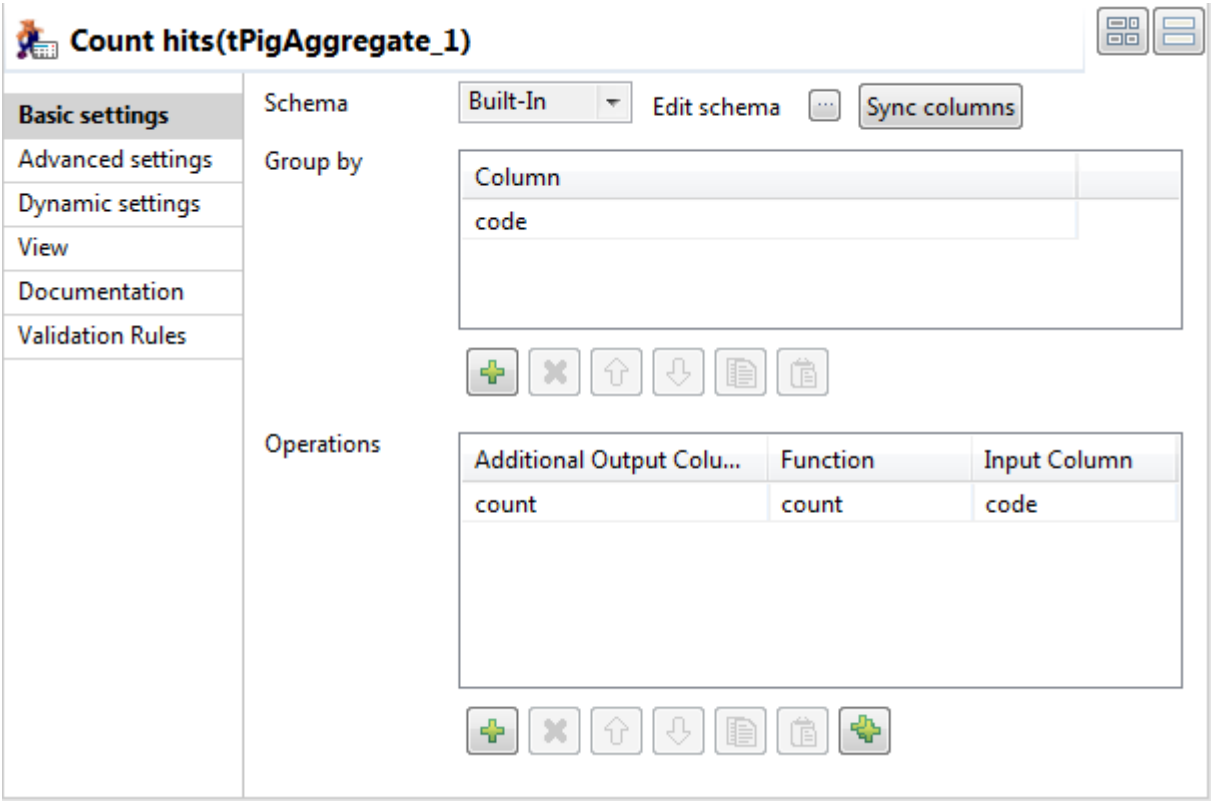
3. Dans la vue **Basic settings** du composant **tPigAggregate**, cliquez sur le bouton **Sync columns** afin de récupérer le schéma du composant précédent et propagez-le au composant suivant.

4. Cliquez sur le bouton **[...]** à côté du champ **Edit schema** afin d'ouvrir la boîte de dialogue **[Schema]** et ajoutez une colonne : `count`.

   Cette colonne stocke le nombre d'occurrences de chaque code des appels de services exécutés avec succès.

5. Configurez les paramètres suivants afin de compter le nombre d'occurrences de chaque code :
   a) Dans la zone **Group by**, cliquez sur le bouton **[+]** pour ajouter une ligne à la table et sélectionnez la colonne `count` dans le champ **Column**.

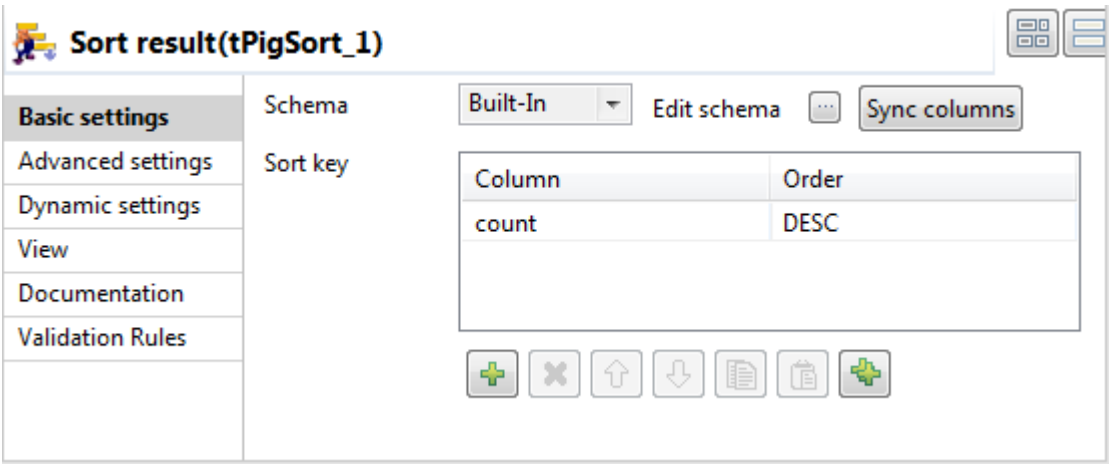
   b) Dans la zone **Operations**, cliquez sur le bouton **[+]** pour ajouter une ligne à la table et sélectionnez la colonne `count` dans le champ **Additional Output Column**, sélectionnez **count** dans la fonction **Function** et sélectionnez la colonne `code` dans le champ **Input Column**.



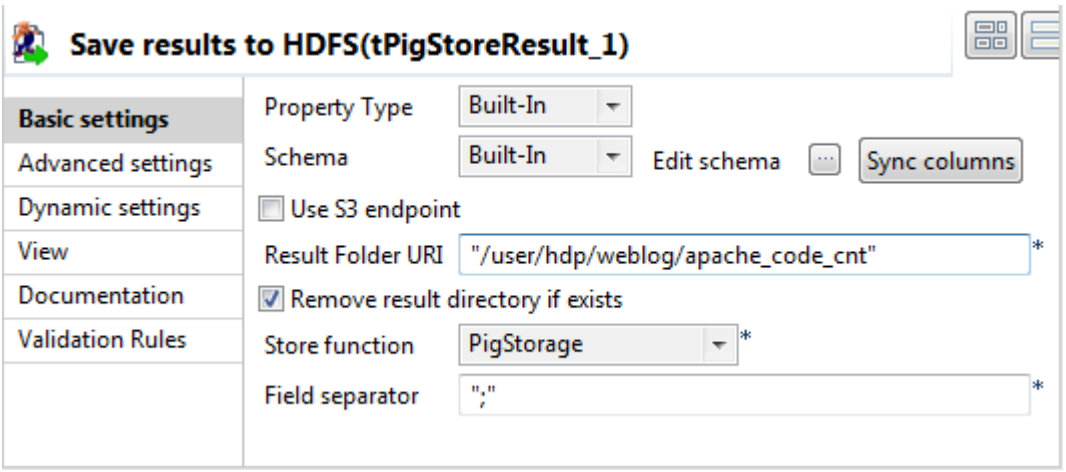
6. Dans la vue **Basic settings** du composant **tPigSort**, configurez les paramètres de tri, afin de trier les données à passer :
   a) Cliquez sur le bouton **[+]** pour ajouter une ligne à la table **Sort key**.

   b) Dans le champ **Column**, sélectionnez **count** pour définir la colonne `count` comme clé.

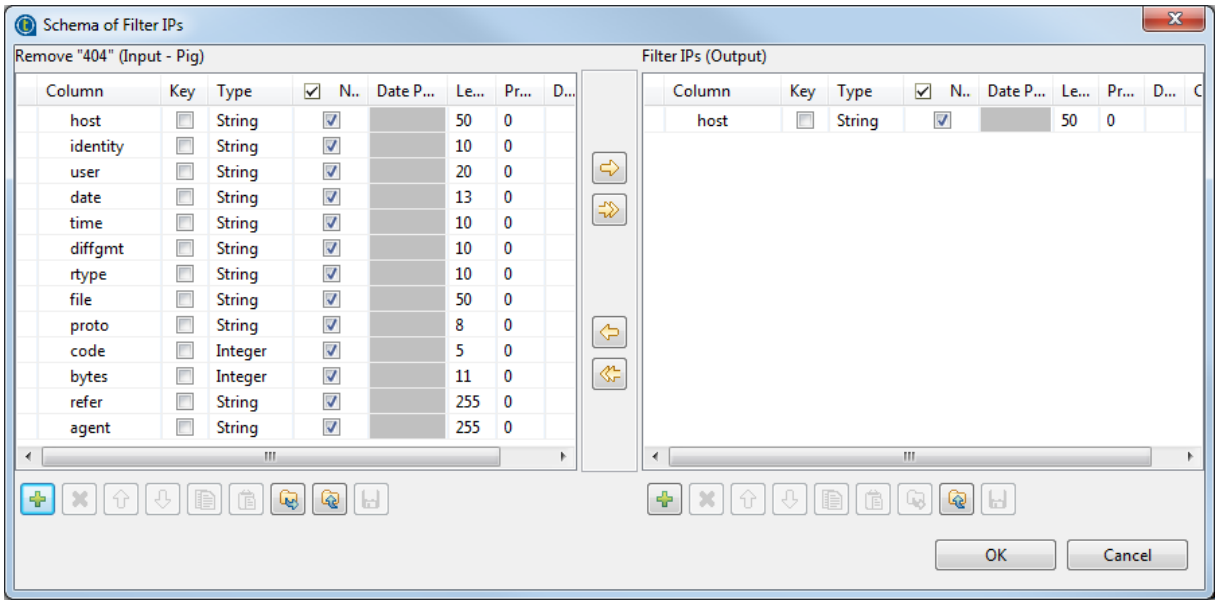
   c) Dans le champ **Order**, sélectionnez **DESC** pour trier les données en ordre descendant.



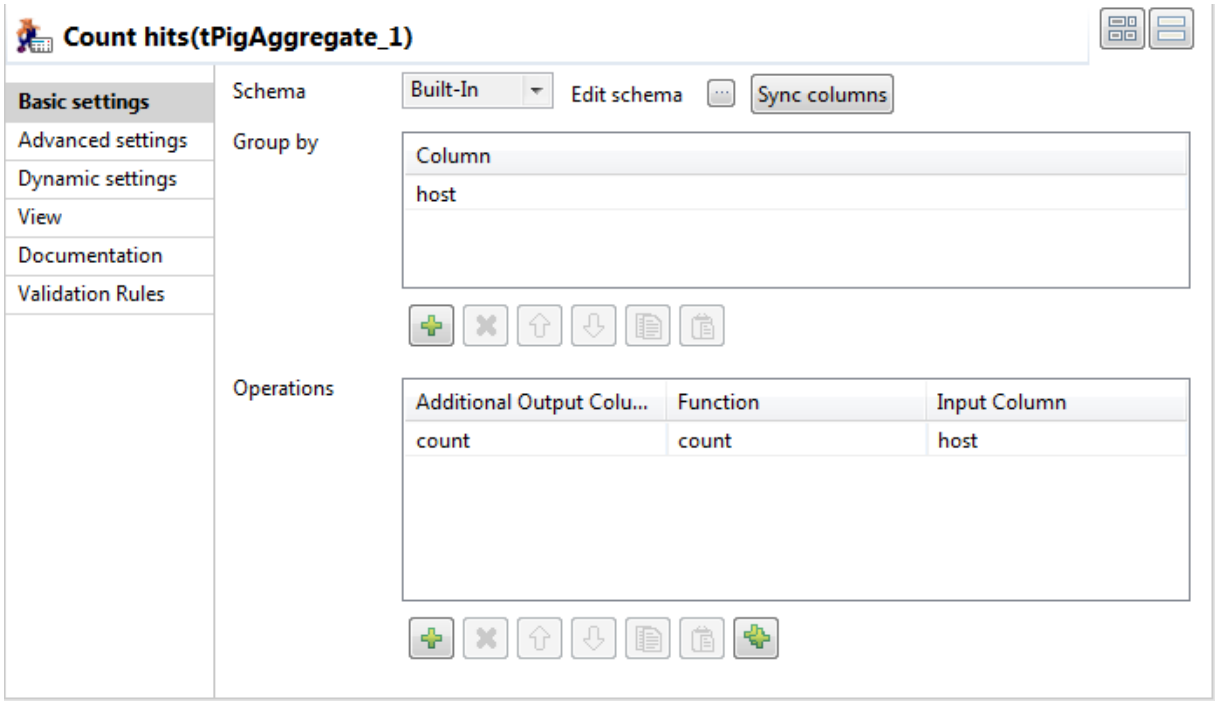
7. Dans la vue **Basic settings** du **tPigStoreResult**, configurez les propriétés du composant afin de charger les données de résultats à l'emplacement spécifié dans le système Hadoop :
   a) Cliquez sur le bouton **Sync columns** afin de récupérer le schéma du composant précédent.
   b) Dans le champ **Result file URI**, saisissez le chemin d'accès au fichier de résultats, `/user/hdp/weblog/apache_code_cnt` dans cet exemple.
   c) Dans la liste **Store function**, sélectionnez **PigStorage**.
   d) Si nécessaire, cochez la case **Remove result directory if exists**.



8. Sauvegardez le schéma de ce composant en tant que schéma générique dans le **Repository** afin de pouvoir le réutiliser facilement dans le dernier Job, comme dans Centraliser le schéma du registre d'accès afin de le réutiliser dans la configuration des Jobs à la page 17. Nommez ce schéma générique `code_count`.
9. Dans cette étape, vous allez configurer le cinquième Job, `E_Pig_Count_IPs`, permettant d'analyser le fichier chargé à l'aide d'une chaîne Pig similaire à celle utilisée dans le Job précédent. Il permet également d'obtenir le nombre d'occurrences d'adresses IP dans les appels de services vers le site Web exécutés avec succès. Vous pouvez utiliser les paramètres du Job précédent avec les différences suivantes :
   a) Dans la boîte de dialogue **[Schema]** du **tPigFilterColumns**, copiez la colonne `host`, au lieu de la colonne `code`, du panneau **Input** vers le panneau **Output**.



b) Dans le **tPigAggregate**, sélectionnez la colonne `host` dans le champ **Column** de la table **Group by** et dans le champ **Input Column** de la table **Operations**.



c) Dans le **tPigStoreResult**, saisissez `/user/hdp/weblog/apache_ip_cnt` dans le champ **Result file URI**.

d) A partir du schéma du **tPigStoreResult**, sauvegardez un schéma générique nommé `ip_count` dans le **Repository** afin de le réutiliser facilement dans le dernier Job.

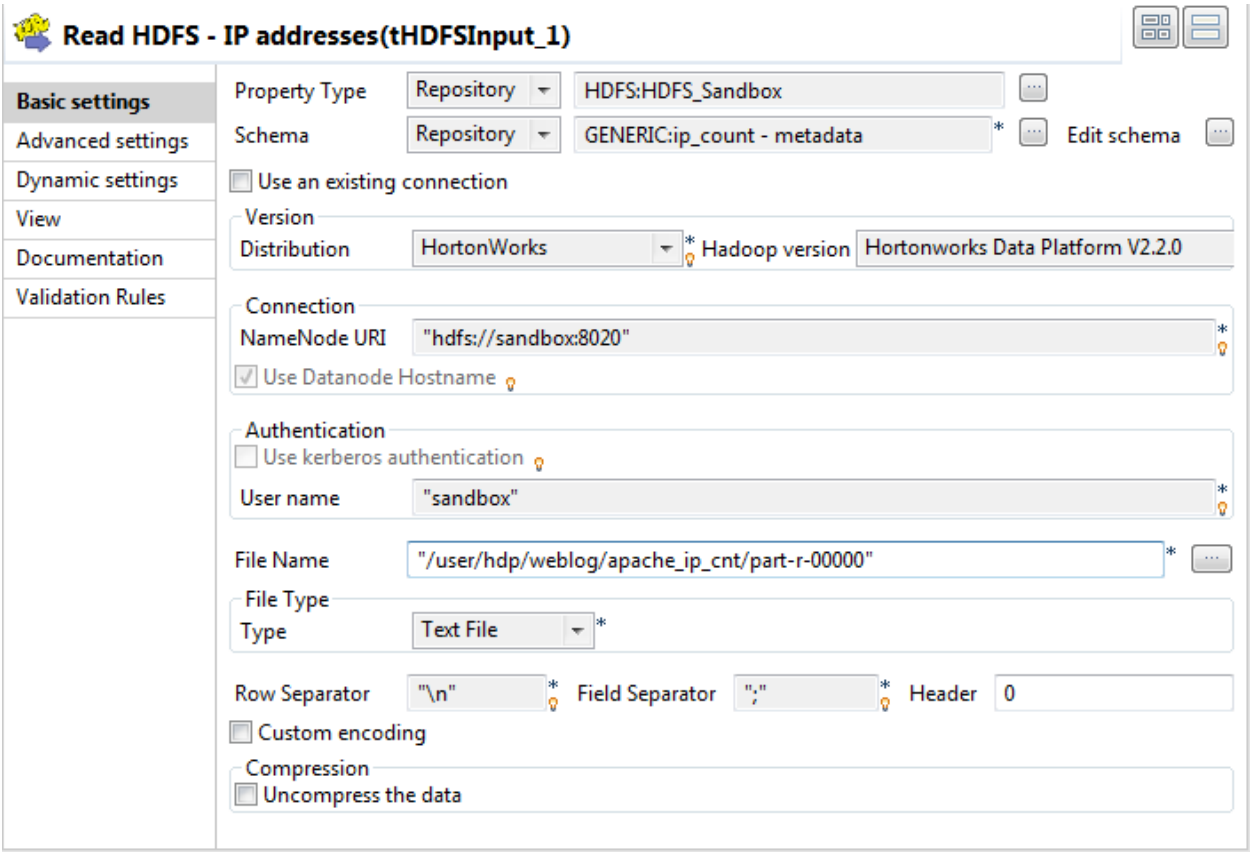
10. Une fois les paramètres des composants définis, appuyez sur **Ctrl+S** pour sauvegarder la configuration du Job.

Configurer le dernier Job

Dans cette étape, vous configurez de dernier Job, `F_Read_Results`, afin de lire les données de résultats depuis Hadoop et les afficher dans la console du système.

Procédure

1. Double-cliquez sur le premier **tHDFSInput** pour ouvrir sa vue **Basic settings**.



2. Afin d'utiliser une connexion à HDFS centralisée, ouvrez la liste **Property Type** et sélectionnez **Repository**. Cliquez ensuite **[...]** afin d'ouvrir la boîte de dialogue **[Repository Content]**.
3. Sélectionnez la connexion à HDFS définie pour la connexion à la base de données HDFS puis cliquez sur **OK**.

   Tous les détails de connexion sont automatiquement saisis dans les champs appropriés.
4. Appliquez le schéma générique `ip_count` à ce composant. Le schéma doit contenir deux co lonnes, `host` (de type **String** et d'une longueur de `50` caractères) et `count` (de type **Integer** et d'une longueur de `5` caractères),
5. Dans le champ **File Name**, saisissez le chemin d'accès vers le fichier de résultats dans HDFS, `/user/hdp/weblog/apache_ip_cnt/part-r-00000` dans cet exemple.
6. Dans la liste **Type**, sélectionnez le type de fichier à lire, **Text File** dans cet exemple.
7. Dans la vue **Basic settings** du **tLogRow**, sélectionnez l'option **Table** pour une meilleure lisibilité des résultats.
8. Configurez l'autre sous-job de la même manière. Cependant, dans le second **tHDFSInput** :
   a) Appliquez le schéma générique `code_count` ou configurez manuellement le schéma de ce composant afin qu'il contienne deux colonnes : `code` (de type **Integer** et d'une longueur de `5` caractères) et `count` (de type **Integer** et d'une longueur de `5` caractères).

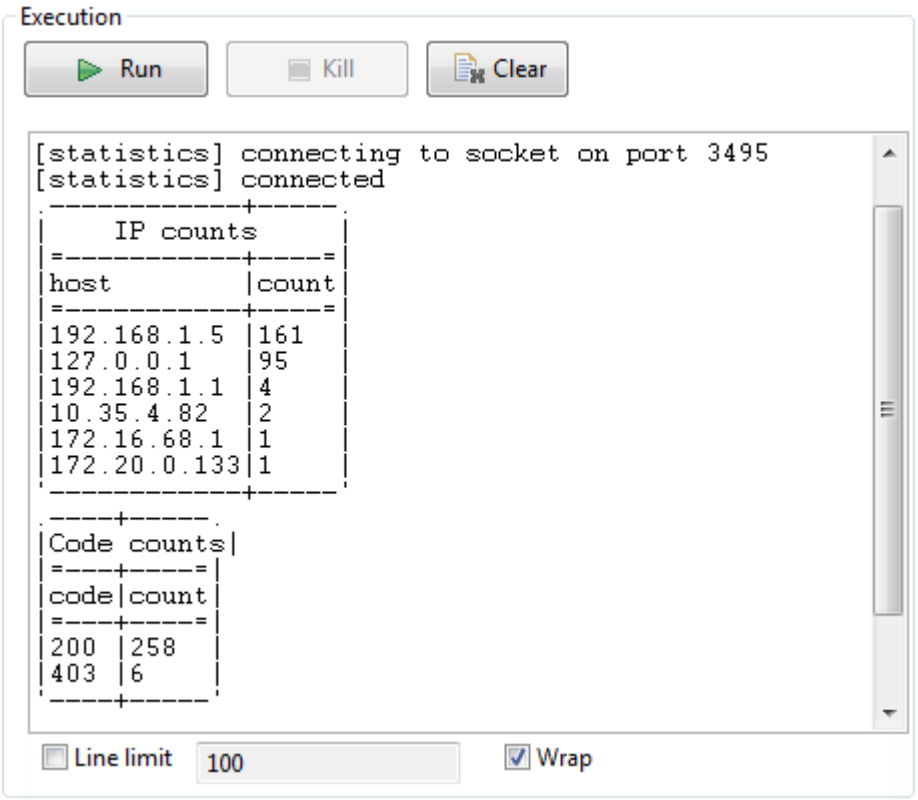
   b) Dans le champ **File Name**, saisissez `/user/hdp/weblog/apache_code_cnt/part-r-00000`.
9. Une fois les paramètres des composants définis, appuyez sur **Ctrl+S** pour sauvegarder la configuration du Job.

Exécuter les Jobs

Après avoir configuré les six Jobs, cliquez sur le bouton **Run** dans l'onglet **Run** ou appuyez sur **F6** afin de les exécuter un à un, dans l'ordre alphabétique.

Vous pouvez observer le résultat des exécutions dans la console de chaque Job.

Après réussite de l'exécution du dernier Job, la console système affiche les adresses IP ainsi que les appels de services correctement exécutés et le nombre de visites sur le site Web pour chaque adresse IP.

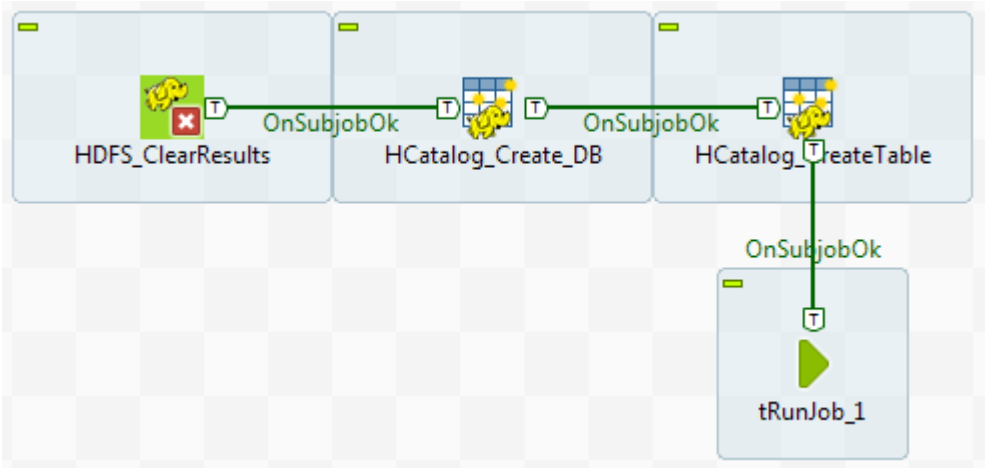


Exécuter les Jobs

Il est également possible d'exécuter tous les Jobs, dans le bon ordre, en un seul clic. Pour ce faire, procédez comme suit :

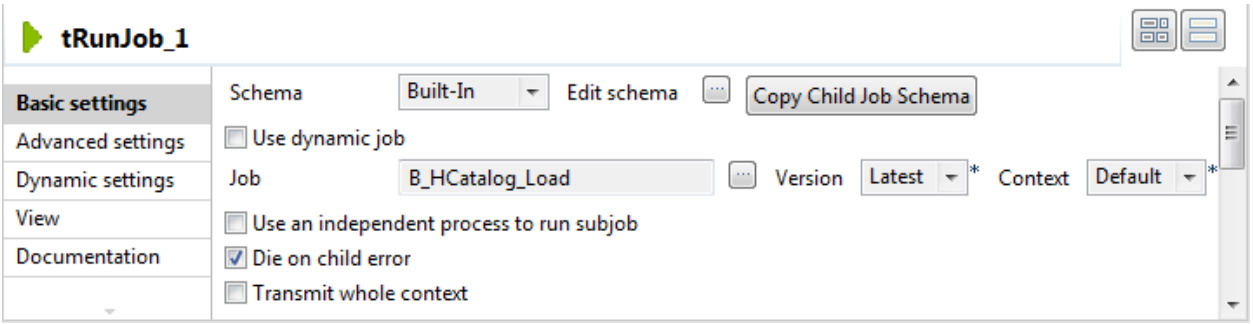
Procédure

1. Déposez un **tRunJob** dans l'espace de modélisation graphique du premier Job, `A_HCatalog_Create` dans cet exemple. Ce composant apparaît comme un sous-job.
2. Reliez le sous-job précédent au **tRunJob** à l'aide d'un lien **Trigger** > **On Subjob Ok**.



3. Double-cliquez sur le **tRunJob** afin d'ouvrir sa vue **Basic settings**.
4. Cliquez sur le bouton **[...]** à côté du champ **Job** afin d'ouvrir la boîte de dialogue **[Repository Content]**. Sélectionnez le Job qui doit être déclenché une fois le Job actuel exécuté avec succès et cliquez sur **OK** afin de fermer la boîte de dialogue.

   Le Job suivant apparaît dans le champ **Job**.



5. Double-cliquez à nouveau sur me **tRunJob** afin d'ouvrir le Job suivant. Répétez les étapes précédentes jusqu'à ce qu'un **tRunJob** soit configuré dans le Job `E_Pig_Count_IPs` afin de déclencher le dernier Job, `F_Read_Results`.
6. Exécutez le premier Job.

   L'exécution avec succès de chaque Job déclenche le Job suivant, jusqu'à ce que tous les Jobs soient exécutés. Les résultats de l'exécution sont affichés dans la console du premier Job.