

Getting Started with CDP Private Cloud Base Upgrade and Migration

Date published: 2019-11-22

Date modified: 2024-04-05



Legal Notice

© Cloudera Inc. 2024. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 (“ASLv2”), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

Contents

CDP Upgrade and Migrations Paths.....	4
Downloadable CDP upgrade checklists.....	6
Supported in-place upgrade paths.....	7
Cloudera Manager support for CDH, Cloudera Runtime and CDP Private Cloud Data Services.....	11
Assessing the Impact of Apache Hive.....	16
Apache Hive features Apache Hive features in Cloudera Data Warehouse.....	16
Apache Hive 3 architectural overview Apache Hive 3 in Cloudera Data Warehouse architectural overview.....	17
Key semantic changes and workarounds.....	19
Casting timestamps.....	19
Casting invalid dates.....	19
Changing incompatible column types.....	20
Understanding CREATE TABLE behavior.....	20
Configuring legacy CREATE TABLE behavior.....	21
Handling table reference syntax.....	22
Add Backticks to Table References.....	23
Handling the Keyword APPLICATION.....	23
Dropping partitions.....	23
Handling output of greatest and least functions.....	24
Renaming tables.....	24
TRUNCATE TABLE on an external table.....	24
Hive unsupported interfaces and features.....	25
Supplemental Upgrade Topics.....	26
Configuring a Local Package Repository.....	26
Creating a Permanent Internal Repository.....	27
Creating a Temporary Internal Repository.....	28
Configuring Hosts to Use the Internal Repository.....	29
Configuring a Local Parcel Repository.....	29
Using an Internally Hosted Remote Parcel Repository.....	29
Using a Local Parcel Repository.....	32
Changes to CDH Hive Tables.....	32
Changes to HDP Hive tables.....	33
Transitioning from the Cloudera Manager Embedded PostgreSQL Database Server to an External PostgreSQL Database.....	34
Prerequisites.....	35
Identify Roles that Use the Embedded Database Server.....	35
Migrate Databases from the Embedded Database Server to the External PostgreSQL Database Server.....	37

CDP Upgrade and Migrations Paths

Take a look at the overview, features, and advantages of CDP and know the upgrade and migration paths from CDH or HDP platform to CDP.

Introduction to CDP

The merger of Cloudera and Hortonworks led to the new Cloudera Data Platform or CDP, which is the combined best of breed Big Data components from both Cloudera and Hortonworks.

Review the following information before you upgrade or migrate to Cloudera Data Platform (CDP):

- CDP Overview
- CDP Private Cloud Base new features

Troubleshooting

A selection of Cloudera Knowledge Base articles are available that describe common issues encountered by Cloudera customers during upgrades and migrations. See [CDP Upgrade/Migrate Troubleshooting Articles](#). (Cloudera login required.)

CDP upgrade and migration paths are:

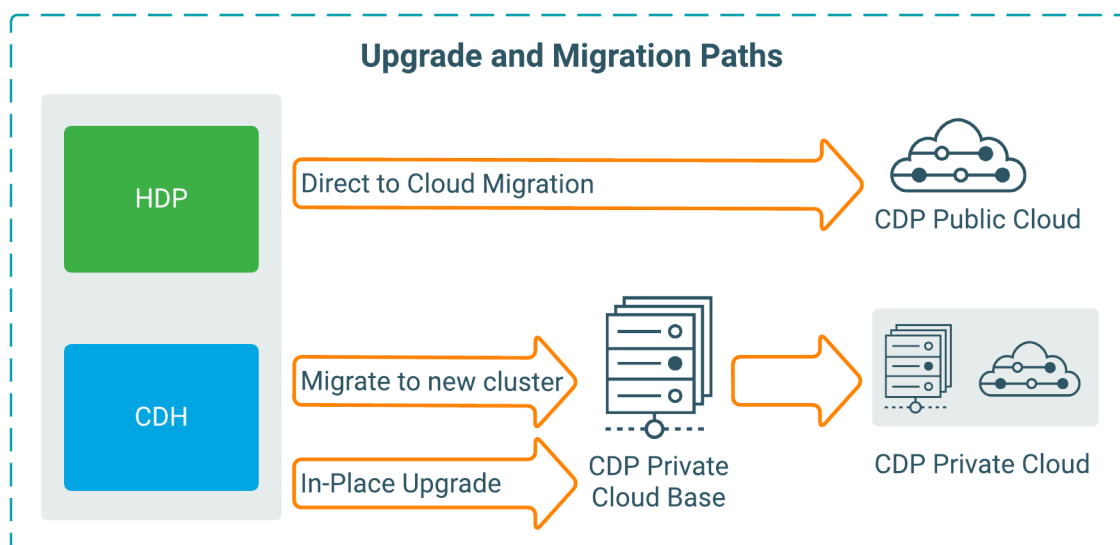
- [CDP Upgrade and Migration paths for Data-at-Rest](#)
- [CDP Upgrade and Migration paths to CDP for CDF components](#)
- [CDP Upgrade and Migration paths for CDSW \(Machine Learning\)](#)



Note: If you are upgrading to Cloudera Manager 7.5.1 or higher in order to install CDP Private Cloud Experiences version 1.3.1, you must use Cloudera Runtime version 7.1.6 or 7.1.7. For more information, see [CDP Private Cloud Experiences](#).

CDP Upgrade and Migration Paths for Data-at-Rest

If you are an HDP or a CDH user, you can follow one of the several upgrade or migration paths to CDP.



Migration and Upgrade Paths	Description
In-place upgrade	<p>Recommended for large clusters. Other paths are not viable. Involves downtime. See the following to know more about the In-place upgrade paths:</p> <ul style="list-style-type: none"> In-place upgrade from CDH to CDP Private Cloud Base In-place upgrade from HDP to CDP Private Cloud Base
Migrate to new cluster	<p>Recommended if you are ready for a hardware refresh or have small clusters. You can fall back to your original cluster in the event of upgrade issues. Requires additional hardware.</p>
Direct to cloud migration	<p>Recommended if you can tolerate some cluster downtime or have bursty workloads. See the following to know more about the direct to cloud migration paths:</p> <ul style="list-style-type: none"> Nifi workloads <p>If you are running NiFi workloads without Hive, Impala, HBase, or Kafka: Migrate to CDP Data Hub and use the Flow Management cluster template.</p> Kafka workloads <p>If you are running Kafka workloads without Hive, Impala, HBase, or NiFi: Migrate to CDP Data Hub and use the Streams Messaging cluster template.</p> HBase workloads <p>If you are running HBase workloads without Hive or Impala: Migrate to CDP Data Hub and use the Operational Database cluster template.</p> Hive or Impala workloads <p>If you are running Hive or Impala workloads without HBase: Migrate to Cloudera Data Warehouse.</p> Other workloads <p>Migrate to CDP Data Hub and use the custom cluster template.</p>

CDP Upgrade and Migration Paths to CDP for CDF components

You can upgrade or migrate CDF components to CDP in the following ways:

What do you want to migrate?	Do you want to migrate or upgrade?
Streaming workloads	<ul style="list-style-type: none"> In-place upgrade from CDH In-place upgrade from HDP Migrating Streaming workloads from HDF to CDP
Flow Management workloads	<ul style="list-style-type: none"> Follow the applicable upgrade or migration path in Upgrade and migration paths.
Workloads for deprecated components	<ul style="list-style-type: none"> Migrate Flume workloads to NiFi for CDP. This workflow is in development.

What do you want to migrate?	Do you want to migrate or upgrade?
	<ul style="list-style-type: none"> Migrate Storm workloads to Flink for CDP. This content is in development.

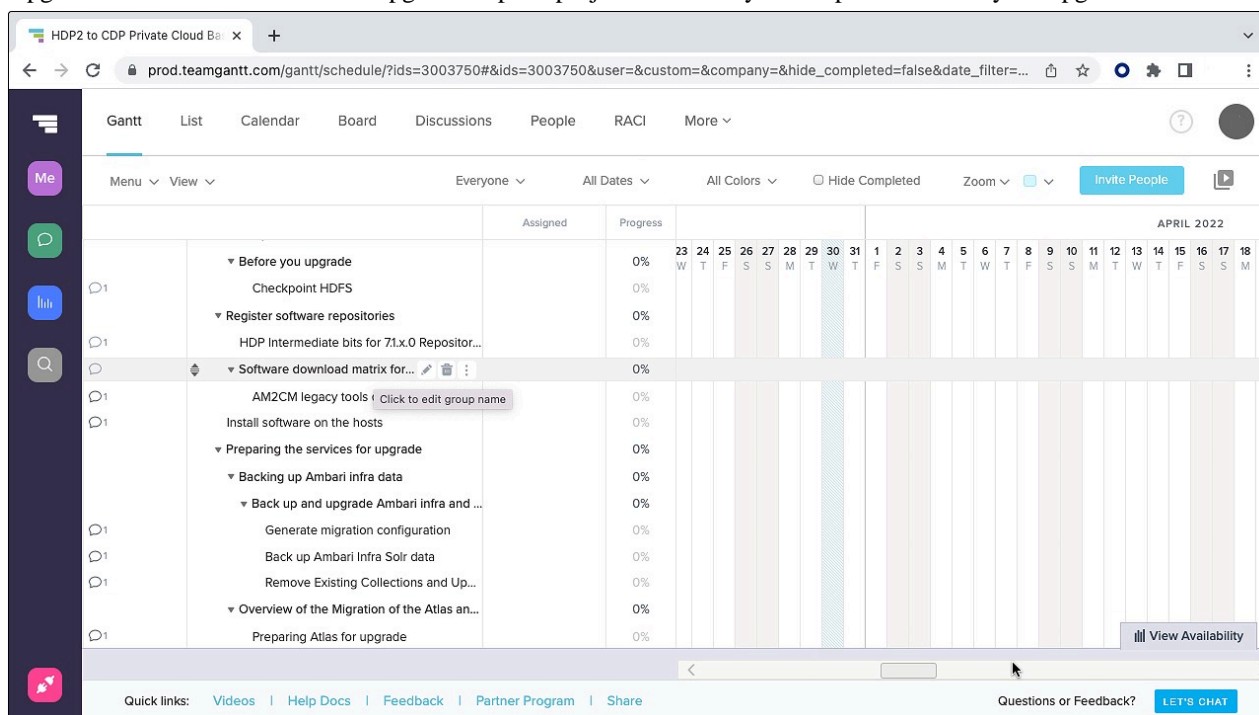
CDP Migration and Upgrade Paths for CDSW (Machine Learning)

You can upgrade or migrate CDSW to CDP Machine Learning in the following ways:

What do you want to migrate?	Do you want to migrate or upgrade?
CDSW with CDH/HDP to CDSW with CDP Private Base	For in-place upgrade follow the upgrade documentation . For migration follow the documented migration steps.
CDSW to CML-Public Cloud	The workflow and tools to migrate from CDSW to CML in CDP Public Cloud are in development. To perform this migration now, contact your Cloud account or professional services representative.
CDSW to CML-Private Cloud	The workflow and tools to migrate from CDSW to CML in CDP Private Cloud is available as technical preview. For more information, see the migration documentation .

Downloadable CDP upgrade checklists

Upgrade checklists include all the upgrade steps in project format so you can plan and track your upgrade activities.



Use the following files to assist your CDP upgrade project in any way helpful. The file structure has rows reflecting topics from CDP upgrade documentation and columns labeling the topics, the topic hierarchy, and URLs back to this website. The columns particularly target TeamGantt.com. Upload one of the files to that site to see a graphical view of your upgrade project with expandable/collapsible hierarchical tasks, the ability to assign owners, check off completed items, insert dates, click to view Cloudera documentation, etc. Using TeamGantt.com, however, is not required.

- [From CDH 5 to CDP Private Cloud Base](#)
- [From CDH 6 to CDP Private Cloud Base](#)
- [From HDP 2 to CDP Private Cloud Base](#)
- [From HDP 3 to CDP Private Cloud Base](#)
- [From CDP Private Cloud Base to CDP Private Cloud Base](#)

Supported in-place upgrade paths

Supported upgrade paths for CDP Private Cloud Base, CDP Private Cloud Data Services, CDH, and HDP.

The tables below detail the supported upgrade paths for upgrades to CDP Private Cloud Base and CDP Private Cloud Data Services. Before upgrading, ensure that you select a compatible version of Cloudera Manager. See [Cloudera Manager support for CDH](#), [Cloudera Runtime and CDP Private Cloud Data Services](#) on page 11.

For supported upgrades to CDH, see [Supported Upgrade Paths](#)

Upgrades from Cloudera Director are not supported.

For CDP Private Cloud Base

Table 1: Upgrade paths for CDP Private Cloud Base

Upgrade supported from:	Upgrade to:	Notes
<ul style="list-style-type: none"> • CDP 7.1.7.2000 (SP2) / Cloudera Manager 7.6.7 • CDP 7.1.7.1000 (SP1) / Cloudera Manager 7.6.1 • CDP 7.1.6 / Cloudera Manager 7.3.1 	CDP 7.1.7.3000 (SP3) / Cloudera Manager 7.11.3 Latest cumulative hotfix	
<ul style="list-style-type: none"> • CDP 7.1.8 CHF2 or higher / Cloudera Manager 7.7.1 • CDP 7.1.8 CHF2 or higher / Cloudera Manager 7.7.3 • CDP 7.1.7.2000 (SP2) / Cloudera Manager 7.6.7 • CDP 7.1.7.1000 (SP1) / Cloudera Manager 7.6.1 	CDP 7.1.9 / Cloudera Manager 7.11.3	
<ul style="list-style-type: none"> • CDP 7.1.7.1000 (SP1) / Cloudera Manager 7.6.1 • CDP 7.1.7 (Log4J2) / Cloudera Manager 7.4.4 (Log4J) 	CDP 7.1.7.2000 (SP2) / Cloudera Manager 7.6.7	
<ul style="list-style-type: none"> • CDP 7.1.8 / Cloudera Manager 7.7.1 • CDP 7.1.7.1000 (SP1) / Cloudera Manager 7.6.1 • CDP 7.1.7.78 (HOTFIX-4836 “Log4j”) • CDP 7.1.7 / Cloudera Manager 7.4.4 	<ul style="list-style-type: none"> • CDP 7.1.8 / Cloudera Manager 7.7.3 • Cloudera Manager 7.7.3-CHF1 	
<ul style="list-style-type: none"> • CDP 7.1.7 / Cloudera Manager 7.4.4 	CDP 7.1.7.1000 (SP1) / Cloudera Manager 7.6.1	
<ul style="list-style-type: none"> • CDP 7.1.7.1000 (SP1) / Cloudera Manager 7.6.1 	Latest cumulative hotfix	

Upgrade supported from:	Upgrade to:	Notes
<ul style="list-style-type: none"> CDP 7.1.7.78 / Cloudera Manager 7.5.4-20668437 (w/ Log4J fixes) CDP 7.1.7.1000 / Cloudera Manager 7.6.1 (SP1) CDP 7.1.7 / Cloudera Manager 7.4.4 CDP 7.1.6 / Cloudera Manager 7.3.1 CDP / Cloudera Manager 7.1.1, 7.1.2, 7.1.3, 7.1.4 CDP 7.1.5 / Cloudera Manager 7.2.4 	CDP 7.1.8 / Cloudera Manager 7.7.1	Upgrading Cloudera Manager to version 7.7.1 or higher from clusters where CDH 5.x is deployed is not supported. To upgrade such clusters: <ol style="list-style-type: none"> 1. Upgrade Cloudera Manager to version 6.3.4. 2. Upgrade CDH to version 6.3.4 3. Upgrade Cloudera Manager to version 7.7.1
<ul style="list-style-type: none"> CDP 7.0.3 - 7.1.6 Cloudera Manager 7.0.3-7.1.4, 7.2.4, 7.3.1 	CDP 7.1.7 / Cloudera Manager 7.4.4	
<ul style="list-style-type: none"> Lower versions of CDP 7.0.3 - 7.1.6 Lower versions of Cloudera Manager 7.0.3-7.1.4, 7.3.1 	CDP 7.1.1 - 7.1.6 / Cloudera Manager 7.1.1-7.1.4, 7.3.1	



Note: You can upgrade to Cloudera Runtime 7.1.7 Service Pack 1 (7.1.7.1000) from all of Cloudera Runtime 7.1.x, CDH5, and CDH6 versions. An upgrade to Cloudera Manager 7.6.1 is required for Service Pack 1 (7.1.7.1000).

Table 2: Upgrade paths for CDP Service Packs

Upgrade supported from:	Upgrade to:	Notes
<ul style="list-style-type: none"> CDP 7.1.7.1000 (SP1) / Cloudera Manager 7.6.1 CDP 7.1.7 (Log4J2) / Cloudera Manager 7.4.4 (Log4J) CDP 7.1.6 / Cloudera Manager 7.3.1 	CDP 7.1.7.2000 (SP2) / Cloudera Manager 7.6.7	
CDP 7.1.7/Cloudera Manager 7.4.4	CDP 7.1.7.1000 (SP1) / Cloudera Manager 7.6.1	<ul style="list-style-type: none"> Parcel upgrade only required (running the Upgrade Wizard is not required). See Applying a Service Pack. Cloudera Manager 7.6.5 is intended for use with CDP Private Cloud with Data Services. Cloudera does not recommend using Cloudera Manager 7.6.5 in CDP Private Cloud Base environments without Data Services installations.

For CDP Private Cloud Data Services

Table 3: Upgrade paths for CDP Private Cloud Data Services 1.5.2

Source version			Target version	
Cloudera Manager	CDP Private Cloud Base	CDP Private Cloud Data Services	Cloudera Manager	CDP Private Cloud Base
Greenfield deployment of CDP Private Cloud Data Services 1.5.2				
NA	NA	NA	7.11.3 Cumulative hotfix 1	7.1.9
NA	NA	NA	7.11.3 Cumulative hotfix 1	7.1.9 Cumulative hotfix 1
NA	NA	NA	7.11.3 Cumulative hotfix 1	7.1.8 CHF11 or higher
NA	NA	NA	7.11.3 Cumulative hotfix 1	7.1.7 SP2
Using CDP Private Cloud Base, but new to Private Cloud Data Services 1.5.2				
7.11.3	7.1.9	NA	7.11.3 Cumulative hotfix 1	7.1.9


Source version			Target version		
Cloudera Manager	CDP Private Cloud Base	CDP Private Cloud Data Services	Cloudera Manager	CDP Private Cloud Base	
7.11.3	7.1.9 Cumulative hotfix 1	NA	7.11.3 Cumulative hotfix 1	7.1.9 Cumulative hotfix 1	
7.7.3 Latest cumulative hotfix	7.1.8 Cumulative hotfix 11 or higher	NA	7.11.3 Cumulative hotfix 1	7.1.8 Cumulative hotfix 11 or higher	
7.7.1 Latest cumulative hotfix	7.1.8 Cumulative hotfix 11 or higher	NA	7.11.3 Cumulative hotfix 1	7.1.8 Cumulative hotfix 11 or higher	
7.6.7 Latest cumulative hotfix	7.1.7 SP2	NA	7.11.3 Cumulative hotfix 1	7.1.7 SP2	
Using CDP Private Cloud Data Services and wanting to upgrade to the latest version without upgrading Base version					
7.10.1	7.1.7 SP2	1.5.1	7.11.3 Cumulative hotfix 1	7.1.7 SP2	
7.10.1	7.1.8 Cumulative hotfix 11 or higher	1.5.1	7.11.3 Cumulative hotfix 1	7.1.8 Cumulative hotfix 11 or higher	
7.9.5	7.1.8 Cumulative hotfix 11 or higher	1.5.0	7.11.3 Cumulative hotfix 1	7.1.8 Cumulative hotfix 11 or higher	
7.9.5	7.1.8	1.5.0	7.11.3 Cumulative hotfix 1	7.1.8	
7.9.5	7.1.7 SP2	1.5.0	7.11.3 Cumulative hotfix 1	7.1.7 SP2	
 Important: Upgrading from Cloudera Manager 7.7.3 version to Cloudera Manager 7.10.1 is currently not supported. Note that, Cloudera Manager 7.7.3 version is supported on Python 3 and Cloudera Manager 7.10.1 supports only Python 2. Private Cloud Data Services 1.5.1 support for users using Cloudera Manager 7.7.3 is intended to be made available in the future with a new version of Cloudera Manager that has support for both Python 3 and 2 versions respectively.					

Table 4: Upgrade paths for CDP Private Cloud Data Services 1.5.1

Source version			Target version		
Cloudera Manager	CDP Private Cloud Base	CDP Private Cloud Data Services	Cloudera Manager	CDP Private Cloud Base	
Greenfield deployment of CDP Private Cloud Data Services 1.5.1					
NA	NA	NA	7.10.1	7.1.7 SP2	
NA	NA	NA	7.10.1	7.1.8 CHF4	
Using CDP Private Cloud Base, but new to Private Cloud Data Services 1.5.1					
7.6.7	7.1.7 SP2	NA	7.10.1	7.1.8 CHF4 ¹	
7.7.1	7.1.8	NA	7.10.1	7.1.8 CHF4 ²	
7.6.7	7.1.7 SP2	NA	7.10.1	7.1.8	
7.7.1	7.1.8	NA	7.10.1	7.1.8	
Using CDP Private Cloud Data Services and wanting to upgrade to the latest version without upgrading Base version					
7.8.1	7.1.7 SP1	1.4.1	7.10.1	7.1.7 SP1	
7.8.1	7.1.8	1.4.1	7.10.1	7.1.8	
7.9.5	7.1.7 SP2	1.5.0	7.10.1	7.1.7 SP2	
7.9.5	7.1.7 SP1	1.5.0	7.10.1	7.1.7 SP1	
7.9.5	7.1.8	1.5.0	7.10.1	7.1.8	

¹ Upgrade from 7.1.7 SP2 to 7.1.8 CHF4 is not a mandatory upgrade. In 7.1.8 CHF4, you can install Ozone as a parcel.

² Upgrade from 7.1.8 to 7.1.8 CHF4 is not a mandatory upgrade. In 7.1.8 CHF4, you can install Ozone as a parcel.



Important: Upgrading from Cloudera Manager 7.7.3 version to Cloudera Manager 7.9.5 is currently not supported. Note that, Cloudera Manager 7.7.3 version is supported on Python 3 and Cloudera Manager 7.9.5 supports only Python 2. Private Cloud Data Services 1.5.0 support for users using Cloudera Manager 7.7.3 is intended to be made available in the future with a new version of Cloudera Manager that has support for both Python 3 and 2 versions respectively.

Table 5: Upgrade paths for CDP Private Cloud Data Services 1.5.0

Source version			Target version		
Cloudera Manager	CDP Private Cloud Base	CDP Private Cloud Data Services	Cloudera Manager	CDP Private Cloud Base	
Greenfield deployment of CDP Private Cloud Data Services 1.5.0					
NA	NA	NA	7.9.5	7.1.7 SP2	
NA	NA	NA	7.9.5	7.1.7 SP1	
NA	NA	NA	7.9.5	7.1.8	
Using CDP Private Cloud Base, but new to Private Cloud Data Services 1.5.0					
7.6.7	7.1.7 SP2	NA	7.9.5	7.1.7 SP2	
7.6.1	7.1.7 SP1	NA	7.9.5	7.1.7 SP1	
7.7.1	7.1.8	NA	7.9.5	7.1.8	
Using CDP Private Cloud Data Services and wanting to upgrade to the latest version without upgrading Base version					
7.8.1	7.1.7 SP1	1.4.1	7.9.5	7.1.7 SP1	
7.8.1	7.1.8	1.4.1	7.9.5	7.1.8	
7.6.5	7.1.7 SP1	1.4.0-H1	7.9.5	7.1.7 SP1	
7.6.5	7.1.7	1.4.0-H1	7.9.5	7.1.7	

Table 6: Upgrade paths for CDP Private Cloud Data Services 1.4.1

Source version			Target version		
Cloudera Manager	CDP Private Cloud Base	CDP Private Cloud Data Services	Cloudera Manager	CDP Private Cloud Base	
Greenfield deployment of CDP Private Cloud Data Services 1.4.1					
NA	NA	NA	7.8.1	7.1.7 SP1	
NA	NA	NA	7.8.1	7.1.8	
Using CDP Private Cloud Base, but new to Private Cloud Data Services 1.4.1					
7.7.1	7.1.8	NA	7.8.1	7.1.8	
Using CDP Private Cloud Data Services and wanting to upgrade to the latest version without upgrading Base version					
7.6.5	7.1.7 SP1	1.4.0-H1	7.8.1	7.1.7 SP1	
7.6.5	7.1.7	1.4.0-H1	7.8.1	7.1.7	

For Upgrade paths for CDH and HDP

Table 7: Upgrade paths for CDH

Upgrade supported from:	Upgrade to:	Notes
<ul style="list-style-type: none"> CDH 6.3.4 / Cloudera Manager 6.3.4 CDH 6.2.1 / Cloudera Manager 6.2.1 	CDP 7.1.9 / Cloudera Manager 7.11.3	

Upgrade supported from:	Upgrade to:	Notes
<ul style="list-style-type: none"> CDH 6.3.4 / Cloudera Manager 6.3.4 CDH 6.2.1 / Cloudera Manager 6.2.1 CDH 6.1.1 / Cloudera Manager 6.1.1 CDH 5.16.2 / Cloudera Manager 5.16.2 CDH 5.16.2 / Cloudera Manager 6.3.1 	CDP 7.1.7.2000 (SP2) / Cloudera Manager 7.6.7	
<ul style="list-style-type: none"> CDH / Cloudera Manager 6.1 - 6.3 	CDP 7.1.8 / Cloudera Manager 7.7.1	Upgrading Cloudera Manager to version 7.7.1 or higher from clusters where CDH 5.x is deployed is not supported. To upgrade such clusters: <ol style="list-style-type: none"> 1. Upgrade Cloudera Manager to version 6.3.4. 2. Upgrade CDH to version 6.3.4 3. Upgrade Cloudera Manager to version 7.7.1
<ul style="list-style-type: none"> CDH / Cloudera Manager 5.13-5.16 CDH / Cloudera Manager 6.1-6.3 	CDP 7.1.7 / Cloudera Manager 7.4.4	
<ul style="list-style-type: none"> CDH / Cloudera Manager 5.13 - 5.16 CDH / Cloudera Manager 6.1, 6.2 	CDH 6.3	
CDH / Cloudera Manager 5.0 - 5.12	CDH / Cloudera Manager 5.13 - 5.16	Upgrades from Cloudera Manager/CDH 5.0 - 5.12 to CDP Private Cloud Base require that you first upgrade to Cloudera Manager/CDH 5.13 or higher.

Table 8: Upgrade paths for HDP

Upgrade supported from:	Upgrade to:	Notes
HDP 3.1.5	CDP 7.1.7 SP1, CDP 7.1.8, CDP 7.1.7 SP2, and 7.1.9.	One stage upgrade.
HDP 2.6.5	CDP 7.1.8 and CDP 7.1.7 SP2.	One stage upgrade.
HDP 3.1.5	CDP 7.1.8 / Cloudera Manager 7.7.1	Two stage upgrade from Ambari required.
HDP 2.6.5	CDP 7.1.8 / Cloudera Manager 7.7.1 . Requires an interim upgrade to CDP 7.1.7/ Cloudera Manager 7.4.4.	Two stage upgrade from Ambari required.
HDP 2.6.5	CDP 7.1.1 - 7.1.7	You can upgrade from HDP 2.6.5 and Ambari 2.6.2.x to CDP Private Cloud Base 7.1.x This upgrade requires several major steps, including upgrading to an interim version of Ambari. After the upgrade, your cluster will be managed by Cloudera Manager and the components will be upgraded to Cloudera Runtime 7.1.1 or higher.

Cloudera Manager support for CDH, Cloudera Runtime and CDP Private Cloud Data Services

Describes which versions of CDH, Cloudera Runtime and CDP Private Cloud Data Services are supported by Cloudera Manager.





Note: Not all combinations of Cloudera Manager, Cloudera Runtime, and CDP Private Cloud Data Services are supported. Ensure that the version of Cloudera Manager you are using supports the version of Cloudera Runtime and CDP Private Cloud Data Services you have selected. For more information, see the [Cloudera Support Matrix](#).

The versions of Cloudera Runtime, CDP Private Cloud Data Services, and CDH clusters that can be managed by Cloudera Manager are limited to the following:

For CDP Private Cloud Base

Table 9: Cloudera Manager support for CDP Private Cloud Base

Cloudera Manager Version	Supported CDH/Runtime versions	Supported CDP Private Cloud Data Services versions
Cloudera Manager 7.11.3 Latest cumulative hotfix	<ul style="list-style-type: none"> • Cloudera Runtime 7.1.7 SP3 • Cloudera Runtime 7.1.9 • Cloudera Runtime 7.1.7 SP2 • Cloudera Runtime 7.1.8 • Cloudera Runtime 7.1.7 SP1 • Cloudera Runtime 7.1.7 • Cloudera Runtime 7.1.6 • Cloudera Runtime 7.1.5 • Cloudera Runtime 7.1.4 • Cloudera Runtime 7.1.3 • Cloudera Runtime 7.1.2 • Cloudera Runtime 7.1.1 • Cloudera Runtime 7.0.3 • CDH 6.3 • CDH 6.2 • CDH 6.1 • CDH 6.0 	None
Cloudera Manager 7.11.3  Note: You must install Python 3.8 (or 3.9 for RHEL 9.1) on all hosts before installing or upgrading to Cloudera Manager 7.11.3. For more information, see the Installing Python 3 .	<ul style="list-style-type: none"> • Cloudera Runtime 7.1.9 • Cloudera Runtime 7.1.7 SP2 • Cloudera Runtime 7.1.8 • Cloudera Runtime 7.1.7 SP1 • Cloudera Runtime 7.1.7 • Cloudera Runtime 7.1.6 • Cloudera Runtime 7.1.5 • Cloudera Runtime 7.1.4 • Cloudera Runtime 7.1.3 • Cloudera Runtime 7.1.2 • Cloudera Runtime 7.1.1 • Cloudera Runtime 7.0.3 • CDH 6.3 • CDH 6.2 • CDH 6.1 • CDH 6.0 	None
Cloudera Manager 7.7.3 should only be used when you need to use Python 3.8 for the Cloudera Manager agents. You must install Python 3.8 on all hosts before installing or upgrading to Cloudera Manager 7.7.3. Cloudera Manager 7.7.3-CHF2 supports only RHEL 8.4, RHEL 8.6, and RHEL 7.9. See the CDP Private Cloud Base Installation Guide for more information.	<ul style="list-style-type: none"> • Cloudera Runtime 7.1.8 	None




Cloudera Manager Version	Supported CDH/Runtime versions	Supported CDP Private Cloud Data Services versions
Cloudera Manager 7.7.1  Note: Cloudera recommends you to use latest cumulative hotfix of Cloudera Manager 7.7.1 with Cloudera Runtime 7.1.7-SP2.	<ul style="list-style-type: none"> • Cloudera Runtime 7.1.7 SP2 • Cloudera Runtime 7.1.8 • Cloudera Runtime 7.1.7 SP1 • Cloudera Runtime 7.1.7 • Cloudera Runtime 7.1.6 • Cloudera Runtime 7.1.5 • Cloudera Runtime 7.1.4 • Cloudera Runtime 7.1.3 • Cloudera Runtime 7.1.2 • Cloudera Runtime 7.1.1 • Cloudera Runtime 7.0.3 • CDH 6.3 • CDH 6.2 • CDH 6.1 • CDH 6.0 	None
7.6.7  Important: Do not upgrade to Cloudera Manager 7.6.7 if you are running CDP Private Cloud Data Services in your deployment.	<ul style="list-style-type: none"> • Cloudera Runtime 7.1.7 SP2 • Cloudera Runtime 7.1.7 SP1 • Cloudera Runtime 7.1.7 • Cloudera Runtime 7.1.6 • Cloudera Runtime 7.1.5 • Cloudera Runtime 7.1.4 • Cloudera Runtime 7.1.3 • Cloudera Runtime 7.1.2 • Cloudera Runtime 7.1.1 • Cloudera Runtime 7.0.3 • CDH 6.3 • CDH 6.2 • CDH 6.1 • CDH 6.0 • CDH 5.16.2 	None
7.6.1  Important: Do not upgrade to Cloudera Manager 7.6.1 if you are running CDP Private Cloud Data Services in your deployment.	<ul style="list-style-type: none"> • Cloudera Runtime 7.1.7 SP1 • Cloudera Runtime 7.1.7 • Cloudera Runtime 7.1.6 • Cloudera Runtime 7.1.5 • Cloudera Runtime 7.1.4 • Cloudera Runtime 7.1.3 • Cloudera Runtime 7.1.2 • Cloudera Runtime 7.1.1 • Cloudera Runtime 7.0.3 • CDH 6.3 • CDH 6.2 • CDH 6.1 • CDH 6.0 • CDH 5.13 - 5.16 	None






Cloudera Manager Version	Supported CDH/Runtime versions	Supported CDP Private Cloud Data Services versions
7.4.4	<ul style="list-style-type: none"> Cloudera Runtime 7.1.7 Cloudera Runtime 7.1.6 Cloudera Runtime 7.1.5 Cloudera Runtime 7.1.4 Cloudera Runtime 7.1.3 Cloudera Runtime 7.1.2 Cloudera Runtime 7.1.1 Cloudera Runtime 7.0.3 CDH 6.3 CDH 6.2 CDH 6.1 CDH 6.0 CDH 5.13 - 5.16 	None
7.3.1	<ul style="list-style-type: none"> Cloudera Runtime 7.1.6 Cloudera Runtime 7.1.5 Cloudera Runtime 7.1.4 Cloudera Runtime 7.1.3 Cloudera Runtime 7.1.2 Cloudera Runtime 7.1.1 Cloudera Runtime 7.0.3 CDH 6.3 CDH 6.2 CDH 6.1 CDH 6.0 CDH 5.13 - 5.16 	None
7.2.4	<ul style="list-style-type: none"> Cloudera Runtime 7.1.5 Cloudera Runtime 7.1.4 Cloudera Runtime 7.1.3 Cloudera Runtime 7.1.2 Cloudera Runtime 7.1.1 Cloudera Runtime 7.0.3 CDH 6.3 CDH 6.2 CDH 6.1 CDH 6.0 CDH 5.13 - 5.16 	1.2 Supported with Cloudera Runtime 7.1.5 only
7.1.4	<ul style="list-style-type: none"> Cloudera Runtime 7.1.4 Cloudera Runtime 7.1.3 Cloudera Runtime 7.1.2 Cloudera Runtime 7.1.1 Cloudera Runtime 7.0.3 CDH 6.3 CDH 6.2 CDH 6.1 CDH 6.0 CDH 5.13 - 5.16 	None

Cloudera Manager Version	Supported CDH/Runtime versions	Supported CDP Private Cloud Data Services versions
7.1.3	<ul style="list-style-type: none"> Cloudera Runtime 7.1.3 Cloudera Runtime 7.1.2 Cloudera Runtime 7.1.1 Cloudera Runtime 7.0.3 CDH 6.3 CDH 6.2 CDH 6.1 CDH 6.0 CDH 5.13 - 5.16 	1.1
7.1.2	<ul style="list-style-type: none"> Cloudera Runtime 7.1.2 Cloudera Runtime 7.1.1 Cloudera Runtime 7.0.3 CDH 6.3 CDH 6.2 CDH 6.1 CDH 6.0 CDH 5.13 - 5.16 	1.0
7.1.1	<ul style="list-style-type: none"> Cloudera Runtime 7.1.1 Cloudera Runtime 7.0.3 CDH 6.3 CDH 6.2 CDH 6.1 CDH 6.0 CDH 5.13 - 5.16 	None
7.0.3	<ul style="list-style-type: none"> Cloudera Runtime 7.0.3 	None

For CDP Private Cloud Data Services

Table 10: Cloudera Manager support for CDP Private Cloud Data Services

Cloudera Manager Version	Supported CDH/Runtime versions	Supported CDP Private Cloud Data Services versions
 Important: Upgrade to Cloudera Manager 7.11.3 cumulative hotfix 1 only if you are running CDP Private Cloud Data Services in your deployment.	<ul style="list-style-type: none"> Supported with Cloudera Runtime 7.1.7 SP2, 7.1.8 CHF11 or higher, 7.1.9, and 7.1.9 CHF1 only when CDP Private Cloud Data Services is deployed. 	1.5.2 Only supported with Cloudera Runtime 7.1.7 SP2, 7.1.8 CHF11 or higher, 7.1.9, and 7.1.9 CHF1.
 Important: Upgrade to Cloudera Manager 7.10.1 only if you are running CDP Private Cloud Data Services in your deployment.	<ul style="list-style-type: none"> Supported with Cloudera Runtime 7.1.7 SP2, 7.1.7 SP1, and 7.1.8 CHF4 only when CDP Private Cloud Data Services is deployed. 	1.5.1 Only supported with Cloudera Runtime 7.1.7 SP2, 7.1.7 SP1, and 7.1.8 CHF4.
 Important: Upgrade to Cloudera Manager 7.9.5 only if you are running CDP Private Cloud Data Services in your deployment.	<ul style="list-style-type: none"> Supported with Cloudera Runtime 7.1.7 SP2, 7.1.7 SP1, and 7.1.8 only when CDP Private Cloud Data Services is deployed. 	1.5.0 Only supported with Cloudera Runtime 7.1.7 SP2, 7.1.7 SP1 and 7.1.8.

Cloudera Manager Version	Supported CDH/Runtime versions	Supported CDP Private Cloud Data Services versions
7.8.1  Important: Upgrade to Cloudera Manager 7.8.1 only if you are running CDP Private Cloud Data Services in your deployment.	<ul style="list-style-type: none"> Supported with Cloudera Runtime 7.1.7 SP1, and 7.1.8 only when CDP Private Cloud Data Services is deployed. 	1.4.1 Only supported with Cloudera Runtime 7.1.7 SP1 and 7.1.8.
7.6.5  Important: Upgrade to Cloudera Manager 7.6.5 only if you are running CDP Private Cloud Data Services in your deployment.	<ul style="list-style-type: none"> Supported with Cloudera Runtime 7.1.6, 7.1.7, and 7.1.7 SP1 only when CDP Private Cloud Data Services is deployed. 	1.3.1, 1.3.2, 1.3.3. 1.3.4 are supported with Cloudera Runtime 7.1.6, 7.1.7. 1.4.0 is supported with Cloudera Runtime 7.1.7 SP1 only.
7.5.5	<ul style="list-style-type: none">  Note: Cloudera Manager 7.5.5 is not compatible with the Spark 3 CDS parcel. Cloudera Runtime 7.1.7 Cloudera Runtime 7.1.6 	1.3.1, 1.3.2, 1.3.3. 1.3.4 Supported with Cloudera Runtime 7.1.6 and 7.1.7 only
7.5.4	<ul style="list-style-type: none">  Note: Cloudera Manager 7.5.4 is not compatible with the Spark 3 CDS parcel. Cloudera Runtime 7.1.7 Cloudera Runtime 7.1.6 	1.3.1, 1.3.2, 1.3.3 Supported with Cloudera Runtime 7.1.6 and 7.1.7 only
7.5.1	<ul style="list-style-type: none">  Note: Cloudera Manager 7.5.1 is not compatible with the Spark 3 CDS parcel. Cloudera Runtime 7.1.7 Cloudera Runtime 7.1.6 	1.3.1 Supported with Cloudera Runtime 7.1.6 and 7.1.7 only

Assessing the Impact of Apache Hive

Apache Hive features Apache Hive features in Cloudera Data Warehouse

Major changes to Apache Hive 2.x improve Apache Hive 3.x transactions and security. Knowing the major differences between these versions is critical for SQL users, including those who use Apache Spark and Apache Impala.

Hive is a data warehouse system for summarizing, querying, and analyzing huge, disparate data sets.

ACID transaction processing

Hive 3 tables are ACID (Atomicity, Consistency, Isolation, and Durability)-compliant. Hive 3 write and read operations improve the performance of transactional tables. Atomic operations include simple writes and inserts, writes to multiple partitions, and multiple inserts in a single SELECT statement. A read operation is not affected by changes that occur during the operation. You can insert or delete data, and it remains consistent throughout software and hardware crashes. Creation and maintenance of Hive tables is simplified because there is no longer any need to bucket tables.

Materialized views

Because multiple queries frequently need the same intermediate roll up or joined table, you can avoid costly, repetitious query portion sharing, by precomputing and caching intermediate tables into views.

Query results cache

Hive filters and caches similar or identical queries. Hive does not recompute the data that has not changed. Caching repetitive queries can reduce the load substantially when hundreds or thousands of users of BI tools and web services query Hive.

Scheduled Queries

Using SQL statements, you can schedule Hive queries to run on a recurring basis, monitor query progress, temporarily ignore a query schedule, and limit the number running in parallel. You can use scheduled queries to start compaction and periodically rebuild materialized views, for example.

Security improvements

Apache Ranger secures Hive data by default. To meet demands for concurrency improvements, ACID support, render security, and other features, Hive tightly controls the location of the warehouse on a file system, or object store, and memory resources.

With Apache Ranger and Apache Hive ACID support, your organization will be ready to support and implement GDPR (General Data Protection Regulation).

Connection Pooling

Hive supports HikariCP JDBC connection pooling.

Unsupported features

CDP does not support the following features that were available in HDP and CDH platforms:

- **CREATE TABLE** that specifies a managed table location
Do not use the **LOCATION** clause to create a managed table. Hive assigns a default location in the warehouse to managed tables.
- **CREATE INDEX**
Hive builds and stores indexes in ORC or Parquet within the main table, instead of a different table, automatically. Set `hive.optimize.index.filter` to enable use (not recommended--use materialized views instead). Existing indexes are preserved and migrated in Parquet or ORC to CDP during upgrade.

Related Information

[Blog: Enabling high-speed Spark direct reader for Apache Hive ACID tables](#)

Apache Hive 3 architectural overview Apache Hive 3 in Cloudera Data Warehouse architectural overview

Understanding Apache Hive 3 major design features, such as default ACID transaction processing, can help you use Hive to address the growing needs of enterprise data warehouse systems.

Data storage and access control

One of the major architectural changes to support Hive 3 design gives Hive much more control over metadata memory resources and the file system, or object store. The following architectural changes from Hive 2 to Hive 3 provide improved security:

- Tightly controlled file system and computer memory resources, replacing flexible boundaries: Definitive boundaries increase predictability. Greater file system control improves security.
- Optimized workloads in shared files and containers

Hive 3 is optimized for object stores in the following ways:

- Hive uses ACID to determine which files to read rather than relying on the storage system.
- In Hive 3, file movement is reduced from that in Hive 2.
- Hive caches metadata and data aggressively to reduce file system operations

The major authorization model for Hive is Ranger. Hive enforces access controls specified in Ranger. This model offers stronger security than other security schemes and more flexibility in managing policies.

This model permits only Hive to access the Hive warehouse.

Transaction processing

You can deploy new Hive application types by taking advantage of the following transaction processing characteristics:

- Mature versions of ACID transaction processing:

ACID tables are the default table type.

ACID enabled by default causes no performance or operational overload.

- Simplified application development, operations with strong transactional guarantees, and simple semantics for SQL commands

You do not need to bucket ACID tables.

- Materialized view rewrites
- Automatic query cache
- Advanced optimizations

Hive client changes

You can use the thin client Beeline for querying Hive from a client. You can run Hive administrative commands from the client. Beeline uses a JDBC connection to Hive to run commands. Hive parses, compiles, and runs operations. Beeline supports many of the command-line options that Hive CLI supported. Beeline does not support `hive -e set key=value` to configure the Hive Metastore.

You enter supported Hive CLI commands by invoking Beeline using the `hive` keyword, command option, and command. For example, `hive -e set`. Using Beeline instead of the thick client Hive CLI, which is no longer supported, has several advantages, including low overhead. Beeline does not use the entire Hive code base. A small number of daemons required to run queries simplifies monitoring and debugging.

Hive enforces allowlist and denylist settings that you can change using SET commands. Using the denylist, you can restrict memory configuration changes to prevent instability. Different Hive instances with different allowlists and denylists to establish different levels of stability.

Apache Hive Metastore sharing

Hive, Impala, and other components can share a remote Hive metastore.

Query execution of batch and interactive workloads

You can connect to Hive using a JDBC command-line tool, such as Beeline, or using an JDBC/ODBC driver with a BI tool, such as Tableau. You configure the settings file for each instance to perform either batch or interactive processing.

Related Information

[Blog: Enabling high-speed Spark direct reader for Apache Hive ACID tables](#)

Key semantic changes and workarounds

As SQL Developer, Analyst, or other Hive user, you need to know potential problems with queries due to semantic changes. Some of the operations that changed were not widely used, so you might not encounter any of the problems associated with the changes.

Over the years, Apache Hive committers enhanced versions of Hive supported in legacy releases of CDH and HDP, with users in mind. Changes were designed to maintain compatibility with Hive applications. Consequently, few syntax changes occurred over the years. A number of semantic changes, described in this section did occur, however. Workarounds are described for these semantic changes.

Casting timestamps

Results of applications that cast numerics to timestamps differ from Hive 2 to Hive 3. Apache Hive changed the behavior of CAST to comply with the SQL Standard, which does not associate a time zone with the TIMESTAMP type.

Before Upgrade to CDP

Casting a numeric type value into a timestamp could be used to produce a result that reflected the time zone of the cluster. For example, 1597217764557 is 2020-08-12 00:36:04 PDT. Running the following query casts the numeric to a timestamp in PDT:

```
> SELECT CAST(1597217764557 AS TIMESTAMP);
| 2020-08-12 00:36:04 |
```

After Upgrade to CDP

Casting a numeric type value into a timestamp produces a result that reflects the UTC instead of the time zone of the cluster. Running the following query casts the numeric to a timestamp in UTC.

```
> SELECT CAST(1597217764557 AS TIMESTAMP);
| 2020-08-12 07:36:04.557 |
```

Action Required

Change applications. Do not cast from a numeral to obtain a local time zone. Built-in functions `from_utc_timestamp` and `to_utc_timestamp` can be used to mimic behavior before the upgrade.

Related Information

[Apache Hive web site summary of timestamp semantics](#)

Casting invalid dates

Casting of an invalid date differs from Hive 1 in CDH 5 to Hive 3 in CDP. Hive 3 uses a different parser/formatter from the one used in Hive 1, which affects semantics. Hive 1 considers 00 invalid for date fields. Hive 3 considers 00 valid for date fields. Neither Hive 1 nor Hive 3 correctly handles invalid dates, and Hive-25056 addresses this issue.

Before Upgrade to CDP

Casting of invalid date (zero value in one or more of the 3 fields of date, month, year) returns a NULL value:

```
SELECT CAST ('0000-00-00' as date) , CAST ('000-00-00 00:00:00' AS TIMESTAMP) ;
```

After Upgrade to CDP

Casting of an invalid date returns a result.

```
> SELECT CAST ('0000-00-00' as date) , CAST ('000-00-00 00:00:00' AS TIMESTAMP) ;
...

```

```
00002-11-30 00:00:00.0
```

Action Required

Do not cast invalid dates in Hive 3.

Changing incompatible column types

A default configuration change can cause applications that change column types to fail.

Before Upgrade to CDP

In HDP 2.x and CDH 5.x and CDH 6 `hive.metastore.disallow.incompatible.col.type.changes` is false by default to allow changes to incompatible column types. For example, you can change a `STRING` column to a column of an incompatible type, such as `MAP<STRING, STRING>`. No error occurs.

After Upgrade to CDP

In CDP, `hive.metastore.disallow.incompatible.col.type.changes` is true by default. Hive prevents changes to incompatible column types. Compatible column type changes, such as `INT`, `STRING`, `BIGINT`, are not blocked.

Action Required

Change applications to disallow incompatible column type changes to prevent possible data corruption. Check `ALTER TABLE` statements and change those that would fail due to incompatible column types.

Related Information

[HIVE-12320](#)

Understanding CREATE TABLE behavior

Hive table creation has changed significantly since Hive 3 to improve useability and functionality. If you are upgrading from CDH or HDP, you must understand the changes affecting legacy table creation behavior.

Hive has changed table creation in the following ways:

- Creates ACID-compliant table, which is the default in CDP
- Supports simple writes and inserts
- Writes to multiple partitions
- Inserts multiple data updates in a single `SELECT` statement
- Eliminates the need for bucketing.

If you have an ETL pipeline that creates tables in Hive, the tables will be created as ACID. Hive now tightly controls access and performs compaction periodically on the tables. Using ACID-compliant, transactional tables causes no performance or operational overload. The way you access managed Hive tables from Spark and other clients changes. In CDP, access to external tables requires you to set up security access permissions.

You must understand the behavior of the `CREATE TABLE` statement in legacy platforms like CDH or HDP and how the behavior changes after you upgrade to CDP.

Before upgrading to CDP

In CDH 5, CDH 6, and HDP 2, by default `CREATE TABLE` creates a non-ACID managed table in plain text format.

In HDP 3 and CDP 7.1.0 through 7.1.7.x, by default `CREATE TABLE` creates either a full ACID transactional table in ORC format or insert-only ACID transactional tables for all other table formats.

After upgrading to CDP

- If you are upgrading from HDP 2, CDH 5, or CDH 6 to CDP 7.1.0 through CDP 7.1.8, by default `CREATE TABLE` creates a full ACID transactional table in ORC format or insert-only ACID transactional tables for all other table formats.

- If you are upgrading from HDP 3 or CDP 7.1.0 through 7.1.7.x to CDP 7.1.8, the existing behavior persists and CREATE TABLE creates either a full ACID transactional table in ORC format or insert-only ACID transactional tables for all other table formats.

Now that you understand the behavior of the CREATE TABLE statement, you can choose to modify the default table behavior by configuring certain properties. The order of preference for configuration is as follows:

Modify the default CREATE TABLE behavior

Override default behavior when creating the table

Irrespective of the database, session, or site-level settings, you can override the default table behavior by using the MANAGED or EXTERNAL keyword in the CREATE TABLE statement.

```
CREATE [MANAGED][EXTERNAL] TABLE foo (id INT);
```

Set the default table type at a database level

You can use the database property, defaultTableType=EXTERNAL or ACID to specify the default table type to be created using the CREATE TABLE statement. You can specify this property when creating the database or at a later point using the ALTER DATABASE statement. For example:

```
CREATE DATABASE test_db WITH DBPROPERTIES ('defaultTableType'='EXTERNAL');
```

In this example, tables created under the test_db database using the CREATE TABLE statement creates external tables with the purge functionality enabled (external.table.purge = 'true').

You can also choose to configure a database to allow only external tables to be created and prevent creation of ACID tables. While creating a database, you can set the database property, EXTERNAL_TABLES_ONLY=true to ensure that only external tables are created in the database. For example:

```
CREATE DATABASE test_db WITH DBPROPERTIES ('EXTERNAL_TABLES_ONLY'='true');
```

Set the default table type at a session level

You can configure the CREATE TABLE behavior within an existing beeline session by setting hive.create.as.external.legacy to true or false. Setting the value to true results in configuring the CREATE TABLE statement to create external tables by default.

When the session ends, the default CREATE TABLE behavior also ends.

Set the default table type at a site level

You can configure the CREATE TABLE behavior at the site level by configuring the hive.create.as.insert.only and hive.create.as.acid properties in Cloudera Manager under Hive configuration. When configured at the site level, the behavior persists from session to session. For more information, see Configuring CREATE TABLE behavior.

If you are a Spark user, switching to legacy behavior is unnecessary. Calling 'create table' from SparkSQL, for example, creates an external table after upgrading to CDP as it did before the upgrade. You can connect to Hive using the Hive Warehouse Connector (HWC) to read Hive ACID tables from Spark. To write ACID tables to Hive from Spark, you use the HWC and HWC API. Spark creates an external table with the purge property when you do not use the HWC API. For more information, see Hive Warehouse Connector for accessing Spark data.

Configuring legacy CREATE TABLE behavior

After you upgrade to CDP Private Cloud Base and migrate old tables, the legacy CREATE TABLE behavior of Hive is no longer available by default and you might want to switch to the legacy behavior. Legacy behavior might solve compatibility problems with your scripts during data migration, for example, when running ETL.

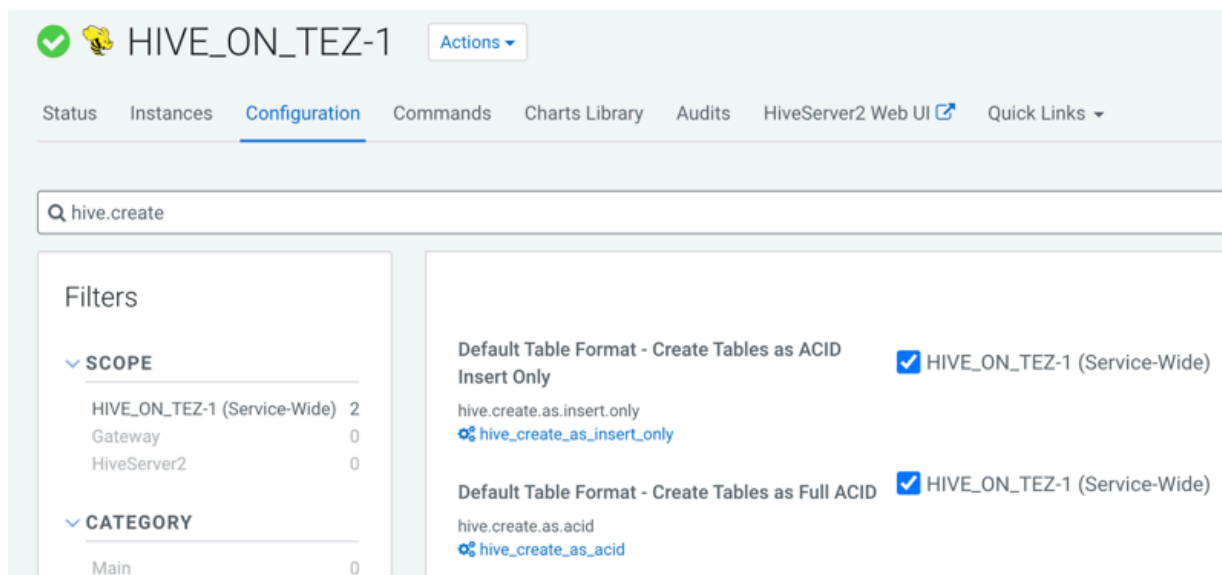
About this task

In CDP, running a CREATE TABLE statement by default creates a full ACID table for ORC file format and insert-only ACID table for other file formats. You can change the default behavior to use the legacy CREATE TABLE behavior. When you configure legacy behavior, CREATE TABLE creates external tables with the purge functionality enabled (external.table.purge = 'true'). Therefore, when the table is dropped, data is also deleted from the file system.

You can configure legacy CREATE TABLE behavior at the site level by configuring properties in Cloudera Manager. When configured at the site level, the behavior persists from session to session.

Procedure

1. In Cloudera Manager, click Clusters and select the Hive on Tez service.
2. From the Hive on Tez service, go to the Configuration tab and search for hive.create.



3. If the following properties are selected, clear the selection to enable legacy CREATE TABLE behavior.
 - Default Table Format - Create Tables as ACID Insert Only (hive.create.as.insert.only)
 - Default Table Format - Create Tables as Full ACID (hive.create.as.acid)

Results

Legacy behavior is enabled and the CREATE TABLE statement now creates external tables with the external.table.purge table property set to true.

Handling table reference syntax

For ANSI SQL compliance, Hive 3.x rejects `db.table` in SQL queries as described by the Hive-16907 bug fix. A dot (.) is not allowed in table names. As a Data Engineer, you need to ensure that Hive tables do not contain these references before migrating the tables to CDP, that scripts are changed to comply with the SQL standard references, and that users are aware of the requirement.

About this task

To change queries that use such `db.table` references thereby preventing Hive from interpreting the entire db.table string incorrectly as the table name, you enclose the database name and the table name in backticks as follows:

A dot (.) is not allowed in table names.

Procedure

1. Find a table having the problematic table reference.
For example, `math.students` appears in a `CREATE TABLE` statement.
2. Enclose the database name and the table name in backticks.

```
CREATE TABLE `math`.`students` (name VARCHAR(64), age INT, gpa DECIMAL(3,2));
```

Add Backticks to Table References

CDP includes the Hive-16907 bug fix, which rejects ``db.table`` in SQL queries. A dot (.) is not allowed in table names. You need to change queries that use such references to prevent Hive from interpreting the entire `db.table` string as the table name.

Procedure

1. Find a table having the problematic table reference.

```
math.students
```

appears in a `CREATE TABLE` statement.

2. Enclose the database name and the table name in backticks.

```
CREATE TABLE `math`.`students` (name VARCHAR(64), age INT, gpa DECIMAL(3,2));
```

Handling the Keyword APPLICATION

If you use the keyword `APPLICATION` in your queries, you might need to modify the queries to prevent failure.

To prevent a query that uses a keyword from failing, enclose the query in backticks.

Before Upgrade to CDP

In CDH releases, such as CDH 5.13, queries that use the word `APPLICATION` in queries execute successfully. For example, you could use this word as a table name.

```
> select f1, f2 from application
```

After Upgrade to CDP

A query that uses the keyword `APPLICATION` fails.

Action Required

Change applications. Enclose queries in backticks. `SELECT field1, field2 FROM `application`;`

Dropping partitions

The `OFFLINE` and `NO_DROP` keywords in the `CASCADE` clause for dropping partitions causes performance problems and is no longer supported.

Before Upgrade to CDP

You could use `OFFLINE` and `NO_DROP` keywords in the `DROP CASCADE` clause to prevent partitions from being read or dropped.

After Upgrade to CDP

`OFFLINE` and `NO_DROP` are not supported in the `DROP CASCADE` clause.

Action Required

Change applications to remove OFFLINE and NO_DROP from the DROP CASCADE clause. Use an authorization scheme, such as Ranger, to prevent partitions from being dropped or read.

Handling output of greatest and least functions

To calculate the greatest (or least) value in a column, you need to work around a problem that occurs when the column has a NULL value.

Before Upgrade to CDP

The greatest function returned the highest value of the list of values. The least function returned the lowest value of the list of values.

After Upgrade to CDP

Returns NULL when one or more arguments are NULL.

Action Required

Use NULL filters or the nvl function on the columns you use as arguments to the greatest or least functions.

```
SELECT greatest(nvl(col1,default value incase of NULL),nvl(col2,default value incase of NULL));
```

Renaming tables

To harden the system, Hive data can be stored in HDFS encryption zones. RENAME has been changed to prevent moving a table outside the same encryption zone or into a no-encryption zone.

Before Upgrade to CDP

In CDH and HDP, renaming a managed table moves its HDFS location.

After Upgrade to CDP

Renaming a managed table moves its location only if the table is created without a LOCATION clause and is under its database directory.

Action Required

None

TRUNCATE TABLE on an external table

Hive 3 does not support TRUNCATE TABLE on external tables. Truncating an external table results in an error. You can truncate an external table if you change your applications to set a table property to purge data.

Before Upgrade to CDP

Some legacy versions of Hive supported TRUNCATE TABLE on external tables.

After Upgrade to CDP Private Cloud Base

By default, TRUNCATE TABLE is supported only on managed tables. Attempting to truncate an external table results in the following error:

```
Error: org.apache.spark.sql.AnalysisException: Operation not allowed: TRUNCATE TABLE on external tables
```

Action Required

Change applications. Do not attempt to run TRUNCATE TABLE on an external table.

Alternatively, change applications to alter a table property to set `external.table.purge` to `true` to allow truncation of an external table:

```
ALTER TABLE mytable SET TBLPROPERTIES ( 'external.table.purge'='true' );
```

Hive unsupported interfaces and features

You need to know the interfaces available in HDP or CDH platforms that are not supported.

Unsupported Interfaces

The following interfaces are not supported in CDP Private Cloud Base:

- Druid
- Hcat CLI (however HCatalog is supported)
- Hive CLI (replaced by Beeline)
- Hive View UI feature in Ambari
- LLAP
- MapReduce execution engine (replaced by Tez)
- Pig
- S3 for storing tables (available in CDP Public Cloud only)
- Spark execution engine (replaced by Tez)
- Spark thrift server

Spark and Hive tables interoperate using the Hive Warehouse Connector.

- SQL Standard Authorization
- Storage Based Authorization
- Tez View UI feature in Ambari
- WebHCat

You can use Hue in lieu of Hive View.

Storage Based Authorization

Storage Based Authorization (SBA) is no longer supported in CDP. Ranger integration with Hive metastore provides consistency in Ranger authorization enabled in HiveServer (HS2). SBA did not provide authorization support for metadata that does not have a file/directory associated with it. Ranger-based authorization has no such limitation.

Hive-Kudu integration

CDP does not support the integration of HiveServer (HS2) with Kudu tables. You cannot run queries against Kudu tables from HS2.

Partially unsupported interfaces

Apache Hadoop Distributed Copy (DistCP) is not supported for copying Hive ACID tables.

Unsupported Features

CDP does not support the following features that were available in HDP and CDH platforms:

- CREATE TABLE that specifies a managed table location

Do not use the `LOCATION` clause to create a managed table. Hive assigns a default location in the warehouse to managed tables. That default location is configured in Hive using the `hive.metastore.warehouse.dir` configuration property, but can be overridden for the database by setting the `CREATE DATABASE MANAGEDLOCATION` parameter.

- CREATE INDEX and related index commands were removed in Hive 3, and consequently are not supported in CDP.

In CDP, you use the Hive 3 default ORC columnar file formats to achieve the performance benefits of indexing. Materialized Views with automatic query rewriting also improves performance. Indexes migrated to CDP are preserved but render any Hive tables with an undroppable index. To drop the index, google the Known Issue for CDPD-23041.

- Hive metastore (HMS) high availability (HA) load balancing in CDH

You need to set up HMS HA as described in the documentation.

- Local or Embedded Hive metastore server

CDP does not support the use of a local or embedded Hive metastore setup.

Unsupported Connector Use

CDP does not support the Sqoop exports using the Hadoop jar command (the Java API) that Teradata documents. For more information, see [Migrating data using Sqoop](#).

Supplemental Upgrade Topics

Additional topics to help with special situations.

Configuring a Local Package Repository

You can create a package repository for Cloudera Manager either by hosting an internal web repository or by manually copying the repository files to the Cloudera Manager Server host for distribution to Cloudera Manager Agent hosts.



Note: This internal web repository can also be used when your Cloudera Manager server or cluster does not have access to internet. You must download the installable separately from archive.cloudera.com and place the installable in the internal repository.

Loading Filters ... 7.7.3 7.7.1 7.6.7 7.6.1 7.5.1 7.4.4 7.3.1 7.2.4 7.1.4 7.1.3 7.1.2 7.1.1 7.0.3 5.16 5.15 5.14 5.13 7.11.3 5.16 5.15 5.14 5.13 7.1.9 7.1.7.2000 7.1.8 7.1.7.1000 7.1.7 7.1.6 7.1.5 7.1.4 7.1.3 7.1.2 7.1.1 7.1.7.3000



Important: Select a supported operating system for the versions of Cloudera Manager or CDH that you are downloading. See [CDH and Cloudera Manager Supported Operating Systems](#).



Important: Cloudera Manager 7.7.3 should only be used when Python 3.8 is required and only when using RHEL 8.4 or 8.6. Prior installation of Python 3.8 is required. See [Installing Python 3.8 for Cloudera Manager 7.7.3](#). Cloudera Manager 7.7.3-CHF1 adds support for RHEL 7.9.



Important: You must install Python 3 on all hosts before upgrading to Cloudera Manager 7.11.3. See [Installing Python 3](#).



Warning: Upgrades from Cloudera Manager 5.12 and lower to Cloudera Manager 7.1.1 or higher are not supported



Important: Upgrading Cloudera Manager to version 7.7.1 or higher from clusters where CDH 5.x is deployed is not supported. To upgrade such clusters:

1. Upgrade Cloudera Manager to version 6.3.4.
2. Upgrade CDH to version 6.3.4
3. Upgrade Cloudera Manager to version 7.6.5 or higher



Warning: For upgrades from CDH clusters with Cloudera Navigator to Cloudera Runtime 7.1.1 (or higher) clusters where Navigator is to be migrated to Apache Atlas, the cluster must have Kerberos enabled before upgrading.



Warning: Before upgrading CDH 5 clusters with Sentry to Cloudera Runtime 7.1.x clusters where Sentry privileges are to be transitioned to Apache Ranger:

- The cluster must have Kerberos enabled.
- Verify that HDFS gateway roles exist on the hosts that runs the Sentry service.



Important: If HDFS ACL sync is enabled (`hdfs_sentry_sync_enable=true`) on the CDH cluster, then you must install Ranger RMS to support the same functionality. For steps to install Ranger RMS, see [Installing Ranger RMS](#).



Note: If the cluster you are upgrading will include Atlas, Ranger, or both, the upgrade wizard deploys one infrastructure Solr service to provide a search capability of the audit logs through the Ranger Admin UI and/or to store and serve Atlas metadata. Cloudera recommends that you do not use this service for customer workloads to avoid interference with audit and timeline performance.

Creating a Permanent Internal Repository

The following sections describe how to create a permanent internal repository using Apache HTTP Server:

Setting Up a Web server

To host an internal repository, you must install or use an existing Web server on an internal host that is reachable by the Cloudera Manager host, and then download the repository files to the Web server host. The examples in this section use Apache HTTP Server as the Web server. If you already have a Web server in your organization, you can skip to [Downloading and publishing the package repository for Cloudera Manager](#) on page 28.

1. Install Apache HTTP Server:

RHEL / CentOS

```
sudo yum install httpd
```

SLES

```
sudo zypper install httpd
```

Ubuntu

```
sudo apt-get install httpd
```

2. Start Apache HTTP Server:

RHEL 7, 8

```
sudo systemctl start httpd
```

SLES 12, Ubuntu 16 or later

```
sudo systemctl start apache2
```

Downloading and publishing the package repository for Cloudera Manager

You can download cloudera-manager-cnav OS package for migrating Cloudera Navigator to Atlas when upgrading Cloudera Manager 6 with CDH 6 to Cloudera Runtime 7.1.9 version.

1. Download the package repository for the product you want to install:

Cloudera Manager 7

Do the following steps to download the files for a Cloudera Manager release:

- a. Run the following command to create a local repository directory to hold the Cloudera package repository:

```
sudo mkdir -p /var/www/html/cloudera-repos/cm7
```

- b. Run the following command to download the repository tarball for your operating system:

```
wget https://[username]:[password]@archive.cloudera.com/p/cm7/7.0.3/repo-as-tarball/cm7.0.3-redhat7.tar.gz
```

- c. Run the following command to unpack the tarball into the local repository directory:

```
tar xvfz cm7.0.3-redhat7.tar.gz -C /var/www/html/cloudera-repos/cm7 --strip-components=1
```

- d. Run the following command to modify the file permission that allows you to download the files under the local repository directory:

```
sudo chmod -R ugo+rX /var/www/html/cloudera-repos/cm7
```

2. Visit the Repository URL `http://<web_server>/cloudera-repos/` in your browser and verify the files you downloaded are present.



Important: If you do not see the list of downloaded files in your web browser, then you might have been configured not to display indexes. Verify your web browser settings.



Important: Include the `allkeys.asc` file at the top level of the package repository. The `allkeys.asc` file is included in the `repo-as-tarball` file. Ensure to include `allkeys.asc` file, if you are manually copying the package files between hosts.

The `allkeys.asc` file is used to validate the signatures of the package files during host installation. If `allkeys.asc` file is not available in the repository, then you cannot add a host in the Cloudera Manager.

Creating a Temporary Internal Repository

You can quickly create a temporary remote repository to deploy packages on a one-time basis. Cloudera recommends using the same host that runs Cloudera Manager, or a gateway host. This example uses [Python SimpleHTTPServer](#) as the Web server to host the `/var/www/html` directory, but you can use a different directory.

1. Download the repository you need following the instructions in [Downloading and publishing the package repository for Cloudera Manager](#) on page 28.
2. Determine a port that your system is not listening on. This example uses port 8900.
3. Start a Python SimpleHTTPServer in the `/var/www/html` directory:

```
cd /var/www/html
python -m SimpleHTTPServer 8900
```

```
Serving HTTP on 0.0.0.0 port 8900 ...
```

4. Visit the Repository URL `http://<web_server>:8900/cloudera-repos/` in your browser and verify the files you downloaded are present.

Configuring Hosts to Use the Internal Repository

After establishing the repository, modify the client configuration to use it:

OS	Procedure
RHEL compatible	<p>Create <code>/etc/yum.repos.d/cloudera-repo.repo</code> files on cluster hosts with the following content, where <code><web_server></code> is the hostname of the Web server:</p> <pre>[cloudera-repo] name=cloudera-repo baseurl=http://<web_server>/cm/5 enabled=1 gpgcheck=0</pre>
SLES	<p>Use the <code>zypper</code> utility to update client system repository information by issuing the following command:</p> <pre>zypper addrepo http://<web_server>/cm <alias></pre>
Ubuntu	<p>Create <code>/etc/apt/sources.list.d/cloudera-repo.list</code> files on all cluster hosts with the following content, where <code><web_server></code> is the hostname of the Web server:</p> <pre>deb http://<web_server>/cm <codename> <components></pre> <p>You can find the <code><codename></code> and <code><components></code> variables in the <code>./conf/distributions</code> file in the repository. After creating the <code>.list</code> file, run the following command:</p> <pre>sudo apt-get update</pre>

Configuring a Local Parcel Repository

You can create a parcel repository for Cloudera Runtime either by hosting an internal Web repository or by manually copying the repository files to the Cloudera Manager Server host for distribution to Cloudera Manager Agent hosts.

Loading Filters ... 7.7.3 7.7.1 7.6.7 7.6.1 7.5.1 7.4.4 7.3.1 7.2.4 7.1.4 7.1.3 7.1.2 7.1.1 7.0.3 5.16 5.15 5.14 5.13 7.11.3 5.16 5.15 5.14 5.13 7.1.9 7.1.7.2000 7.1.8 7.1.7.1000 7.1.7 7.1.6 7.1.5 7.1.4 7.1.3 7.1.2 7.1.1 7.1.7.3000

Using an Internally Hosted Remote Parcel Repository

The following sections describe how to use an internal Web server to host a parcel repository:

Setting Up a Web Server

To host an internal repository, you must install or use an existing Web server on an internal host that is reachable by the Cloudera Manager host, and then download the repository files to the Web server host. The examples on this page use Apache HTTP Server as the Web server. If you already have a Web server in your organization, you can skip to [Downloading and Publishing the Parcel Repository](#) on page 31.

1. Install Apache HTTP Server:

RHEL / CentOS


```
sudo yum install httpd
```

SLES

```
sudo zypper install httpd
```

Ubuntu

```
sudo apt-get install httpd
```

2.  **Warning:** Skipping this step could result in an error message Hash verification failed when trying to download the parcel from a local repository, especially in Cloudera Manager 6 and higher.

Edit the Apache HTTP Server configuration file (/etc/httpd/conf/httpd.conf by default) to add or edit the following line in the <IfModule mime_module> section:

```
AddType application/x-gzip .gz .tgz .parcel
```

If the <IfModule mime_module> section does not exist, you can add it in its entirety as follows:



Note: This example configuration was modified from the default configuration provided after installing Apache HTTP Server on RHEL 7.

```
<IfModule mime_module>
#
# TypesConfig points to the file containing the list of mappings from
# filename extension to MIME-type.
#
TypesConfig /etc/mime.types
#
# AddType allows you to add to or override the MIME configuration
# file specified in TypesConfig for specific file types.
#
#AddType application/x-gzip .tgz
#
# AddEncoding allows you to have certain browsers uncompress
# information on the fly. Note: Not all browsers support this.
#
#AddEncoding x-compress .Z
#AddEncoding x-gzip .gz .tgz
#
# If the AddEncoding directives above are commented-out, then you
# probably should define those extensions to indicate media types:
#
AddType application/x-compress .Z
AddType application/x-gzip .gz .tgz .parcel

#
# AddHandler allows you to map certain file extensions to "handlers":
# actions unrelated to filetype. These can be either built into the se
rver
# or added with the Action directive (see below)
#
# To use CGI scripts outside of ScriptAliased directories:
# (You will also need to add "ExecCGI" to the "Options" directive.)
#
#AddHandler cgi-script .cgi
```

```
# For type maps (negotiated resources):
#AddHandler type-map var

#
# Filters allow you to process content before it is sent to the client
.
#
# To parse .shtml files for server-side includes (SSI):
# (You will also need to add "Includes" to the "Options" directive.)
#
AddType text/html .shtml
AddOutputFilter INCLUDES .shtml
</IfModule>
```

3. Start Apache HTTP Server:

RHEL 7, 8

```
sudo systemctl start httpd
```

SLES 12, Ubuntu 16 or later

```
sudo systemctl start apache2
```

Downloading and Publishing the Parcel Repository

1. Look up the *Cloudera Runtime* version number for your deployment on the [Cloudera Runtime Download Information](#) page. You will need this version number in the next step.
2. Download manifest.json and the parcel files for the product you want to install:

To download the files for the latest Runtime 7 release, run the following commands on the Web server host:

```
sudo mkdir -p /var/www/html/cloudera-repos
sudo wget --recursive --no-parent --no-host-directories https://
[username]:[password]@archive.cloudera.com/p/cdh7/Cloudera Runtime
version/parcels/ -P /var/www/html/cloudera-repos
```

Disposition: / Status:

```
sudo wget --recursive --no-parent --no-host-directories https://
[username]:password@archive.cloudera.com/gplextras6/6.3.1/parcels/ -P /
var/www/html/cloudera-repos
```

```
sudo chmod -R ugo+rX /var/www/html/cloudera-repos/cdh7
```

Disposition: / Status:

```
sudo chmod -R ugo+rX /var/www/html/cloudera-repos/gplextras7
```

3. Visit the Repository URL `http://<Web_server>/cloudera-repos/` in your browser and verify the files you downloaded are present. If you do not see anything, your Web server may have been configured to not show indexes.

Configuring Cloudera Manager to Use an Internal Remote Parcel Repository

1. Use one of the following methods to open the parcel settings page:

- Navigation bar
 - a. Click the parcel icon in the top navigation bar or click Hosts and click the Parcels tab.
 - b. Click the Configuration button.
- Menu
 - a. Select AdministrationSettings.
 - b. Select CategoryParcels.

2. In the Remote Parcel Repository URLs list, click the addition symbol to open an additional row.
3. Enter the path to the parcel. For example: `http://<web_server>/cloudera-parcels/cdh7/7.2.16.0.0/`

4. Enter a Reason for change, and then click Save Changes to commit the changes.

Using a Local Parcel Repository

To use a local parcel repository, complete the following steps:

1. Open the Cloudera Manager Admin Console and navigate to the Parcels page.
2. Select Configuration and verify that you have a Local Parcel Repository path set. By default, the directory is `/opt/cloudera/parcel-repo`.
3. Remove any Remote Parcel Repository URLs you are not using, including ones that point to Cloudera archives.
4. Add the parcel you want to use to the local parcel repository directory that you specified. For instructions on downloading parcels, see [Downloading and Publishing the Parcel Repository](#) on page 31 above.
5. In the command line, navigate to the local parcel repository directory.
6. Create a SHA1 hash for the parcel you added and save it to a file named `parcel_name.parcel.sha`.

For example, the following command generates a SHA1 hash for the parcel `CDH-6.1.0-1.cdh6.1.0.p0.770702-el7.parcel`:

```
shasum CDH-6.1.0-1.cdh6.1.0.p0.770702-el7.parcel | awk '{ print $1 }'
> CDH-6.1.0-1.cdh6.1.0.p0.770702-el7.parcel.sha
```

7. Change the ownership of the parcel and hash files to cloudera-scm:

```
sudo chown -R cloudera-scm:cloudera-scm /opt/cloudera/parcel-repo/*
```

8. In the Cloudera Manager Admin Console, navigate to the Parcels page.
9. Click Check for New Parcels and verify that the new parcel appears.
10. Download, distribute, and activate the parcel.

Changes to CDH Hive Tables

As a Data Scientist, Architect, Analyst, or other Hive user you need to locate and use your Apache Hive 3 tables after an upgrade. You also need to understand the changes that occur during the upgrade process. The location of existing tables after a CDH to CDP upgrade does not change. Upgrading CDH to CDP Private Cloud Base converts Hive managed tables to external tables in Hive 3.

About this task

When the upgrade process converts a managed table to external, it sets the table property `external.table.purge` to true. The table is equivalent to a managed table having `purge` set to true in your old CDH cluster.

Managed tables on the HDFS in `/user/hive/warehouse` before the upgrade remain there after the conversion to external. Tables that were external before the upgrade are not relocated. You need to set HDFS policies to access external tables in Ranger, or set up HDFS ACLs.

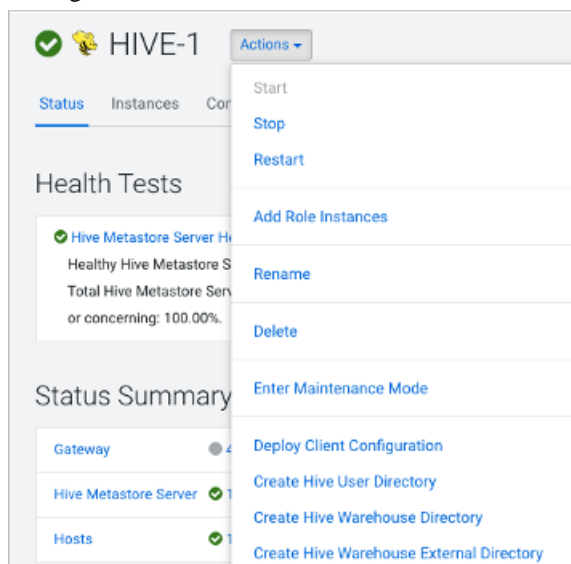
The upgrade process sets the `hive.metastore.warehouse.dir` property to `/warehouse/tablespace/managed/hive`, designating it the Hive warehouse location for managed tables. New managed tables that you create in CDP are stored in the Hive warehouse. New external tables are stored in the Hive external warehouse `/warehouse/tablespace/external/hive`.

To change the location of the Hive warehouses, you navigate to one of the following menu items in the first step below.

- Hive Action Menu Create Hive Warehouse Directory
- Hive Action Menu Create Hive Warehouse External Directory

Procedure

1. Set up directories for the Hive warehouse directory and Hive warehouse external directory from Cloudera Manager Actions.



2. In Cloudera Manager, click Clusters Hive (the Hive Metastore service) Configuration , and change the hive.metastore.warehouse.dir property value to the path you specified for the new Hive warehouse directory.
3. Change the hive.metastore.warehouse.external.dir property value to the path you specified for the Hive warehouse external directory.
4. Configure Ranger policies or set up ACL permissions to access the directories.

Changes to HDP Hive tables

As a Data Scientist, Architect, Analyst, or other Hive user you need to locate and use your Apache Hive 3 tables after an upgrade. You also need to understand the changes that occur during the upgrade process.

Managed, ACID tables that are not owned by the hive user remain managed tables after the upgrade, but hive becomes the owner.

After the upgrade, the format of a Hive table is the same as before the upgrade. For example, native or non-native tables remain native or non-native, respectively.

After the upgrade, the location of managed tables or partitions do not change under any one of the following conditions:

- The old table or partition directory was not in its default location /apps/hive/warehouse before the upgrade.
- The old table or partition is in a different file system than the new warehouse directory.
- The old table or partition directory is in a different encryption zone than the new warehouse directory.

Otherwise, the upgrade process from HDP to CDP moves managed files to the Hive warehouse /warehouse/tablespace/managed/hive. The upgrade process carries the external files over to CDP with no change in location. By default, Hive places any new external tables you create in /warehouse/tablespace/external/hive. The upgrade process sets the hive.metastore.warehouse.dir property to this location, designating it the Hive warehouse location.

Changes to table references using dot notation

Upgrading to CDP includes the Hive-16907 bug fix, which rejects `db.table` in SQL queries. The dot (.) is not allowed in table names. To reference the database and table in a table name, both must be enclosed in backticks as follows: `db`.`table`.

Changes to ACID properties

Hive 3.x in CDP Private Cloud Base supports transactional and non-transactional tables. Transactional tables have atomic, consistent, isolation, and durable (ACID) properties. In Hive 2.x, the initial version of ACID transaction processing was ACID v1. In Hive 3.x, the mature version of ACID is ACID v2, which is the default table type in CDP Private Cloud Base.

Native and non-native storage formats

Storage formats are a factor in upgrade changes to table types. Hive 2.x and 3.x support the following native and non-native storage formats:

- Native: Tables with built-in support in Hive, such as those in the following file formats:
 - Text
 - Sequence File
 - RC File
 - AVRO File
 - ORC File
 - Parquet File
- Non-native: Tables that use a storage handler, such as the `DruidStorageHandler` or `HBaseStorageHandler`

CDP upgrade changes to HDP table types

The following table compares Hive table types and ACID operations before an upgrade from HDP 2.x and after an upgrade to CDP. The ownership of the Hive table file is a factor in determining table types and ACID operations after the upgrade.

Table 11: HDP 2.x and CDP Table Type Comparison

HDP 2.x				CDP	
Table Type	ACID v1	Format	Owner (user) of Hive Table File	Table Type	ACID v2
External	No	Native or non-native	hive or non-hive	External	No
Managed	Yes	ORC	hive or non-hive	Managed, updatable	Yes
Managed	No	ORC	hive	Managed, updatable	Yes
			non-hive	External, with data delete	No
Managed	No	Native (but non-ORC)	hive	Managed, insert only	Yes
			non-hive	External, with data delete	No
Managed	No	Non-native	hive or non-hive	External, with data delete	No

Transitioning from the Cloudera Manager Embedded PostgreSQL Database Server to an External PostgreSQL Database

Navigation title: Transitioning Embedded PostgreSQL Database to External PostgreSQL Database

Cloudera Manager provides an embedded PostgreSQL database server for demonstration and proof of concept deployments when creating a cluster. To remind users that this embedded database is not suitable for production, Cloudera Manager displays the banner text: "You are running Cloudera Manager in non-production mode, which uses an embedded PostgreSQL database. Switch to using a supported external database before moving into production."

If, however, you have already used the embedded database, and you are unable to redeploy a fresh cluster, then you must migrate to an external PostgreSQL database.



Note: This procedure does not describe how to migrate to a database server other than PostgreSQL. Moving databases from one database server to a different type of database server is a complex process that requires modification of the schema and matching the data in the database tables to the new schema. It is strongly recommended that you engage with Cloudera Professional Services if you wish to perform a transition to an external database server other than PostgreSQL.

Prerequisites

Before migrating the Cloudera Manager embedded PostgreSQL database to an external PostgreSQL database, ensure that your setup meets the following conditions:

- The external PostgreSQL database server is running.
- The database server is configured to accept remote connections.
- The database server is configured to accept user logins using md5.
- No one has manually created any databases in the external database server for roles that will be migrated.



Note: To view a list of databases in the external database server (requires default superuser permission):

```
sudo -u postgres psql -l
```

- All health issues with your cluster have been resolved.

For details about configuring the database server, see [Configuring and Starting the PostgreSQL Server](#).



Important: Only perform the steps in [Configuring and Starting the PostgreSQL Server](#). Do not proceed with the creation of databases as described in the subsequent section.

For large clusters, Cloudera recommends running your database server on a dedicated host. Engage Cloudera Professional Services or a certified database administrator to correctly tune your external database server.

Identify Roles that Use the Embedded Database Server

Before you can migrate to another database server, you must first identify the databases using the embedded database server. When the Cloudera Manager Embedded Database server is initialized, it creates the Cloudera Manager database and databases for roles in the Management Services. The Installation Wizard (which runs automatically the first time you log in to Cloudera Manager) or Add Service action for a cluster creates additional databases for roles when run. It is in this context that you identify which roles are used in the embedded database server.

To identify which roles are using the Cloudera Manager embedded database server:

1. Obtain and save the cloudera-scm superuser password from the embedded database server. You will need this password in subsequent steps:

```
head -1 /var/lib/cloudera-scm-server-db/data/generated_password.txt
```

2. Make a list of all services that are using the embedded database server. Then, after determining which services are not using the embedded database server, remove those services from the list. The scm database must remain in your list. Use the following table as a guide:

Table 12: Cloudera Manager Embedded Database Server Databases

Service	Role	Default Database Name	Default Username
Cloudera Manager Server		scm	scm
Cloudera Management Service	Activity Monitor	amon	amon
Hive	Hive Metastore Server	hive	hive

Service	Role	Default Database Name	Default Username
Hue	Hue Server	hue	7uu7uu7uhue
Cloudera Management Service	Navigator Audit Server	nav	nav
Cloudera Management Service	Navigator Metadata Server	navms	navms
Oozie	Oozie Server	oozie_oozie_server	oozie_oozie_server
Cloudera Management Service	Reports Manager	rman	rman
Sentry	Sentry Server	sentry	sentry

3. Verify which roles are using the embedded database. Roles using the embedded database server always use port 7432 (the default port for the embedded database) on the Cloudera Manager Server host.

For Cloudera Management Services:

- a. Select Cloudera Management Service > Configuration, and type "7432" in the Search field.
- b. Confirm that the hostname for the services being used is the same hostname used by the Cloudera Manager Server.



Note:

If any of the following fields contain the value "7432", then the service is using the embedded database:

- Activity Monitor
- Navigator Audit Server
- Navigator Metadata Server
- Reports Manager

For the Oozie Service:

- a. Select Oozie service > Configuration, and type "7432" in the Search field.
- b. Confirm that the hostname is the Cloudera Manager Server.

For Hive, Hue, and Sentry Services:

- a. Select the specific service > Configuration, and type "database host" in the Search field.
 - b. Confirm that the hostname is the Cloudera Manager Server.
 - c. In the Search field, type "database port" and confirm that the port is 7432.
 - d. Repeat these steps for each of the services (Hive, Hue and Sentry).
4. Verify the database names in the embedded database server match the database names on your list (Step 2). Databases that exist on the database server and not used by their roles do not need to be migrated. This step is to confirm that your list is correct.



Note: Do not add the postgres, template0, or template1 databases to your list. These are used only by the PostgreSQL server.

```
psql -h localhost -p 7432 -U cloudera-scm -l
```

```
Password for user cloudera-scm: <password>
```

Name		Owner	List of databases		
	Access		Encoding	Collate	Ctype
amon		amon	UTF8	en_US.UTF8	en_US.U
TF8					
hive		hive	UTF8	en_US.UTF8	en_US.UT
F8					

hue	hue	UTF8	en_US.UTF8	en_US
.UTF8				
nav	nav	UTF8	en_US.UTF8	en_US.
UTF8				
navms	navms	UTF8	en_US.UTF8	en_US.U
TF8				
oozie_oozie_server	oozie_oozie_server	UTF8	en_US.UTF8	en_US.UT
F8				
postgres	cloudera-scm	UTF8	en_US.UTF8	en_US
.UTF8				
rman	rman	UTF8	en_US.UTF8	en_US.
UTF8				
scm	scm	UTF8	en_US.UTF8	en_US.U
TF8				
sentry	sentry	UTF8	en_US.UTF8	en_US.UT
F8				
template0	cloudera-scm	UTF8	en_US.UTF8	en_US
.UTF8	=c/"cloudera-scm"			
template1	cloudera-scm	UTF8	en_US.UTF8	en_US.UT
F8	=c/"cloudera-scm"			
(12 rows)				

You should now have a list of all roles and database names that use the embedded database server, and are ready to proceed with the transition of databases from the embedded database server to the external PostgreSQL database server.

Migrate Databases from the Embedded Database Server to the External PostgreSQL Database Server

While performing this procedure, ensure that the Cloudera Manager Agents remain running on all hosts. Unless otherwise specified, when prompted for a password use the cloudera-scm password.



Note: After completing this transition, you cannot delete the cloudera-scm postgres superuser unless you remove the access privileges for the migrated databases. Minimally, you should change the cloudera-scm postgres superuser password.

1. In Cloudera Manager, stop the cluster services identified as using the embedded database server (see [Identify Roles that Use the Embedded Database Server](#) on page 35). Be sure to stop the Cloudera Management Service as well. Also be sure to stop any services with dependencies on these services. The remaining services will continue to run without downtime.



Note: If you do not stop the services from within Cloudera Manager before stopping Cloudera Manager Server from the command line, they will continue to run and maintain a network connection to the embedded database server. If this occurs, then the embedded database server will ignore any command line stop commands (Step 2) and require that you manually kill the process, which in turn causes the services to crash instead of stopping cleanly.

2. Navigate to Hosts > All Hosts, and make note of the number of roles assigned to hosts. Also take note whether or not they are in a commissioned state. You will need this information later to validate that your scm database was migrated correctly.
3. Stop the Cloudera Manager Server. To stop the server:

```
sudo service cloudera-scm-server stop
```

4. Obtain and save the embedded database superuser password (you will need this password in subsequent steps) from the generated_password.txt file:

```
head -1 /var/lib/cloudera-scm-server-db/data/generated_password.txt
```

5. Export the PostgreSQL user roles from the embedded database server to ensure the correct users, permissions, and passwords are preserved for database access. Passwords are exported as an md5sum and are not visible in plain text. To export the database user roles (you will need the cloudera-scm user password):

```
pg_dumpall -h localhost -p 7432 -U cloudera-scm -v --roles-only -f "/var/
tmp/cloudera_user_roles.sql"
```

6. Edit /var/tmp/cloudera_user_roles.sql to remove any CREATE ROLE and ALTER ROLE commands for databases not in your list. Leave the entries for cloudera-scm untouched, because this user role is used during the database import.
7. Export the data from each of the databases on your list you created in [Identify Roles that Use the Embedded Database Server](#) on page 35:

```
pg_dump -F c -h localhost -p 7432 -U cloudera-scm [database_name] > /var/
tmp/[database_name]_db_backup-$(date +%m-%d-%Y).dump
```

Following is a sample data export command for the scm database:

```
pg_dump -F c -h localhost -p 7432 -U cloudera-scm scm > /var/tmp/scm_db_
backup-$(date +%m-%d-%Y).dump
```

Password:

8. Stop and disable the embedded database server:

```
service cloudera-scm-server-db stop
chkconfig cloudera-scm-server-db off
```

Confirm that the embedded database server is stopped:

```
netstat -at | grep 7432
```

9. Back up the Cloudera Manager Server database configuration file:

```
cp /etc/cloudera-scm-server/db.properties /etc/cloudera-scm-server/db.pr
operties.embedded
```

10. Copy the file /var/tmp/cloudera_user_roles.sql and the database dump files from the embedded database server host to /var/tmp on the external database server host:

```
cd /var/tmp
scp cloudera_user_roles.sql *.dump <user>@<postgres-server>:/var/tmp
```

11. Import the PostgreSQL user roles into the external database server.

The external PostgreSQL database server superuser password is required to import the user roles. If the superuser role has been changed, you will be prompted for the username and password.



Note: Only run the command that applies to your context; do not execute both commands.

- To import users when using the default PostgreSQL superuser role:

```
sudo -u postgres psql -f /var/tmp/cloudera_user_roles.sql
```

- To import users when the superuser role has been changed:

```
psql -h <database-hostname> -p <database-port> -U <superuser> -f /var/tmp/cloudera_user_roles.sql
```

For example:

```
psql -h pg-server.example.com -p 5432 -U postgres -f /var/tmp/cloudera_user_roles.sql
```

```
Password for user postgres
```

12. Import the Cloudera Manager database on the external server. First copy the database dump files from the Cloudera Manager Server host to your external PostgreSQL database server, and then import the database data:



Note: To successfully run the `pg_restore` command, there must be an existing database on the database server to complete the connection; the existing database will not be modified. If the `-d <existing-database>` option is not included, then the `pg_restore` command will fail.

```
pg_restore -C -h <database-hostname> -p <database-port> -d <existing-database> -U cloudera-scm -v <data-file>
```

Repeat this import for each database.

The following example is for the scm database:

```
pg_restore -C -h pg-server.example.com -p 5432 -d postgres -U cloudera-scm -v /var/tmp/scm_server_db_backup-20180312.dump
```

```
pg_restore: connecting to database for restore
Password:
```

13. Update the Cloudera Manager Server database configuration file to use the external database server. Edit the `/etc/cloudera-scm-server/db.properties` file as follows:

- Update the `com.cloudera.cmf.db.host` value with the hostname and port number of the external database server.
- Change the `com.cloudera.cmf.db.setupType` value from "EMBEDDED" to "EXTERNAL".

14. Start the Cloudera Manager Server and confirm it is working:

```
service cloudera-scm-server start
```

Note that if you start the Cloudera Manager GUI at this point, it may take up to five minutes after executing the start command before it becomes available.

In Cloudera Manager Server, navigate to Hosts > All Hosts and confirm the number of roles assigned to hosts (this number should match what you found in Step 2); also confirm that they are in a commissioned state that matches what you observed in Step 2.

15. Update the role configurations to use the external database hostname and port number. Only perform this task for services where the database has been migrated.

For Cloudera Management Services:

- a. Select Cloudera Management Service > Configuration, and type "7432" in the Search field.
- b. Change any database hostname properties from the embedded database to the external database hostname and port number.
- c. Click Save Changes.

For the Oozie Service:

- a. Select Oozie service > Configuration, and type "7432" in the Search field.
- b. Change any database hostname properties from the embedded database to the external database hostname and port number.
- c. Click Save Changes.

For Hive, Hue, and Sentry Services:

- a. Select the specific service > Configuration, and type "database host" in the Search field.
- b. Change the hostname from the embedded database name to the external database hostname.
- c. Click Save Changes.

16. Start the Cloudera Management Service and confirm that all management services are up and no health tests are failing.
17. Start all Services via the Cloudera Manager web UI. This should start all services that were stopped for the database transition. Confirm that all services are up and no health tests are failing.
18. On the embedded database server host, remove the embedded PostgreSQL database server:

- a. Make a backup of the /var/lib/cloudera-scm-server-db/data directory:

```
tar czvf /var/tmp/embedded_db_data_backup-$(date +%m-%d-%Y).tgz /var/lib/cloudera-scm-server-db/data
```

- b. Remove the embedded database package:

For RHEL/SLES:

```
rpm --erase cloudera-manager-server-db-2
```

For Ubuntu:

```
apt-get remove cloudera-manager-server-db-2
```

- c. Delete the /var/lib/cloudera-scm-server-db/data directory.