# Red Hat build of Debezium 2.5.4

## Release Notes for Red Hat build of Debezium 2.5.4

What's new in Red Hat build of Debezium

Last Updated: 2024-04-24

# Red Hat build of Debezium 2.5.4 Release Notes for Red Hat build of Debezium 2.5.4

What's new in Red Hat build of Debezium

## Legal Notice

Copyright © 2024 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at
http://creativecommons.org/licenses/by-sa/3.0/
. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

## Abstract

Describes the Red Hat build of Debezium product and provides the latest details on what's new in this release.

# Table of Contents

# CHAPTER 1. RED HAT BUILD OF DEBEZIUM

Red Hat build of Debezium is a comprehensive set of integration and event processing technologies for creating, extending, and deploying container-based integration services across hybrid and multicloud environments. Red Hat build of Debezium provides an agile, distributed, and API-centric solution that organizations can use to connect and share data between applications and systems required in a digital world.

Red Hat build of Debezium includes the following capabilities:

- Real-time messaging
- Cross-datacenter message streaming
- API connectivity
- Application connectors
- Enterprise integration patterns
- API management
- Data transformation
- Service composition and orchestration

**Additional resources**

- Understanding enterprise integration

# CHAPTER 2. DEBEZIUM 2.5.4 RELEASE NOTES

Debezium is a distributed change data capture platform that captures row-level changes that occur in database tables and then passes corresponding change event records to Apache Kafka topics. Applications can read these *change event streams* and access the change events in the order in which they occurred. Debezium is built on Apache Kafka and is deployed and integrated with AMQ Streams on OpenShift Container Platform or on Red Hat Enterprise Linux.

The following topics provide release details:



## 2.1. DEBEZIUM DATABASE CONNECTORS

Debezium provides connectors based on Kafka Connect for the following common databases:

- Db2
- JDBC Sink connector
- MongoDB
- MySQL
- Oracle
- PostgreSQL
- SQL Server

### 2.1.1. Connector usage notes

- Db2

- The Debezium Db2 connector does not include the Db2 JDBC driver (**jcc-11.5.0.0.jar**). See the Db2 connector deployment instructions for information about how to deploy the necessary JDBC driver.
- The Db2 connector requires the use of the abstract syntax notation (ASN) libraries, which are available as a standard part of Db2 for Linux.
- To use the ASN libraries, you must have a license for IBM InfoSphere Data Replication (IIDR). You do not have to install IIDR to use the libraries.

- MySQL
  - Beginning with this release, use of the MySQL connector with MariaDB is available as a Technology Preview feature. For more information see Section 2.4.4.1, "Use of the Debezium connector for MySQL with MariaDB".
- Oracle
  - The Debezium Oracle connector does not include the Oracle JDBC driver (**ojdbc8.jar**). See the Oracle connector deployment instructions for information about how to deploy the necessary JDBC driver.
- PostgreSQL
  - To use the Debezium PostgreSQL connector you must use the **pgoutput** logical decoding output plug-in, which is the default for PostgreSQL versions 10 and later.

**Additional resources**

- Getting Started with Debezium
- Debezium User Guide

## 2.2. DEBEZIUM SUPPORTED CONFIGURATIONS

For information about Debezium supported configurations, including information about supported database versions, see the Debezium 2.5.4 Supported configurations page .

### 2.2.1. AMQ Streams API version

Debezium runs on AMQ Streams 2.6.

AMQ Streams supports the **v1beta2** API version, which updates the schemas of the AMQ Streams custom resources. Older API versions are deprecated.

## 2.3. DEBEZIUM INSTALLATION OPTIONS

You can install Debezium with AMQ Streams on OpenShift or on Red Hat Enterprise Linux:

- Installing Debezium on OpenShift
- Installing Debezium on RHEL

## 2.4. DEBEZIUM 2.5.4 FEATURES AND IMPROVEMENTS

### 2.4.1. Breaking changes

Breaking changes represent significant differences in connector behavior and require configuration changes that are not compatible with earlier Debezium versions.

Debezium 2.5.4 introduces breaking changes that affect the following components:



For information about breaking changes in the previous Debezium release, see the Debezium 2.3.7 Release Notes.

#### 2.4.1.1. Breaking changes relevant to all connectors

**ComputePartition SMT removed (DBZ-7141)**

The **ComputePartition** single message transformation (SMT) was formerly used to compute the target Kafka topic partition to which Debezium wrote an event record. This SMT was previously deprecated and is now removed. Use the **PartitionRouting** SMT in its place.

#### 2.4.1.2. CloudEvents converter breaking changes

**metadata.location property renamed to metadata.source (DBZ-7060)**

The CloudEvents configuration property **metadata.location** is superseded by the **metadata.source** property. If you use use the CloudEvents converter to emit events that conform to the CloudEvents format, update the property in your connector configurations. For more information, see Configuration of sources of metadata and some CloudEvents fields in the Debezium User Guide.

**CloudEvent header changes (DBZ-7216)**

The schema name prefix and letter casing for CloudEvent headers was not consistent with the payload name. The schema name is now aligned so that the event headers and payload share the same namespace and follow the same rules for letter casing.

#### 2.4.1.3. JDBC sink connector breaking changes

**JDBC sink value serialization changes (DBZ-7191)**

In some cases, when the JDBC sink connector consumed an event that included fields with **null** values, when writing the event record to the target database, it incorrectly wrote the default field value rather, than the the value **NULL**.

#### 2.4.1.4. MongoDB connector breaking changes

**New read preferences (DBZ-6591)**

Previous versions of MongoDB connector used a hard-coded **secondaryPreferred** read preference. This behavior conflicts with the default and recommended MongoDB configuration in which all MongoDB operations read from the **primary** member. Beginning with Debezium 2.5.4 the connector now always honors the read preference set in connection string.

If you prefer to retain the original behavior, customize the connector configuration by adding the following option to the connection string:

```
&readPreference=secondaryPreferred
```

If the connector reads from a sharded cluster, and the **mongodb.connection.mode** is set to **replica_set**, use the new **mongodb.connection.string.shard.params** property to specify a connection string that sets the read preference to **secondary**. For more information, see the documentation for the connector in the Debezium User Guide.

**Default connection mode for sharded clusters now set to sharded (DBZ-7108)**

In a sharded cluster the default connection mode for the MongoDB connector is now **sharded**, rather than **replica_set**. Using the **replica_set** mode to establish connections to a sharded cluster bypasses the **mongos** router to connect directly to a shard. This configuration can lead to problems and is discouraged by MongoDB.

After you upgrade the connector, the new default setting invalidates existing offsets, and triggers a silent re-execution of the initial snapshot (DBZ-7272). To prevent this behavior, a check was added that fails the connector if the restart would trigger a snapshot.

If your connector reads from a sharded cluster, after you upgrade, review your connector configuration and adjust the connection mode as needed. You can either set the **replica_set** connection mode explicitly, or remove the existing offsets.

**MongoDB snapshots in sharded deployments (DBZ-7139)**

In earlier versions of Debezium, when you used the MongoDB connector in a sharded deployment, you could specify the shard that the connector used to perform a snapshot of a collection. Unfortunately, although some customers found this behavior useful, it was unintended and is not supported. This option is no longer available.

#### 2.4.1.5. MySQL connector breaking changes

**MySQL BIT default length (DBZ-7230)**

The MySQL **BIT** data type does not have an implicit length if any was not set. This is incorrect as the default length if none is provided is **1** . users who use schema registry should not experience any change in the output schema.

**MySQL BIGINT precision changed (DBZ-6714)**

In earlier releases, when the **bigint.unsigned.handling.mode** property was set to **precise** in MySQL connector configuration, the connector did not apply the correct precision when it converted columns for **BIGINT** data types. As a result, the schema for **BIGINT** fields included inaccurate precision values. The decimal precision is now set to **20** to represent unsigned **BIGINT** values.

If you use a schema registry with a connector that is configured with the preceding settings, this change can lead to incompatibilities if the registry enforces strict compatibility rules. After you upgrade the connector, take action as needed to ensure that output values are accurately represented, with no loss of precision, for example, adjusting compatibility settings.

**MySQL 5.7 support now best-effort (DBZ-6874)**

MySQL 5.7 recently entered its end of life cycle and will no longer receive updates or security patches. In accordance with this announcement, beginning with Debezium 2.5.4 use of MySQL 5.7 is no longer considered a supported configuration.

## 2.4.2. Features promoted to General Availability

The following features are promoted from Developer Preview or Technology Preview to General Availability in the Debezium 2.5.4 release:

- Section 2.4.2.1, "JDBC sink connector (Promoted from Developer Preview)"
- Section 2.4.2.2, "MongoDB incremental snapshots for multi-replica and sharded cluster (Promoted from Technology Preview)"
- Section 2.4.2.3, "MongoDB Sharded Cluster Improvements (Promoted from Technology Preview)"

### 2.4.2.1. JDBC sink connector (Promoted from Developer Preview)

Unlike implementations from other vendors,the Debezium JDBC sink connector can ingest change events emitted by Debezium source connectors without the need to first process through an event flattening transoformation. The Debezium JDBC connector can reduce the processing footprint of your pipeline and simplify its configuration. The connector can also take advantage of Debezium source connector features, such as column type propagation. For more information, see the Debezium User Guide

### 2.4.2.2. MongoDB incremental snapshots for multi-replica and sharded cluster (Promoted from Technology Preview)

Use of the MongoDB connector to perform incremental snapshots multi-replica a sharded clusters is now fully supported. For more information, see the Debezium User Guide.

### 2.4.2.3. MongoDB Sharded Cluster Improvements (Promoted from Technology Preview)

In previous versions, when you used the Debezium connector for MongoDB in a sharded cluster deployment, the connector opened connections with each of the shard's replica sets directly. This is not a recommended approach and instead MongoDB suggests that the connector open a connection with the mongos instance (the router) instead.

This release aligns with the recommended strategy. It might be necessary to adjust your configurations to point the connector to the **mongos** instance.

To retain backwards compatibility, a new configuration property mongodb.connection.mode is available. The default value for this property, **replica_set** preserves the current behavior so that the connector processes each shard as an individual replica set. Set the value to **sharded** to instruct the connector to internally create a single instance of ReplicaSet with an artificially named cluster. This configuration results in the connector using the initially supplied connection string to process connections to all members of ths shard.

For more information, see MongoDB sharded cluster support.

### 2.4.3. General availability features

Debezium 2.5.4 provides new features for the following connectors:

- Section 2.4.3.1, "Ad hoc blocking snapshots (all source connectors) (DBZ-6566)"
- Section 2.4.3.3, "Custom tags in the connector metrics (all source connectors) DBZ-6603"
- Section 2.4.3.5, "Initial snapshot progress notifications (all connectors) (DBZ-6416, DBZ-6878)"
- Section 2.4.3.6, "INSERT/DELETE for incremental snapshot watermarking (all connectors) (DBZ-6834)"
- Section 2.4.3.7, "Reselect columns post processor to fill missing Large Objects (LOBs) (DBZ-7602)"
- Section 2.4.3.2, "Timezone conversion SMT (all source connectors) (DBZ-6567)"
- Section 2.4.3.4, "Support for MongoDB large messages (DBZ-6726)"
- Section 2.4.3.8, "Post-image support in MongoDB (DBZ-7299, DBZ-7647)"

#### 2.4.3.1. Ad hoc blocking snapshots (all source connectors) ( DBZ-6566)

Incremental snapshots are a popular method for handling a variety of re-snapshot use cases. However, in some cases where snapshot read events are intertwined with streaming events (create, update, and delete), the use of incremental snapshots might not be indicated or even supported by a consuming application. In these cases, you can perform an ad hoc blocking snapshot.

Although you trigger both ad hoc blocking snapshots and ad hoc incremental snapshots by sending a signal to Debezium, when the connector processes the blocking snapshot signal, it places a hold on streaming for the duration of the snapshot process. As a result, the snapshot maintains separation between the snapshot read events and streaming create, update, and delete events. In other words, the blocking snapshot process is similar to the process followed by a traditional snapshot. As a result, the throughput is generally higher than with incremental snapshots.

**IMPORTANT**

In an ad-hoc blocking snapshot, the reading of the transaction logs is placed on hold during the snapshot. As with a traditional snapshot, transaction logs must be availabe when using an ad-hoc snapshot mode. When streaming resumes, if a transaction log that is needed has since been removed, the connector will raise an error and stop.

The signal to initiate an ad hoc blocking snapshot is similar to the signal that you use to initiate an ad hoc incremental snapshot. The following example shows the payload of a signal to initiate a blocking snapshot a specific table with a condition. The only difference between this signal and the signal for an incremental snapshots is that the **type** of the signal is set to **BLOCKING** rather than **INCREMENTAL**.

```
{
  "type": "execute-snapshot",
  "data": {
    "data-collections": ["public.my_table"],
```

```
    "type": "BLOCKING",
    "additional-condition": "last_update_date >= '2023-01-01'"
  }
}
```

For more information, see *Blocking snapshots* in the Debezium User Guide chapter for your connector type.

#### 2.4.3.2. Timezone conversion SMT (all source connectors) ( DBZ-6567)

Customers often request the ability to emit temporal columns in which the time zone is set to a value other than **UTC**. In past releases you could create a **CustomConverter**, or write your own single message transformation to specify how a connector emitted temporal columns.

In this release, Debezium provides a new time zone transformation that enables you to modify the timezone of temporal columns in events that the connector emits. You can convert columns from their default UTC timezone to any other time zone, based on the needs of your pipeline. To get started with this new transformation, add the following basic configuration to your connector:

```
{
  "transforms": "tz",
  "transforms.tz.type": "io.debezium.transforms.TimezoneConverter",
  "transforms.tz.converted.timezone": "America/New_York"
}
```

After you add the preceding configuration, when the connector emits an event, all temporal columns that were in UTC are converted to the` America/New_York` time zone.

Not only can you change the timezone for temporal fields, you can also modify timezone settings for other fields, by using the **include.fields** property as shown in the following example:

```
{
  "transforms": "tz",
  "transforms.tz.type": "io.debezium.transforms.TimezoneConverter",
  "transforms.tz.converted.timezone": "America/New_York",
  "transforms.tz.include.fields": "source:customers:created_at,customers:updated_at"
}
```

In the preceding example, the first value in **include.fields** converts the timezone of the **created_at** field for the **source** table with the name **customers**. The next value converts the **updated_at** field for the topic with the name **customers**.

Or, if you want to apply the conversion to all but subset of fields, you can set the **exclude.fields** property, as in the following example:

```
{
  "transforms": "tz",
  "transforms.tz.type": "io.debezium.transforms.TimezoneConverter",
  "transforms.tz.converted.timezone": "America/New_York",
  "transforms.tz.exclude.fields": "source:customers:updated_at"
}
```

In the preceding example, the transformation converts all temporal fields to the **America/New_York** timezone, except where the **source** table name is **customers**, and the field is **updated_at**. For more information, see Converting timezone values in Debezium event records in the Debezium User Guide.

### 2.4.3.3. Custom tags in the connector metrics (all source connectors) DBZ-6603

You can now specify custom tags to append to the name of the connector MBean object name. Custom tags enable you to configure more reliable connector monitoring.

Debezium connectors expose metrics that provide data about the behavior of snapshots, streaming, and schema history processes. These metrics are available for each connector, and are exposed via the MBean name for the connector.

By default, when you deploy a correctly configured connector, Debezium generates a unique MBean for the connector. You can configure your observability stack to monitor the metrics associated with that MBean. If you later change the connector configuration, these changes can modify the original MBean name. If the MBean name changes, the linkage between the connector instance and the MBean breaks, and monitoring tools can no longer retrieve metrics data for the connector unless you reconfigure the observability stack to use the new MBean name.

To avoid the potential churn that results from MBean name changes, you can configure connectors to use custom metrics tags by adding the the **custom.metric.tags** property to the connector configuration. The **custom.metric.tags** property accepts key-value pairs. Each key represents a tag for the MBean object name, and the corresponding value represents the value of that tag. For example: **k1=v1,k2=v2**.

After you configure the **custom.metric.tags** property for a connector, configure the observability stack to retrieve metrics associated with the specified tags. The observability stack then uses the specified tags, rather than the mutable MBean names to uniquely identify connectors. Later, if Debezium redefines how it constructs MBean names, or if the **topic.prefix** in the connector configuration changes, metrics collection is uninterrupted, because the metrics scrape task uses the specified tag patterns to identify the connector.

A further benefit of using custom tags, is that you can use tags that reflect the architecture of your data pipeline, so that metrics are organized in a way that suits you operational needs. For example, you might specify tags with values that declare the type of connector activity, the application context, or the data source, for example, **db1-streaming-for-application-abc**. If you specify multiple key-value pairs, all of the specified pairs are appended to the connector's MBean name.

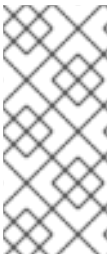
The following example illustrates how tags modify the default MBean name.

> **Example 2.1. How custom tags modify the connector MBean name**
>
> By default, the Oracle connector uses the following MBean name for streaming metrics:
>
> ```
> debezium.oracle:type=connector-metrics,context=streaming,server=<topic.prefix>
> ```
>
> If you set the value of **custom.metric.tags** to **database=salesdb-streaming,table=inventory**, Debezium generates the following custom MBean name:
>
> ```
> debezium.oracle:type=connector-metrics,context=streaming,server=
> <topic.prefix>,database=salesdb-streaming,table=inventory
> ```

### 2.4.3.4. Support for MongoDB large messages (DBZ-6726)

The Debezium MongoDB connector now supports large document processing for MongoDB 6. This change prevents the connector from returning a **BSONObjectTooLarge** exception when it attempts to capture a document that exceeds 16MB. To configure the connector to take advantage of this

capability, set the **oversize.handling.mode** option.



NOTE

Support for MongoDB large messages relies on an underlying feature of the MongoDB database. Because the database imposes limitations on processing large documents, the connector might still return an exception if it attempts to capture the document that does not comply with the rules for dividing large documents. For more information, see the MongoDB documentation.

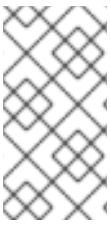
#### 2.4.3.5. Initial snapshot progress notifications (all connectors) ( DBZ-6416, DBZ-6878)

The Debezium notification subsystem provides an easy way to integrate third-party tools and applications with Debezium to gain insight into the ongoing change data capture process, beyond the traditional JMX approach. In earlier release, the Debezium notification subsystem provided only basic information about initial snapshots, such as when the snapshot started, when each table started and concluded, and when the snapshot ended. The notification subsystem now includes the ability to notify you about the ongoing status of an initial snapshot.

Initial snapshot notifications are emitted with an **aggregatetType** of **Initial Snapshot** and contain a **type** field that exposes the current status of the snapshot. The possible values include: **STARTED**, **ABORTED**, **PAUSED**, **RESUMED**, **IN_PROGRESS**, **TABLE_SCAN_COMPLETED**, and **COMPLETED**.

DBZ-6878 extends the basic functionality to provide enhanced details about the snapshot. For example, the **IN_PROGRESS** notification now provides additional details about the tables being captured and which table is currently in-progress, as in the following example:

```
{
  "id":"6d82a3ec-ba86-4b36-9168-7423b0dd5c1d",
  "aggregate_type":"Initial Snapshot",
  "type":"IN_PROGRESS",
  "additional_data":{
    "connector_name":"my-connector",
    "data_collections":"table1, table2",
    "current_collection_in_progress":"table1"
  },
  "timestamp": "1695817046353"
}
```



NOTE

Some fields in the preceding example, such as **data_collection**, are not currently available for MongoDB snapshots, and are only available for SQL-based relational connectors. For more information see Debezium notification that reports on the status of an initial snapshot in the Debezium User Guide.

#### 2.4.3.6. INSERT/DELETE for incremental snapshot watermarking (all connectors) ( DBZ-6834)

This release introduces the property **incremental.snapshot.watermarking.strategy**, which permits you to specify the watermarking strategy to use during an incremental snapshot. Past releases, use the **insert_insert** approach, in which Debezium created two entries in the signaling data collection during the snapshot for each chunk. The first entry signaled the opening of the snapshot window, while the next entry marked its closure.

The new **insert_delete** option writes a single entry to the signaling data collection for each chunk at the beginning of the snapshot window. After the snapshot completes, the entry is removed, and no corresponding entry is added to signify the closure of the snapshot window. This method provides for more efficient management of the signaling data collection.

#### 2.4.3.7. Reselect columns post processor to fill missing Large Objects (LOBs) ( DBZ-7602)

Because of the way that certain source databases function, when a Debezium connector emits a change event, the event might exclude values for column types that store large volumes of data. For example, values for **TOAST** columns in PostgreSQL, **LOB** columns in Oracle, or **Extended String** columns in Oracle Exadata, might all be excluded from a change event.

Debezium 2.5.4 introduces a *Reselect columns post processor*, which provides a mechanism for reselecting one or more columns that the connector captures from a database table, and fetching the current state of those columns. You can configure the post processor to re-select the following column types:

- **null** columns
- Columns that contain the **unavailable.value.placeholder** sentinel value.

Configuring a PostProcessor is similar to configuring a custom converter or single message transformation, except that it works on the mutable payload's **Struct** rather than on the **SourceRecord**. For more information, see using-the-reselect-columns-post-processor-to-add-source-fields-to-change-event-records in the Debezium User Guide.

#### 2.4.3.8. Post-image support in MongoDB (DBZ-7299, DBZ-7647)

When the Debezium connector for MongoDB emits a change event, you can configure the connector so that the event payload includes the full source document that was changed in an update. In past releases, the connector did not consistently stream full documents in the event payload.

To explicitly control how the connector looks up the full document in a change stream, set the configuration option, **capture.mode.full.update.type**. The default value for this option looks up the full document, which requires the database to perform a separate look-up to fetch the full document. If you use the connector with MongoDB 6 or greater, you can configure the connector to use post_image to rely on MongoDB change stream's post-image support.

For more information, see capture.mode.full.update.type in the MongoDB connector chapter of the Debezium User Guide.

### 2.4.4. Technology Preview features

The following Technology Preview features are available in Debezium 2.5.4:

- Section 2.4.4.1, "Use of the Debezium connector for MySQL with MariaDB"
- Section 2.4.4.2, "Streaming from a PostgreSQL 16 standby (DBZ-7181)"

#### Previously available Technology Preview features

The following features that were introduced in earlier releases remain in Technology Preview:

- Section 2.4.4.3, "CloudEvents converter"
- Section 2.4.4.4, "Custom converters"

- Section 2.4.4.5, “Use of the **BLOB**, **CLOB**, and **NCLOB** data types with the Oracle connector”
- Section 2.4.4.6, “Parallel initial snapshots for SQL connectors”

**IMPORTANT**

Technology Preview features are not supported with Red Hat production service-level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend implementing any Technology Preview features in production environments. Technology Preview features provide early access to upcoming product innovations, enabling you to test functionality and provide feedback during the development process. For more information about support scope, see Technology Preview Features Support Scope.

#### 2.4.4.1. Use of the Debezium connector for MySQL with MariaDB

Beginning with Debezium 2.5.4, you can run the MySQL connector against a single MariaDB database deployment. Both MySQL and MariaDB can use Global Transaction Identifiers (GTIDs) for replication. GTIDs help to uniquely identify transactions within a cluster.

For the MySQL connector to work with MariaDB GTIDs, you must apply a supplemental configuration to the connector. For more information, see the Debezium User Guide

#### 2.4.4.2. Streaming from a PostgreSQL 16 standby ( DBZ-7181)

With PostgreSQL 16, you can now define replication slots on a stand-by instance. Based on this change to the database, in Debezium you can now connect to a standby PostgreSQL 16 server, and stream changes from it. By performing change data capture from a replica, rather than from the production system, you can better distribute server load, particularly in a very active database.

### Previously available Technology Preview features

The following features that were introduced in earlier releases remain in Technology Preview:

#### 2.4.4.3. CloudEvents converter

The CloudEvwents converter emits change event records that conform to the CloudEvents specification. The CloudEvents change event envelope can be JSON or Avro and each envelope type supports JSON or Avro as the **data** format. For more information, see CloudEvents converter.

#### 2.4.4.4. Custom converters

In cases where the default data type conversions do not meet your needs, you can create custom converters to use with a connector. For more information, see Custom-developed converters.

#### 2.4.4.5. Use of the BLOB, CLOB, and NCLOB data types with the Oracle connector

For more information, see Oracle binary character LOB types in the Debezium User Guide.

#### 2.4.4.6. Parallel initial snapshots for SQL connectors

The ability to use multiple parallel threads to perform an initial snapshot is available as a Technology Preview feature for all of the Debezium SQL connectors, except MySQL. Parallel initial snapshots for the Debezium MySQL connector are available as a Developer Preview feature.

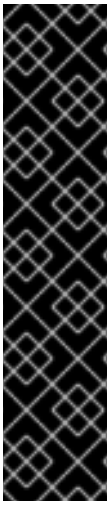
To configure connectors to use parallel initial snapshots, set the **snapshot.max.threads** property in the connector configuration to a value greater than **1**.

For more information, see **snapshot.max.threads** in the Debezium User Guide documentation for your connector.

### 2.4.5. Developer Preview features

ProductName} 2.5.4 includes the following Developer Preview features:name: value

- Section 2.4.5.1, "MySQL parallel schema snapshots (DBZ-6472)"
- Section 2.4.5.2, "MySQL parallel initial snapshots"
- Section 2.4.5.3, "Ingesting changes from a logical standby"
- Section 2.4.5.4, "Exactly-once delivery for PostgreSQL streaming"



**IMPORTANT**

Developer Preview features are not supported by Red Hat in any way and are not functionally complete or production-ready. Do not use Developer Preview software for production or business-critical workloads. Developer Preview software provides early access to upcoming product software in advance of its possible inclusion in a Red Hat product offering. Customers can use this software to test functionality and provide feedback during the development process. This software might not have any documentation, is subject to change or removal at any time, and has received limited testing. Red Hat might provide ways to submit feedback on Developer Preview software without an associated SLA.

For more information about the support scope of Red Hat Developer Preview software, see Developer Preview Support Scope.

#### 2.4.5.1. MySQL parallel schema snapshots (DBZ-6472)

To improve snapshot performance the MySQL connector, implements parallelization to concurrently snapshot change events and geneate schema events for tables

This reduces snapshot time when capturing the schema for many tables in your database.

After you configure this property, the connector uses the name pattern that is based on these key-value pairs when it gathers metrics. This establishes a stable name so that the connector always collects the same set of metrics, regardless of whether you modify the topic.prefix, or if the MBean name changes. customize the MBean object name for each connector instance. By adding these custom metric tags, you can ensure that each connector has a unique MBean name, preventing conflicts between multiple instances.

#### 2.4.5.2. MySQL parallel initial snapshots

The Debezium initial snapshot for MySQL has always been single-threaded. This limitation primarily stems from the complexities of ensuring data consistency across multiple transactions.

In this release, you can configure a MySQL connector to use multiple threads to execute table-level snapshots in parallel.

In order to take advantage of this new feature, add the **snapshot.max.threads** property to the connector configuration, and set the property to a value greater than **1**.

#### Example configuration using parallel snapshots

```
snapshot.max.threads=4
```

In the preceding example, if the connector needs to snapshot more than 4 tables, a maximum of 4 tables can be snapshot in parallel. When one thread finishes processing a table, it will find the next table in the queue and begin to perform a snapshot. The process continues until the connector finishes performing snapshots on all of the designated tables.

For more information, see **snapshot.max.threads** in the Debezium User Guide.

### 2.4.5.3. Ingesting changes from a logical standby

When the Debezium connector for Oracle connects to a production or primary database, it uses an internal flush table to manage the flush cycles of the Oracle Log Writer Buffer (LGWR) process. The flush process requires that the user account through which the connector accesses the database has permission to create and write to this flush table. However, logical stand-by databases often have restrictive data manipulation rules, and may even be read-only. As a result, writing to the database might not be feasible.

To support an Oracle read-only logical stand-by database, Debezium introduces a property to disable the creation and management of the flush table. You can use this feature with both Oracle Standalone and Oracle RAC installations.

To enable the Oracle connector to use a read-only logical stand-by, add the following connector option:

```
internal.log.mining.read.only=true
```

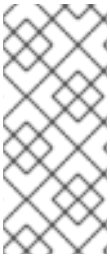
For more information, see the Oracle connector documentation in the Debezium User Guide.

### 2.4.5.4. Exactly-once delivery for PostgreSQL streaming

In this release, Debezium provides support for exactly-once semantics for the PostgreSQL connector. Exactly-once delivery for PostgreSQL applies only to the streaming phase; exactly-once delivery does not apply to snapshots.

Debezium was designed to provide at-least-once delivery with a goal of ensuring that connectors capture all change events that occur in the monitored sources. In KIP-618 the Apache Kafka community proposes a solution to address problems that occur when a producer retries a message. Source connectors sometimes resend batches of events to the Kafka broker, even after the broker previously committed the batch. This situation can result in duplicate events being sent to consumers (sink connectors), which can cause problems for consumers that do not handle duplicates easily.

No connector configuration changes are required to enable exactly-once delivery. However, exactly-once delivery must be configured in your Kafka Connect worker configuration. For information about setting the required Kafka Connect configuration properties, see KIP-618.



NOTE

To set **exactly.once.support** to **required** in the Kafka worker configuration, all connectors in the Kafka Connect cluster must supports exactly-once delivery, If you attempt to set this option in a cluster in which workers do not consistently support exactly-once delivery, the connectors that do not support this feature fail validation at start-up.

### 2.4.6. Other updates in this release

This Debezium 2.5.4 release provides multiple other feature updates and fixes, including the items in the following list:

- DBZ-3605 Add support for XML_TYPE column type to Debezium connector for Oracle (LogMiner)
- DBZ-3642 Debezium outbox not working with CloudEventsConverter
- DBZ-3925 MySQL connector fails to parse statement FLUSH FIREWALL_RULES
- DBZ-4321 Explore BLOB support via re-selection
- DBZ-5350 Oracle RAC throws ORA-00310: archive log sequence required
- DBZ-5359 Add the API endpoint to expose running connector metrics
- DBZ-5464 Snapshot result not saved if LAST record is filtered out
- DBZ-5518 Define and document schema history topic messages schema
- DBZ-5656 Oracle missing CDC data
- DBZ-5676 Align **query.fetch.size** across connectors
- DBZ-5750 Missing Oracle CDC records
- DBZ-6182 Don't require cluster-wide privileges when watching a single database/collection
- DBZ-6236 PostgreSQL connector doesn't restart properly if database if not reachable
- DBZ-6240 Provide by DDL type schema event filtering
- DBZ-6317 JDBC Sink Connector - Support batch operations
- DBZ-6416 Notify about initial snapshot progress
- DBZ-6417 Make signal actions extensible
- DBZ-6434 NullPointerException in MongoDB connector
- DBZ-6458 Only publish deltas instead of full snapshots to reduce size of sync event messages
- DBZ-6468 Set Readpreference tags in the MongoDB client
- DBZ-6472 MySqlSnapshotChangeEventSource parallel execute createSchemaEventsForTables

- DBZ-6481 PostgreSQL incremental snapshot fails on tables with an enum type in the primary key
- DBZ-6484 JDBC schema history: When the table name is passed as **dbName.tableName**, the connector does not start
- DBZ-6518 Update MongoDB incremental snapshot to allow multiple threads reading chunks
- DBZ-6521 MongoDB change stream pipeline not respecting hard coded **readPreference=secondaryPreferred**
- DBZ-6566 Support blocking ad-hoc snapshots
- DBZ-6567 SMT for handling timezone conversions
- DBZ-6577 Explain failure on existing publication update when switching to **filtered** from **all_tables**
- DBZ-6578 Debezium should honor read preference from connection string
- DBZ-6595 Use source field in topic in **table.format.name**
- DBZ-6602 Support for getting primary key from header
- DBZ-6603 Support for custom tags in the connector metrics
- DBZ-6615 Max transaction duration for Oracle connector
- DBZ-6617 Add shard field to events
- DBZ-6635 Debezium **heartbeat.action.query** does not start before writing to WAL
- DBZ-6636 Include only certain columns in JDBC sink connector
- DBZ-6637 Received an unexpected message type that does not have an 'after' Debezium block
- DBZ-6641 Schema name changed with custom topic naming strategy
- DBZ-6653 Add configurable timeout to initialization procedure
- DBZ-6654 CloudEvents converter is broken for JSON message deserialization
- DBZ-6658 Send tombstone events when partition queries are finished
- DBZ-6669 Snapshot does not capture data when **signal.data.collection** is present without **table.include.list**
- DBZ-6679 Log Mining Processor advances SCN incorrectly if LogMiner query returns no rows
- DBZ-6682 Wrong behavior of **quote.identifiers** in JdbcSinkConnector
- DBZ-6684 Propagate source column name and allow sink to use it
- DBZ-6685 Partition duplication after rebalances with single leader task
- DBZ-6686 JDBC sink connector fails on loading flat data containing Struct type fields from Kafka

- DBZ-6687 SQLSyntaxErrorException using Debezium JDBC sink connector
- DBZ-6689 Make the Kafka channel consumer group ID configurable for the PostgreSQL connector
- DBZ-6700 Missing **operationTime** field on ping command when executed against Atlas
- DBZ-6712 **schema.history.internal.store.only.captured.databases.ddl** flag not considered while snapshot schema to history topic
- DBZ-6714 Specify decimal precision in schema for MySQL unsigned BIGINTs in precise mode
- DBZ-6720 Toasted UUID array is not properly processed
- DBZ-6726 Utilize $changeStreamSplitLargeEvent to handle large change events with post- and pre-images
- DBZ-6727 Support alternative JDBC drivers in MySQL connector
- DBZ-6731 Blocking snapshot must take snapshot configurations from signal
- DBZ-6741 Support custom authentication on MongoDB connector
- DBZ-6742 Use JSON format for JMX Notification userData
- DBZ-6745 SingleProcessor remove redundant filter logic
- DBZ-6748 Debezium should convert _bin collate varchar columns to strings not byte arrays
- DBZ-6775 Table schemas should be updated for each shard individually
- DBZ-6778 Refactor ElapsedTimeStrategy
- DBZ-6782 Oracle XML column types are not properly resolved when adding XMLTYPE column during streaming
- DBZ-6786 Use custom RowDeserializers in case of binlog compression
- DBZ-6787 Incremental snapshot data-collections are not deduplicated
- DBZ-6793 Add timestamp to notification
- DBZ-6801 JDBC sink does not support SQL Server identity inserts
- DBZ-6811 SQL Server connector send heartbeats when there is no change in the database
- DBZ-6814 Make finished partition deletion delay configurable
- DBZ-6820 Fix bug with getsnapshottingtask
- DBZ-6828 Ad hoc blocking snapshot trigger emits schema changes of all tables
- DBZ-6831 Error with propagation source column name
- DBZ-6834 Provide INSERT/DELETE semantics for incremental snapshot watermarking
- DBZ-6843 When using **skip.messages.without.change=true** a WARN log message is reported for each record

- DBZ-6844 Support truncating large columns
- DBZ-6853 Kafka offset store fails with NPE
- DBZ-6855 JDBC Offset storage - configuration of table name does not work
- DBZ-6857 JDBC sink insert fails with Oracle target database due to semicolon
- DBZ-6862 Tombstone events causes NPE on JDBC connector
- DBZ-6864 MySQL connector not filtering AWS RDS internal events
- DBZ-6865 Avoid NPE when executing the arrived method in ExecuteSnapshot
- DBZ-6869 On initial Oracle connector start, if start_scn for a transaction is 0, log mining starts from oldest scn
- DBZ-6870 Ensure that the connector can handle rebalance events robustly
- DBZ-6871 ChangeStream aggregation pipeline fails on large documents which should be excluded
- DBZ-6878 Enhance Notification information and more notifications for Initial Snapshots
- DBZ-6893 Migrate all examples from mongodb.hosts to **mongodb.connection.string**
- DBZ-6899 Refactor Oracle streaming metrics beans
- DBZ-6907 Setting **none** to **delete.handle.mode** is recommended
- DBZ-6935 fix logger named
- DBZ-6940 Timezone transformation doesn't work
- DBZ-6941 MySQL Kafka signaling documentation is incorrect
- DBZ-6945 Drop events has wrong table changes information
- DBZ-6956 Infinite loop when using OR condition in additional-condition
- DBZ-6958 Wrong case-behavior for non-Avro column name in sink connector
- DBZ-6966 Filter out specified DDL events logic is reverted
- DBZ-6967 Handle properly bytea field for JDBC sink to PostgreSQL
- DBZ-6968 Documentation for **cursor.oversize.skip.threshold** is missing units
- DBZ-6970 Debezium JDBC sink process truncate event failure
- DBZ-6971 DDL parser does not support NOCOPY keyword
- DBZ-6973 Add MongoDB connector support for **filtering.match.mode=regex|literal** property
- DBZ-6974 Decrease time spent in handling rebalance events
- DBZ-6975 Single quote replication includes escaped quotes for N(CHAR/VARCHAR) columns

- DBZ-6982 Provide configuration option to exclude extension attributes from a CloudEvent
- DBZ-6983 Add the ability to sanitize field name when inferencing JSON schema
- DBZ-6990 Debezium JDBC sink should throw not supporting schema change topic exception
- DBZ-7015 Enable replication slot advance check
- DBZ-7016 Add configuration option to CloudEventsConverter to retrieve id and type from headers
- DBZ-7030 DDL statement couldn't be parsed
- DBZ-7035 Blocking ad-hoc snapshot is not really blocking for MySQL
- DBZ-7037 Fake ROTATE event on connection restart cleans metadata
- DBZ-7043 Emit a notification when completed reading from a capture instance
- DBZ-7050 Support snapshot with automatic retry
- DBZ-7058 Field exclusion does not work with events of removed fields
- DBZ-7060 Rename **metadata.location** to **metadata.source**
- DBZ-7065 JDBC sink connector not working with CloudEvent
- DBZ-7066 The DefaultDeleteHandlingStrategy couldn't add the rewrite "__deleted" field to a non-struct value
- DBZ-7067 Improve logging at DEBUG level for Commit events
- DBZ-7069 JDBC connection leak when error occurs during processing
- DBZ-7071 Replace schema tracking restriction for SYS/SYSTEM users with configurable option
- DBZ-7083 Implement strategy pattern for MariaDB and MySQL differences
- DBZ-7093 Check schema length when create value to find missed DDL by SQL_BIN_LOG=OFF
- DBZ-7095 MySQL parser does not conform to arithmetical operation priorities
- DBZ-7105 When RelationalBaseSourceConnector#validateConnection is called with invalid config [inside Connector#validate()] can lead to exceptions
- DBZ-7108 Switch default connection mode to shared for sharded clusters
- DBZ-7119 Debezium crashes on parsing MySQL DDL statement (specific INSERT)
- DBZ-7132 Failed to authenticate to the MySQL database after snapshot
- DBZ-7139 MongoDB data collection filter requires replica set specification on blocking/initial snapshot execution
- DBZ-7140 Debezium DDL parser crashes when parsing MySQL DDL statement (specific UNION)

- DBZ-7142 Outbox event router SMT throws NullPointerException when there is a whitespace in **fields.additional.placement** value
- DBZ-7146 Inactivity pause in MongoDB connector should be configurable
- DBZ-7152 Debezium DDL parser crashes when parsing MySQL DDL statement (specific UPDATE)
- DBZ-7157 JsonSerialisation is unable to process changes from sharded collections with composite sharding key
- DBZ-7159 Fail fast during deserialization if a value is not a CloudEvent
- DBZ-7162 Add last event process time, number of events, number of heartbeat events metrics to MongoDb connector
- DBZ-7164 Support persistent history for snapshot requests for the kafka signal topic
- DBZ-7177 Change metrics endpoint of Connect REST Extensions to use the MBeanServerv directly instead of HTTP calls to the Jolokia endpoint
- DBZ-7178 Metrics endpoint must handle connectors with multiple tasks (SQL Server)
- DBZ-7179 Fix DebeziumMySqlConnectorResource not using the new MySQL adapter structure to support different MySQL flavors
- DBZ-7181 Support logical decoding from Postgres 16 stand-bys
- DBZ-7183 Support MySQL 8 high resolution replication timestamps from GTID events
- DBZ-7184 Use buffer queue when reading MongoDB change stream events
- DBZ-7186 Cleanup event processing loop in streaming event source of MongoDB connector
- DBZ-7189 Parsing MySQL indexes for JSON field fails, when casting is used with types double and float
- DBZ-7191 JDBC connector wrongly uses default value when value is NULL on optional fields
- DBZ-7193 Unchanged toasted array columns are substituted with **unavailable.value.placeholder**, even when REPLICA IDENTITY FULL is configured
- DBZ-7194 Enable ability to stream changes against Oracle 23c for LogMiner
- DBZ-7196 Add modify range_partitions to modify_table_partition rule in parsing PL/SQL
- DBZ-7197 Add ability to avoid throwing an exception for missing additional fields
- DBZ-7206 MongoDB streaming pauses for blocking snapshot only when there is no event
- DBZ-7208 Handle Drop Tablespace in PL/SQL
- DBZ-7213 DDL GRANT statement couldn't be parsed
- DBZ-7216 When metadata is in headers, a schema name of a structure in CE **data** field is incorrect
- DBZ-7217 Add tracing logs to track execution time for Debezium JDBC connector

- DBZ-7218 Validate and clarify multiple archive log destination requirements for Oracle
- DBZ-7230 MySQL BIT Type should have a default length 1
- DBZ-7235 Add configuration option to CloudEventsConverter to customize schema type name
- DBZ-7236 Oracle abandoned transaction implementation bug causes memory error
- DBZ-7237 Oracle LOB to be properly ignored if **lob.enabled=false**
- DBZ-7242 Add grammar Oracle truncate cluster
- DBZ-7251 Length value is not removed when changing a column's type
- DBZ-7252 MongoDB collection snapshot notification contains incorrect offsets
- DBZ-7259 Debezium DDL parser crashes whebn parsing MySQL DDL statement (subquery with UNION)
- DBZ-7266 Oracle DDL parsing error in PARTITION REFERENCE
- DBZ-7268 Add PL/SQL Parser for Alter Table Memoptimize
- DBZ-7264 Error during snapshot with multiple snapshot threads does not properly abort snapshot
- DBZ-7271 MySQL RDS UPDATE queries not ignored
- DBZ-7272 Guard against implicit offset invalidation caused by switch of default connection mode
- DBZ-7275 Leaking JDBC connections
- DBZ-7277 Debezium MySQL could not parse certain grant privileges
- DBZ-7279 Add PL/SQL Parser for Create Table Memoptimize
- DBZ-7283 Support for Creating EDITIONABLE or NONEDITIONABLE packages
- DBZ-7299 MongoDb connector doesn't use post-images
- DBZ-7311 PostgreSQL ad-hoc blocking snapshots fails when snapshot mode is set to **never**
- DBZ-7312 Ad-hoc blocking snapshot dies with "invalid snapshot identifier" immediately after connector creation
- DBZ-7315 Specifying a table include list with spaces between elements cause LogMiner queries to miss matches
- DBZ-7316 Debezium **heartbeat.action.query** does not start before writing to WAL: part 2
- DBZ-7347 Initial snapshot notifications should use full identifier
- DBZ-7358 Re-select columns should use the table's primary instead of the event's key
- DBZ-7359 Full incremental snapshot on SQL Server Table skipping block of 36 records
- DBZ-7360 Debezium fails after table split operation

- DBZ-7374 SQL Server wrong default values in database schema for varchar, nvarchar, char columns
- DBZ-7379 Support connector scoped trustore/keystore for MongoDB
- DBZ-7420 ParsingException (MariaDB Only): alterSpec drop foreign key with 'tablename.' prefix
- DBZ-7421 Poor performance with incremental snapshot with long list of tables
- DBZ-7425 Oracle snapshot mistakenly uses LogMiner Offset Loader by default
- DBZ-7429 Reselect columns should source key values from after Struct when not using event-key sources
- DBZ-7431 Allow the C3P0ConnectionProvider to be customized via configuration
- DBZ-7436 Stopwatch throw NPE when toString is called without having statistics
- DBZ-7437 ReselectColumnsPostProcessor filter not use exclude predicate
- DBZ-7441 Ad hoc snapshots are not triggered via File channel signal when submitted before the start of the application
- DBZ-7445 LogMiner batch size does not increase automatically
- DBZ-7456 Oracle connector does not ignore reselection for excluded CLOB/BLOB columns
- DBZ-7460 The expected value pattern for **table.include.list** does not align with the documentation
- DBZ-7467 Signals actions are not loaded for SQL Server
- DBZ-7468 MySQL connector cannot parse table with WITH SYSTEM VERSIONING PARTITION BY SYSTEM_TIME
- DBZ-7479 PreparedStatement leak in Oracle ReselectColumnsProcessor
- DBZ-7488 Poor snapshot performance with new reselect post processor
- DBZ-7489 Debezium Oracle Connector ParsingException on XMLTYPE with **lob.enabled=true**
- DBZ-7500 Fix MySQL 8 event timestamp resolution logic error where fallback to seconds occurs erroneously for non-GTID events
- DBZ-7562 Numeric default value decimal scale mismatch
- DBZ-7567 Fix null event timestamp possible from FORMAT_DESCRIPTION and PREVIOUS_GTIDS events in MySqlStreamingChangeEventSource::setEventTimestamp
- DBZ-7581 Improved logging in case of PostgreSQL failure
- DBZ-7582 Unavailable Toasted HSTORE Json Storage Mode column causes serialization failure
- DBZ-7593 Deploying Debezium for the first time, it fails to capture the schema of all tables in the database
- DBZ-7594 Incorrect value of TIME(n) replicate from MySQL if the original value is negative

- DBZ-7596 Re-select post processor not working for complex types
- DBZ-7599 Null instead of toast placeholder written for binary types when "hex" mode configured
- DBZ-7608 Poor snapshot performance during schema snapshot DDL processing
- DBZ-7617 Incremental snapshot query doesn't honor **message.key.columns** order
- DBZ-7619 Metric ScnFreezeCount never increases
- DBZ-7643 Numeric value without mantissa cannot be parsed
- DBZ-7666 MySQL connector fails to parse DDL with RETURNING keyword
- DBZ-7690 Schema history comparator doesn't handle SERVER_ID_KEY and TIMESTAMP_KEY properly

## 2.5. DEPRECATED FEATURES

The following features are deprecated and will be removed in a future release:

- Oracle 12 support

## 2.6. KNOWN ISSUES

The following known issue affects Debezium 2.5.4:

- Apicurio registry 2.4.3 and 2.4.4 causes endless rebalance loop on Kafka