



# Hibernate Search 7.0.1.Final

## *Migration Guide from 6.2*

2024-04-10

# Table of Contents

|                                      |    |
|--------------------------------------|----|
| Introduction .....                   | 1  |
| Requirements .....                   | 2  |
| Artifact changes .....               | 3  |
| Data format and schema changes ..... | 4  |
| Indexes .....                        | 4  |
| Outbox polling system tables .....   | 4  |
| Configuration changes .....          | 8  |
| API changes .....                    | 9  |
| SPI changes .....                    | 10 |
| Behavior changes .....               | 11 |

# Introduction

The aim of this guide is to assist you migrating an existing application using any version **6.2.x** of Hibernate Search to the latest of the **7.0.x** series.



If you think something is missing or something does not work, please [contact us](#).

If you're looking to migrate from an earlier version, you should migrate step-by-step, from one minor version to the next, following the migration guide of [each version](#).



## To Hibernate Search 5 users

Be aware that a lot of APIs have changed since Hibernate Search 5, some only because of a package change, others because of more fundamental changes (like moving away from using Lucene types in Hibernate Search APIs).

When migrating from Hibernate Search 5, you are encouraged to migrate first to Hibernate Search 6.0 using the [6.0 migration guide](#), and only then to later versions (which will be significantly easier).

# Requirements

Hibernate Search 7.0.1.Final now requires:

- JDK 11 or later;
- Lucene 9 for its Lucene backend;
- Elasticsearch 7.10+ or OpenSearch 1.3+ for its Elasticsearch backend;
- Hibernate ORM 6.4.x for the Hibernate ORM integration.

Regarding the Hibernate ORM requirements, when migrating from previous versions of Hibernate Search:

*If your application uses Hibernate ORM 6.3*

Your Hibernate Search dependencies include references to artifacts ending with `-orm6` (e.g. `hibernate-search-mapper-orm-orm6`).

Just remove the `-orm6` suffix from the dependency coordinates (e.g. use `hibernate-search-mapper-orm` instead) when you upgrade to this version of Hibernate Search.

Then, [migrate to Hibernate ORM 6.4](#) at the same time you upgrade to this version of Hibernate Search.

*If your application uses Hibernate ORM 6.2*

You need to [migrate to Hibernate ORM 6.3](#) first before upgrading to this version of Hibernate Search.

Then follow the instructions for "If your application uses Hibernate ORM 6.3" above.

*If your application uses Hibernate ORM 5.6*

You need to migrate to Hibernate ORM 6.3 first before upgrading to this version of Hibernate Search; see [Hibernate ORM migration guides](#) and [this guide for using Hibernate ORM 6.x with Hibernate Search 6.2](#).

Then follow the instructions for "If your application uses Hibernate ORM 6.3" above.

# Artifact changes

With the move to Jakarta EE Hibernate Search previous batch artifacts (`hibernate-search-mapper-orm-batch-jsr352-core`/`hibernate-search-mapper-orm-batch-jsr352-jberet`) will now relocate to the new ones (`hibernate-search-mapper-orm-jakarta-batch-core`/`hibernate-search-mapper-orm-jakarta-batch-jberet`) matching the naming of Jakarta EE specification. This also means that the corresponding Java module names and packages are also updated:

- The core module is now named `org.hibernate.search.jakarta.batch.core` and the base package for this module is `org.hibernate.search.jakarta.batch.core`
- The JBeret module is now named `org.hibernate.search.jakarta.batch.jberet` and the base package for this module is `org.hibernate.search.jakarta.batch.jberet`

Outbox polling for Hibernate ORM artifact is getting a shorter name: `hibernate-search-mapper-orm-outbox-polling`. For most of the users this should only be the change in the dependencies, unless some setting keys from `HibernateOrmMapperOutboxPollingSettings` where referenced in the code, then an import update will be required as well. The corresponding Java module name and base package are updated to:

- The module is now named `org.hibernate.search.mapper.orm.outboxpolling` and the base package for this module is `org.hibernate.search.mapper.orm.outboxpolling`

# Data format and schema changes

## Indexes

Elasticsearch indexes created with Hibernate Search 6.2 can be read from and written to with Hibernate Search 7.0.1.Final.

Reading and writing to Lucene indexes created with Hibernate Search 6.2 using Hibernate Search 7.0.1.Final may lead to exceptions, since there were incompatible changes applied to internal fields. You must recreate your Lucene indexes and reindex your database. The easiest way to do so is to use [the MassIndexer](#) with `dropAndCreateSchemaOnStart(true)`.

## Outbox polling system tables

If you use the incubating [outbox-polling coordination strategy](#), with the PostgreSQL database you will be impacted by the changes to entities that represents the outbox event and agent, requiring database schema changes. The `payload` column changes its type from `oid` to `bytea` for both outbox event and agent tables. You can find suggested migration scripts below:

*Postgresql:*

```
-- Change outbox event `payload` column type to bytea:
-- Note the way existing LOBs are retrieved using lo_get function.
alter table hsearch_outbox_event
  alter column payload type bytea using lo_get(payload);

-- Change agent `payload` column type to bytea:
-- Note: even though agent table should've not used payload column so far, we still need to
-- have a type cast.
-- It can be done either with the same `lo_get()` function call, or with a pair of casts like
-- `cast( cast( payload as text ) as bytea )`:
alter table hsearch_agent
  alter column payload type bytea using lo_get(payload);
```

Other databases:

- CockroachDB: no migration required. Type of the `payload` is `bytes` in both cases.
- MySQL: no migration required. Type of the `payload` is `longblob` in both cases.
- MariaDB: no migration required. Type of the `payload` is `longblob` in both cases.
- DB2: no migration required. Type of the `payload` is `blob` in both cases.
- Oracle: no migration required. Type of the `payload` is `blob` in both cases.
- MSSQL: no migration required. Type of the `payload` is `varbinary(max)` in both cases.
- H2: no migration required. Type of the `payload` is `blob` in both cases.

If you were using Hibernate Search 6.2 with Hibernate ORM 5, i.e. using regular Hibernate Search artifacts and not `-orm6/-jakarta` ones this upgrade will also mean the upgrade of Hibernate ORM to 6.3. Doing so will lead to a potential type mismatch when using Hibernate ORM's schema validation. To prevent that, `id` column types can be updated from `varchar` to `char` where applicable. You can find

suggested migration scripts for the tested databases below:

#### Postgresql:

```
-- change outbox event `id` column type to char:
alter table hsearch_outbox_event
  alter column id TYPE char(36);

-- change agent `id` column type to char:
alter table hsearch_agent
  alter column id TYPE char(36);
```

#### CockroachDB:

```
-- change outbox event `id` column type to char:
-- altering type directly is not supported: https://go.crdb.dev/issue-v/47636/v22.1
alter table hsearch_outbox_event
  add tmp char(36);
update hsearch_outbox_event
set tmp = id
where 1 = 1;
alter table hsearch_outbox_event
  alter column tmp set not null;
alter table hsearch_outbox_event
  alter primary key using columns (tmp);
alter table hsearch_outbox_event
  drop column id;
alter table hsearch_outbox_event
  rename column tmp to id;

-- change agent `id` column type to char:
alter table hsearch_agent
  add tmp char(36);
update hsearch_agent
set tmp = id
where 1 = 1;
alter table hsearch_agent
  alter column tmp set not null;
alter table hsearch_agent
  alter primary key using columns (tmp);
alter table hsearch_agent
  drop column id;
alter table hsearch_agent
  rename column tmp to id;
```

#### MySQL:

```
-- change outbox event `id` column type to char:
alter table hsearch_outbox_event
  modify column id char(36);

-- change agent `id` column type to char:
alter table hsearch_agent
  modify column id char(36);
```

#### MariaDB:

```
-- change outbox event `id` column type to char:
alter table hsearch_outbox_event
  modify column id char(36);

-- change agent `id` column type to char:
```

```
alter table hsearch_agent
  modify column id char(36);
```

#### DB2:

```
-- change outbox event `id` column type to char:
alter table hsearch_outbox_event
  drop primary key;
alter table hsearch_outbox_event
  alter column id set data type char(36);
-- make this call if the adding constraint fails:
call sysproc.admin_cmd('reorg table hsearch_outbox_event');
alter table hsearch_outbox_event
  add constraint hsearch_outbox_event_pkey primary key (id);

-- change agent `id` column type to char:
alter table hsearch_agent
  drop primary key;
alter table hsearch_agent
  alter column id set data type char(36);
-- make this call if the adding constraint fails:
call sysproc.admin_cmd('reorg table hsearch_agent');
alter table hsearch_agent
  add constraint hsearch_agent_pkey primary key (id);
```

#### Oracle:

```
-- change outbox event `id` column type to char:
alter table hsearch_outbox_event
  add tmp char(36);
update hsearch_outbox_event
  set tmp = id
  where 1 = 1;
alter table hsearch_outbox_event
  modify tmp not null;
alter table hsearch_outbox_event
  drop column id;
alter table hsearch_outbox_event
  rename column tmp to id;
alter table hsearch_outbox_event
  add constraint hsearch_outbox_event_pkey primary key (id);

-- change agent `id` column type to char:
alter table hsearch_agent
  add tmp char(36);
update hsearch_agent
  set tmp = id
  where 1 = 1;
alter table hsearch_agent
  modify tmp not null;
alter table hsearch_agent
  drop column id;
alter table hsearch_agent
  rename column tmp to id;
alter table hsearch_agent
  add constraint hsearch_agent_pkey primary key (id);
```

#### MSSQL:

```
-- change publox event `id` column type to char:
alter table hsearch_outbox_event
  drop constraint if exists hsearch_outbox_event_pkey;
alter table hsearch_outbox_event
```



```
alter column id binary(16) not null;
alter table hsearch_outbox_event
  add constraint hsearch_outbox_event_pkey primary key (id);

-- change agent `id` column type to char:
alter table hsearch_agent
  drop constraint if exists hsearch_agent_pkey;
alter table hsearch_agent
  alter column id binary(16) not null;
alter table hsearch_agent
  add constraint hsearch_agent_pkey primary key (id);
```

H2:

```
-- change outbox event `id` column type to char:
alter table hsearch_outbox_event
  alter column id char(36) not null;

-- change agent `id` column type to char:
alter table hsearch_agent
  alter column id char(36) not null;
```

# Configuration changes

The configuration properties are backward-compatible with Hibernate Search 6.2.

However, some configuration values are deprecated:

- `hibernate.search.coordination.entity.mapping.outboxevent.uuid_type` and `hibernate.search.coordination.entity.mapping.agent.uuid_type` now accept names of SQL type codes from `org.hibernate.type.SqlTypes` or their corresponding int values. The value `default` is still valid. `uuid-binary` and `uuid-char` are accepted and converted to their corresponding `org.hibernate.type.SqlTypes` alternatives, but they are deprecated and will not be accepted in the future versions of Hibernate Search.

# API changes

The [API](#) is for the most part backward-compatible with Hibernate Search 6.2.

However, some APIs changed:

- The complement operator (`~`) used for [matching regular expression patterns with flags](#) is now removed with no alternative to replace it.
- The Hibernate Search job for Jakarta Batch no longer accepts a `customQueryHQL` / `.restrictedBy(String)` parameter. Use `.reindexOnly(String hql, Map parameters)` instead.
- The Hibernate Search job for Jakarta Batch no longer accepts a `sessionClearInterval` / `.sessionClearInterval(int)` parameter. Use `entityFetchSize` / `.entityFetchSize(int)` instead.

# SPI changes

The [SPI](#) are for the most part backward-compatible with Hibernate Search 6.2.

# Behavior changes

The behavior of Hibernate Search 7.0.1.Final is for the most part backward-compatible with Hibernate Search 6.2.

However, parts of Hibernate Search now behave differently:

- The default value for `hibernate.search.backend.query.shard_failure.ignore` is changed from `true` to `false` which means that now Hibernate Search will throw an exception if at least one shard failed during a search operation. To get the previous behavior set this configuration property explicitly to `true`. Note, this setting must be set for each elasticsearch backend, if multiple are defined.
- The Hibernate Search job for Jakarta Batch will now list identifiers in one session (with one DB connection), while loading entities in another (with another DB connection). This is to sidestep limitations of scrolling in some JDBC drivers.
- For entities whose document ID is based on a different property than the entity ID, the Hibernate Search job for Jakarta Batch will now build the partition plan using that property instead of using the entity ID indiscriminately.