# Hibernate Cheat Sheet

Hibernate ORM 7.4.x / Jakarta Persistence 3.2 -- quick reference. AI-generated; verify against the official documentation. See the Hibernate Reference section for full explanations and every option.

## Entity lifecycle

| State | Meaning |
|---|---|
| Transient | `new Foo()`, no identity |
| Managed | in persistence context, tracked |
| Detached | had identity, context closed |
| Removed | scheduled for DELETE |

```
persist -> managed; remove -> removed;
detach/clear/close -> detached; merge(x) returns
a NEW managed copy (x stays detached)
```

## @GeneratedValue strategies

| Strategy | Notes |
|---|---|
| IDENTITY | auto-increment; disables batching |
| SEQUENCE | preferred; keeps batching intact |
| TABLE | portable, slowest |
| UUID | client-side, no round-trip |

**Composite keys**

```
@EmbeddedId (preferred) / @IdClass
@NaturalId -- business key, not PK
@MapsId -- shared-PK @OneToOne
```

## Associations

| Annotation | Owning? |
|---|---|
| @ManyToOne | always owning |
| @OneToMany(mappedBy) | inverse |
| @OneToOne | owner has @JoinColumn |
| @ManyToMany | owner has @JoinTable |

```
cascade=ALL, orphanRemoval=true -> true
parent/child composition
@OnDelete(CASCADE) -> DB-level FK cascade
```

## Inheritance strategies

| Strategy | Trade-off |
|---|---|
| SINGLE_TABLE | 1 table, nullable cols, no join (default) |
| JOINED | normalized, 1 join/level |
| TABLE_PER_CLASS | repeated cols, UNION ALL |

```
@MappedSuperclass -- shared mapping,
no polymorphic query
```

## Collections

- `List` with no @OrderColumn = bag (unordered, dup issues)
- 2 bags in one JOIN FETCH -> MultipleBagFetchException; use Set
- @ElementCollection -- values, no identity, fully owned
- @OrderColumn = stored index; @OrderBy = DB sort at read

## Locking modes

| Mode | Use |
|---|---|
| @Version OPTIMISTIC | version column check |
| OPTIMISTIC_FORCE_INCREMENT | force bump on assoc. change |
| PESSIMISTIC_READ | SELECT ... FOR SHARE |
| PESSIMISTIC_WRITE | SELECT ... FOR UPDATE |
| PESSIMISTIC_FORCE_INCREMENT | write lock + version bump |

```
jakarta.persistence.lock.timeout (ms)
```

## Flushing

| FlushMode | Behavior |
|---|---|
| AUTO | flush before affected query + commit |
| COMMIT | flush only at commit |
| ALWAYS | flush before every query |
| MANUAL | explicit flush() only |

```
@DynamicUpdate / @DynamicInsert -- only
changed / non-null columns
```

## Fetching & N+1 fixes

- Default every association to `FetchType.LAZY`
- `JOIN FETCH` -- 1 query, one to-many max
- @EntityGraph -- reusable fetch plan
- @BatchSize / default_batch_fetch_size -- IN-batches
- @Fetch(SELECT|JOIN|SUBSELECT)
- SELECT new ... DTO -- avoid entities entirely

## Query language decision guide

| Need | Use |
|---|---|
| Static query, concise | HQL/JPQL |
| Dynamic filters, refactor-safe | Criteria API (+ metamodel) |
| Vendor-specific SQL, window fns | Native SQL |
| Existing DB logic | Stored procedures |

```
SELECT new pkg.Dto(...) FROM ... -- DTO projection
```

## Pagination

```
setFirstResult(offset).setMaxResults(n)
key-based: WHERE id > :lastSeenId ORDER BY id
```

```
@NamedQuery -- validated/prepared at bootstrap
```

## Bulk & batching

- `hibernate.jdbc.batch_size` + order_inserts/updates
- StatelessSession -- no context, no dirty check, ETL
- Bulk UPDATE/DELETE/INSERT..SELECT bypasses persistence context + 2nd-level cache
- flush()/clear() loop for large imports

## Caching layers

| Strategy | Use |
|---|---|
| READ_ONLY | never updated |
| NONSTRICT_READ_WRITE | tolerable brief staleness |
| READ_WRITE | updated regularly (default pick) |
| TRANSACTIONAL | JTA + capable provider |

```
Providers: JCache+Caffeine, Infinispan,
Ehcache, Redis (adapter)
```

## Events & filters

- @PrePersist/@PostPersist/@PreUpdate/@PostUpdate/@PreRemove/@PostRemove/@PostLoad
- Interceptor -- session-wide hook, every entity
- @FilterDef/@Filter -- per-session toggle WHERE clause
- @SoftDelete -- remove() becomes UPDATE, auto-excluded from queries

## Schema generation

| hbm2ddl.auto | Use |
|---|---|
| none | no action |
| validate | prod -- fail fast on drift |
| update | dev only, never drops |
| create-drop | local/test only |

```
Migrations own prod schema + ddl-auto=validate
```

## Envers auditing

- @Audited -- entity/field/association history
- _AUD shadow tables + REVINFO (rev/timestamp)
- AuditReader.find(Class, id, rev)
- ValidityAuditStrategy -- adds REVEND, faster point-in-time queries

## Multitenancy

| Strategy | Isolation / cost |
|---|---|
| Separate DB | strongest / highest cost |
| Separate schema | strong / shared infra |
| Discriminator (@TenantId) | weakest / cheapest |

```
CurrentTenantIdentifierResolver +
MultiTenantConnectionProvider
```

## Performance essentials

- LAZY by default; open-in-view=false
- SEQUENCE over IDENTITY for batching
- Statistics API + hibernate.generate_statistics
- LOG_QUERIES_SLOWER_THAN_MS -- slow query log
- Anti-patterns: OSIV on, eager everywhere, entity-not-DTO, @OneToMany w/o @BatchSize

## Hibernate Search annotations

| | |
|---|---|
| @Indexed | root of index |
| @FullTextField | analyzed, tokenized |
| @KeywordField | exact match |
| @GenericField | numbers/dates/bool |
| @IndexedEmbedded | pull assoc. fields in |

```
MassIndexer -- full (re)index from DB
```

## Search backend choice

| Backend | Fit |
|---|---|
| Lucene (embedded) | single node only -- does NOT scale out |
| Elasticsearch/OpenSearch | multi-instance, shared cluster (recommended) |

```
coordination.strategy: none | outbox-polling
(multi-node reliable)
```

## Hibernate Reactive one-liners

- Mutiny.SessionFactory -- unwrap from EntityManagerFactory
- withSession/withTransaction(Uni-returning callback)
- Bound to Vert.x **Context** -- NOT Reactor Context, NOT ThreadLocal
- Needs a Vert.x reactive SQL client, no JDBC
- No Spring Data support; no ReactiveTransactionManager
- mutiny-reactor: UniReactorConverters.toMono() / MultiReactorConverters.toFlux()
- Hibernate Search does NOT support Hibernate Reactive (HSEARCH-4922)

## Data Repositories & Validator

```
@Repository + @Find/@Query/@HQL/@SQL --
Jakarta Data, blocking, StatelessSession-based
```

```
@Valid cascades; @Validated = Spring
method-level validation
```

```
jakarta.persistence.validation.mode = AUTO
-- pre-persist/pre-update checks
```

## Key settings

| | |
|---|---|
| hibernate.hbm2ddl.auto | schema mode |
| hibernate.show_sql/format_sql | SQL logging |
| hibernate.jdbc.batch_size | JDBC batching |
| hibernate.default_batch_fetch_size | N+1 batching |
| hibernate.generate_statistics | Statistics API |