Jync
捷同 · 异构数据库同步

DEPLOYMENT & OPERATIONS

部署与运维手册

从安装、配置到性能调优与故障排查
让同步平台在生产环境稳定运行

JAR / Docker / K8s
全量调优
CDC 运维
备份与升级

v1.12.0

版本 v1.12.0 · 2026 年 11 月 · 适用于 Jync 1.12.x

目录

文档名称	Jync 部署与运维手册	文档版本	v1.12.0
适用产品	Jync 捷同 1.12.x	更新日期	2026 年 11 月
读者对象	系统管理员、DBA、运维工程师	官网	jync.qqmu.com

1 部署规划 6

1.1 环境要求 6

1.2 部署形态选择 6

1.3 网络、端口与时钟 7

1.4 账号与权限清单 7

1.5 上线前检查清单 8

2 安装部署 9

2.1 裸机部署与 systemd 9

2.2 Docker 与 Compose 11

2.3 Kubernetes 12

2.4 反向代理与 TLS 13

2.5 首次登录与初始化 14

3 配置参考 17

3.1 配置来源与优先级 17

3.2 服务与元数据库 17

3.3 同步与全量 18

3.4 增量与 CDC 19

3.5 安全、认证、AI 与更新检查 20

3.6 日志与 JVM 21

3.7 性能采样与指标 22

4 元数据、备份与多实例 24

4.1 元数据库里有什么 24
4.2 H2 与外部 MySQL 25
4.3 密钥文件 26
4.4 备份与恢复 26
4.5 多实例部署 27
5 全量性能调优 29
5.1 先理解吞吐构成 29
5.2 全量参数矩阵 30
5.3 连接池与并行度 30
5.4 Oracle 专项优化 31
5.5 一次完整的调优过程 32
5.6 常见反模式 33
6 日志 CDC 运维 34
6.1 日志 CDC 引擎与位点 34
6.2 源库准备 SQL 35
6.3 启动与位点纪律 37
6.4 状态面板与关键指标 38
6.5 延迟与积压处理 39
6.6 断线、重连与租约接管 40
6.7 重建基线与策略切换 40
7 轮询同步运维 41
7.1 五种游标策略 41
7.2 游标列的选择与维护 41
7.3 轮询间隔与源库负载 42
7.4 安全滞后与漏数判定 42
7.5 何时切换到日志 CDC 43
8 ETL 任务与批次运维 44
8.1 ETL 与同步任务的分工 44

8.2 调度与线程池 45
8.3 批次：共享连接与顺序执行 45
8.4 断点续跑与检查点 46
8.5 运行历史与故障排查 47
9 监控、日志与告警 49
9.1 日志文件与轮转 49
9.2 关键日志与判读 49
9.3 指标采集与探活 50
9.4 告警建议 52
9.5 Webhook 告警运维 53
9.6 审计与留痕 56
9.7 系统日志页面 56
9.8 诊断包与支持取证 56
9.9 内容级校验的运维开销 57
10 专属迁移运维 59
10.1 组件与数据模型 59
10.2 连接与凭据 59
10.3 调度、运行与恢复 59
10.4 监控与排障 60
10.5 限制与升级 60
11 升级与回滚 61
11.1 版本检查与一键升级 61
11.2 离线手工升级 61
11.3 回滚 62
11.4 升级检查清单 62
11.5 兼容性注意事项 63
12 安全加固 66
12.1 账号与口令 66

12.2 传输与 Cookie 66
12.3 密钥管理 66
12.4 数据库最小权限 66
12.5 网络隔离与数据出网 67
12.6 外部认证与单点登录 67
13 故障排查 69
13.1 启动与元数据 69
13.2 连接与权限 69
13.3 结构与全量 70
13.4 增量与 CDC 70
13.5 性能与资源 71
13.6 页面、升级与升级后异常 72
14 附录 73
14.1 参数速查 73
14.2 环境变量速查 75
14.3 运维检查清单 76
14.4 术语表与支持 76

CHAPTER 01

部署规划

Jync 的部署很轻：一个 JAR、一个元数据目录、一个 Web 端口。真正需要提前想清楚的是它要访问哪些数据库、这些数据库允许多大的负载、以及谁来负责日常运维。本章把这些问题在动手安装之前定下来。

1.1 环境要求

类别	最低要求	建议
操作系统	Linux x86_64 / ARM64	主流企业 Linux，内核 3.10+
JDK	17	17 LTS 或 21 LTS，使用发行版 OpenJDK
CPU	2 核	4 核以上，全量并行度大于 2 时按并行度增加
内存	2 GB（堆 512 MB）	4 GB 以上，堆不超过物理内存的 75%
磁盘	1 GB	按元数据与日志增长预留；日志已有 500 MB 总量上限
浏览器	Chrome / Edge 近两年版本	控制台只做管理操作，不需要独立客户端

JDK 版本是硬要求

Jync 使用 Spring Boot 2.7 与 JDK 17 编译。用 JDK 8 或 11 启动会直接失败；用更高版本 JDK 通常可以运行，但生产环境建议与发布验证版本保持一致。

1.2 部署形态选择

形态	适用场景	元数据	多副本
JAR + systemd	单机内网、POC、小规模生产	本地 H2 文件	否
Docker 单机	标准化交付、快速替换	数据卷中的 H2 文件	否
Docker Compose	单机多组件编排	宿主机目录	否
Kubernetes	已有集群、统一运维	PVC 中的 H2，或外部元数据库	单副本；外部元数据库后可多副本
多实例	高可用、跨主机接管	外部元数据库（必需）	是

先单实例，再谈多实例

单实例加定期备份已经能满足绝大多数企业的可用性要求。多实例的价值在于节点故障后的自动接管，代价是必须维护一套外部元数据库，并统一管理加密密钥。

1.3 网络、端口与时钟

方向	端口 / 协议	用途	说明
用户 → Jync	8080/TCP（可改）	Web 控制台	生产建议置于反向代理之后，只暴露 443
Jync → 源库	各数据库端口	读取数据与元数据	需放通到所有源库实例
Jync → 目标库	各数据库端口	建表与写入	需放通到所有目标库实例
Jync → 外网	443/TCP	版本检查、一键升级	纯内网可关闭；关闭后功能自动隐藏
Jync → AI 端点	按供应商	AI 起草	默认不发起请求，只有配置并启用后才出网
Jync → 企业 OA / 单点	接口端口（通常 443）	外部认证校验	仅在启用外部认证后发生，只应指向可信内网地址
节点 → 节点	元数据库端口	多实例共享状态	节点之间不直接通信，全部经元数据库

不要使用容器内 localhost

容器里的 `localhost` 指向容器自身。源库或目标库在宿主机上时，应填写宿主机实际 IP；Docker Desktop 环境可用 `host.docker.internal`。

多节点部署必须保证 NTP 同步。CDC 位点与租约以数据库时间为准，但「落后秒数」、日志时间戳与接管判断都依赖节点时钟；偏差过大会让监控数字失真，也会延长故障接管窗口。

1.4 账号与权限清单

账号	归属	权限	用途
操作系统账号	Jync 服务器	仅运行服务与读写安装目录	专用账号，不使用 root
admin	Jync 控制台	管理员	配置、启停、升级；初始密码 123456
view	Jync 控制台	只读查看	值班与审计查看；初始密码 123456
外部认证账号	企业 OA / 统一门户	由门户授予；Jync 侧命中管理员清单或标识字段者为管理员	仅在启用外部认证时存在；不在 Jync 建本地密码

账号	归属	权限	用途
源库只读账号	每个源库	SELECT、读取元数据	轮询模式只需只读权限
源库 CDC 账号	每个 CDC 源库	在只读基础上增加复制/挖掘权限	见 6.2 各引擎准备 SQL
目标库写账号	每个目标库	建表、索引、DML	结构同步与数据写入

账号分开建

同一个数据库既是源又是目标时，建议分别建只读账号与写账号，按同步任务绑定。这样从权限层面就能阻止误把目标库当源库读取，也便于审计。

源库账号与删表联动

开启「允许执行删除」时，源库删表要联动到目标端，需要源库账号能看见整个 schema：表属主身份，或库级 `SELECT` / Oracle 的 `SELECT ANY TABLE` / SQL Server 的 `VIEW DEFINITION`（PostgreSQL 系默认可见）。只授予部分表 `SELECT` 的账号下，被收回授权的表与真删表无法区分，Jync 会保留目标表并在日志中告警一次。

1.5 上线前检查清单

- ✓ JDK 17 已安装，`java -version` 输出符合要求。
- ✓ 安装目录、数据目录、日志目录已规划，磁盘剩余空间大于 5 GB。
- ✓ 服务器与所有源库、目标库网络互通，端口逐一验证。
- ✓ NTP 已配置；多节点部署时各节点时间偏差在秒级以内。
- ✓ 源库只读账号与目标库写账号已创建，并在客户端实际登录验证。
- ✓ CDC 同步任务对应的日志能力已开启，日志保留时间不少于 24 小时。
- ✓ 确定元数据形态：本地 H2 还是外部 MySQL 8 / PostgreSQL 16 / Oracle 23；确定备份责任人。
- ✓ 如启用外部认证：认证方式、OA 接口地址或网关身份头、管理员清单已与门户团队确认；可信代理列表已规划。
- ✓ 首次全量的时间窗口已与业务方确认，避开高峰。
- ✓ 控制台访问路径已确定，HTTPS 证书与反向代理配置就绪。
- ✓ 管理员初始密码更换计划已确定；如关闭本地登录，应急恢复流程已写入运维手册。
- ✓ 如需把运行异常推到企业 IM，Webhook 地址已确认（只使用可信内网或官方机器人地址）；不需要时保持告警关闭。

CHAPTER 02

安装部署

本章给出四种落地方式：裸机 systemd、Docker、Kubernetes，以及反向代理。它们运行的是同一个 JAR，差别只在进程管理、持久化与网络暴露方式。任选一种即可，不必全部实施。

2.1 裸机部署与 systemd

1. **创建专用账号与目录**
   使用独立的系统账号运行，安装目录固定，避免相对路径漂移。

2. **放置 JAR 与配置**
   发行包统一命名为 `jync.jar`，版本只体现在发布标签上，升级时同名替换。

3. **编写 systemd 单元**
   固定工作目录，配置自动重启与优雅停机时间。

4. **启动并验证**
   访问 `/login`，再用管理员账号登录。

```
# 1) 账号与目录
useradd -r -s /sbin/nologin jync
mkdir -p /opt/jync /opt/jync/data /opt/jync/logs
chown -R jync:jync /opt/jync

# 2) 放置文件
cp jync.jar /opt/jync/jync.jar
cp application.yml /opt/jync/application.yml
chown jync:jync /opt/jync/jync.jar /opt/jync/application.yml
```

```
# /etc/systemd/system/jync.service
[Unit]
Description=Jync 捷同 异构数据库同步平台
After=network-online.target
Wants=network-online.target

[Service]
Type=simple
User=jync
Group=jync
WorkingDirectory=/opt/jync
ExecStart=/usr/bin/java -Xms512m -Xmx2g -XX:+UseG1GC \
  -jar /opt/jync/jync.jar \
  --spring.config.additional-location=file:/opt/jync/application.yml
SuccessExitStatus=143
Restart=always
RestartSec=10
LimitNOFILE=65536

[Install]
WantedBy=multi-user.target
```

```
# 3) 启用与启动
systemctl daemon-reload
systemctl enable --now jync
systemctl status jync

# 4) 验证
curl -fsS -o /dev/null -w "%{http_code}\n" http://127.0.0.1:8080/login
tail -f /opt/jync/logs/jync.log
```

必须使用 additional-location

覆盖配置只能用 `--spring.config.additional-location`。使用替换式的 `--spring.config.location` 会丢掉 JAR 内置配置，典型症状是登录页出现 `??login.title_zh_CN??`、全新部署不建表，甚至启动失败。

最小配置示例

```
# /opt/jync/application.yml —— 只写需要覆盖的项，其余沿用 JAR 内置默认值
server:
  port: 8080

spring:
  datasource:
    url: jdbc:h2:file:./data/jync;MODE=MySQL;AUTO_SERVER=TRUE

sync:
  full-load-parallelism: 2
  max-pool-size: 5
  # 纯内网部署：关闭 AI，应用不再发起任何外网请求
  ai:
    enabled: false

logging:
  file:
    name: ./logs/jync.log

# 纯内网部署：关闭版本检查，一键升级入口同步隐藏
app:
  update-check:
    enabled: false
```

> **只写需要覆盖的项**
> 外部配置文件会与 JAR 内置配置合并，未出现的键沿用默认值。不要把内置配置整份复制出来再改，否则升级后新增的默认值会被旧文件盖住。

2.2 Docker 与 Compose

仓库自带多阶段 `Dockerfile`：构建阶段用 Maven 打包，运行阶段只保留 JRE，并以非 root 用户 `jync`（uid 10001）运行。容器内工作目录固定为 `/app`，数据与日志分别是 `/app/data` 与 `/app/logs`。

```
# 构建镜像
docker build -t jync:1.8.2 .

# 单机运行
docker run -d --name jync \
  -p 8080:8080 \
  -v jync-data:/app/data \
  -v jync-logs:/app/logs \
  -e TZ=Asia/Shanghai \
  --restart unless-stopped \
  jync:1.8.2
```

挂载点	内容	要求
`/app/data`	H2 元数据库与密钥文件	必须持久化，且只能被一个实例挂载
`/app/logs`	运行日志	可持久化，也可走 stdout 收集

```
# docker-compose.yml 关键片段
services:
  jync:
    image: jync:1.8.2
    container_name: jync
    ports:
      - "8080:8080"
    environment:
      TZ: Asia/Shanghai
      JAVA_OPTS: "-XX:MaxRAMPercentage=75"
      # HTTPS 部署时启用
      JYN_COOKIE_SECURE: "true"
      # 使用网关头外部认证时填写网关地址
      JYN_TRUSTED_PROXIES: "10.0.0.10"
    volumes:
      - ./data:/app/data
      - ./logs:/app/logs
    restart: unless-stopped
```

不要两个容器挂同一个 H2 卷

H2 文件库在同一时刻只允许一个进程持有；两个容器同时挂载会出现 `jync.mv.db` 锁冲突并反复重启。需要多副本时，必须先切换到外部元数据库。

2.3 Kubernetes

清单位于仓库 `deploy/k8s/`，用 Kustomize 组织。默认单副本，PVC 为 ReadWriteOnce，更新策略为 `Recreate`。

```
kubectl apply -k deploy/k8s/
kubectl -n jync rollout status deploy/jync
kubectl -n jync logs -f deploy/jync
```

要点	默认	说明
副本数	1	H2 元数据无法多副本共享；多副本需外部 MySQL
探针	GET /login	应用未引入 actuator，登录页 200 代表 Web 与元数据库可用
启动探针	30 × 2 秒	大元数据库首启建表较慢时可调大 failureThreshold

要点	默认	说明
日志	emptyDir	`kubectl logs` 查看；需留存则换 PVC
Ingress 超时	300 秒	大表手动同步、连接测试可能超过默认 60 秒
安全上下文	非 root，uid 10001	与镜像内用户一致

密钥放 Secret

外部元数据库口令、`SYNC_CRYPTO_PASSWORD`、`SYNC_CRYPTO_SALT` 都应放在 Secret 中，通过 `secretKeyRef` 注入，不要写进 ConfigMap 或清单明文。

2.4 反向代理与 TLS

生产环境应只暴露 443，由反向代理转到本机 8080。代理必须传递 `Host`、`X-Forwarded-Proto` 与 `X-Forwarded-For`，否则登录 Cookie、重定向与日志里的客户端地址都会失真。

```
# nginx 关键配置
server {
    listen 443 ssl http2;
    server_name jync.example.com;

    ssl_certificate     /etc/nginx/certs/jync.crt;
    ssl_certificate_key /etc/nginx/certs/jync.key;

    location / {
        proxy_pass http://127.0.0.1:8080;
        proxy_http_version 1.1;
        proxy_set_header Host              $host;
        proxy_set_header X-Real-IP         $remote_addr;
        proxy_set_header X-Forwarded-For   $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;

        # 大表手动同步与连接测试可能较慢
        proxy_connect_timeout 10s;
        proxy_send_timeout    300s;
        proxy_read_timeout    300s;
    }
}
```

```
# HTTPS 部署必须打开 Secure Cookie
JYN_COOKIE_SECURE=true
# 默认 lax；被 iframe 嵌入且跨站时设为 None（必须同时开启 Secure）
JYN_COOKIE_SAME_SITE=lax
# 只有当请求的直接对端在下列地址内，才采信网关注入的身份头
JYN_TRUSTED_PROXIES=10.0.0.10,10.0.0.11
# 或写入配置文件
server:
  servlet:
    session:
      cookie:
        secure: true
        same-site: lax
app:
  security:
    trusted-proxies: 10.0.0.10,10.0.0.11
```

Secure Cookie 与 HTTP 不能混用

开启 Secure 后，通过 `http://` 访问不会携带会话 Cookie，表现为登录后又回到登录页。本地调试可临时关闭，生产必须开启并确保代理正确传递 `X-Forwarded-Proto`。

SameSite=None 必须先有 HTTPS

控制台被 iframe 嵌入且属于跨站场景时，会话 Cookie 需要 `SameSite=None`；浏览器只接受同时带 `Secure` 的 None Cookie，因此必须同时设置 `JYN_COOKIE_SECURE=true`。非嵌入场景保持默认 `lax` 即可。

可信代理列表是网关头模式的安全边界

使用「网关头」外部认证时，Jync 只采信直接 TCP 对端位于回环或 `app.security.trusted-proxies` 内的请求头。列表要填网关 / 反向代理的真实出口地址，不能填客户端网段；Jync 本身也不能直接暴露给客户端，否则伪造身份头即可绕过登录。

2.5 首次登录与初始化

1. **登录并改密**

   admin 与 view 的初始密码均为 `123456`，登录后立即修改；view 只用于查看，不应授予管理操作。

2. **核对系统信息**

   在「系统信息」确认版本、Java 版本、工作目录、元数据库路径与日志配置；再到「日志管理 → 系统日志」确认日志文件可读。

3. **添加第一个连接**

   先加一台测试库，点击「测试连接」确认网络与账号；成功后再批量添加生产库。

4 **确定外网策略**

纯内网部署关闭版本检查与 AI；需要一键升级时保留版本检查。

5 **按需接入统一认证**

需要接入企业门户时，进入「配置管理 → 外部认证」选择账号密码、网关头或票据方式，先点测试再启用；启用后再决定是否保留本地登录。

6 **建立备份任务**

把 `data` 目录与密钥纳入日常备份，首次全量前做一次恢复演练。



图 2–1　系统信息：版本、运行环境、工作目录与日志配置集中展示

初始化后的第一轮自检

- ✓ 系统信息中的版本号与下载的发布标签一致。
- ✓ 工作目录、元数据库路径与日志路径符合部署规划。
- ✓ Java 版本为 17 或更高，堆上限符合服务器内存。
- ✓ 连接测试通过，且连接使用的是专用只读/写账号。
- ✓ 外网开关符合预期：纯内网环境已关闭版本检查与 AI。
- ✓ 如启用外部认证：测试按钮返回的身份与来源符合预期，可信代理列表只包含网关地址，本地登录保留策略已确认。

先跑一张小表

初始化完成后，用一个测试同步任务完整走一遍「结构 → 全量 → 增量」。确认目标表结构、行数与变更日志都正常，再接入生产库。

CHAPTER 03

配置参考

Jync 的所有配置都有内置默认值，生产环境通常只需要覆盖少数几项。本章按「服务、同步、CDC、安全、日志」分组说明，并在附录给出完整速查表。调优类参数的取值方法见第 5 章。

3.1 配置来源与优先级

优先级	来源	示例
最高	命令行参数	`--server.port=8081`
高	环境变量	`SERVER_PORT=8081`
中	additional-location 指定的文件	`--spring.config.additional-location=file:./application.yml`
低	JAR 内置 application.yml	随版本发布，不建议修改

Spring 的宽松绑定把 `sync.full-load-parallelism` 映射为 `SYNC_FULL_LOAD_PARALLELISM`，把 `server.port` 映射为 `SERVER_PORT`。容器部署优先使用环境变量，裸机部署使用外部配置文件，两者可以混用，优先级如上表。

> **不要把配置写进 JAR**
>
> 把 `application.yml` 打进 JAR 会让每次升级都要重新打包，也容易丢失变更记录。外部配置文件加版本管理才是可运维的做法。

3.2 服务与元数据库

配置键	默认值	说明
`server.port`	8080	Web 端口；改端口后同步更新代理与防火墙
`server.servlet.context-path`	/	子路径部署时修改，代理需同步
`spring.datasource.url`	H2 文件库	元数据库；可切换为外部 MySQL 8、PostgreSQL 16 或 Oracle 23
`spring.datasource.username`	sa	H2 默认账号；外部库按实际填写
`spring.datasource.password`	空	外部库口令，放环境变量或 Secret

配置键	默认值	说明
`spring.datasource.hikari.maximum-pool-size`	10	元数据库连接池，与业务库连接池相互独立
`spring.jpa.hibernate.ddl-auto`	update	表结构自动演进；不要改成 create 或 none
`spring.quartz.job-store-type`	memory	每节点内存调度；多节点靠对账与锁保持一致

元数据库连接池不是业务连接池

`spring.datasource` 只服务元数据；源库与目标库各自使用独立连接池，大小由 `sync.max-pool-size` 控制。调大元数据池不会提升同步吞吐。

3.3 同步与全量

配置键	默认值	作用
`sync.poll-interval`	2000	增量轮询间隔（毫秒）
`sync.batch-size`	500	增量与 CDC 回放的批次行数，在线流量保持小
`sync.fetch-size`	1000	增量读取的游标批量
`sync.full-load-batch-size`	2000	仅首次全量的写入批次，1–20000，建议 1000–5000；LOB 表建议 500–2000
`sync.full-load-fetch-size`	2000	仅首次全量的读取批量，1–20000；MySQL 源库按行流式读取
`sync.full-load-parallelism`	2	首次全量并行度，1–8，建议 1–4，受连接池二次约束；有主键表都可切分，无主键表顺序加载
`sync.chunk-size`	100000	每个分块的行数，必须大于 0；小于等于 0 表示关闭分块
`sync.max-pool-size`	5	每个数据库连接池上限
`sync.pool-wait-ms`	10000	同一连接配置被多个任务共用时，池满后排队等待这么久再退让；本轮不记失败，下个调度周期重试。0 = 池一满就退让
`sync.oracle-initial-load-insert-first`	true	Oracle 目标为空时用 INSERT 代替 MERGE
`sync.defer-secondary-indexes`	true	新表全量完成后再建非唯一二级索引
`sync.query-timeout-seconds`	300	数据面查询超时；0 关闭
`sync.metadata-query-timeout-seconds`	60	元数据枚举查询超时

配置键	默认值	作用
`sync.shutdown-wait-ms`	20000	停机时等待在途批次的时间
`sync.lock-ttl-ms`	300000	同步任务执行锁有效期，节点崩溃后自动过期
`sync.stuck-threshold-ms`	1800000	一轮无进展多久算卡死（30 分钟；0 关闭告警与页面提示）。高于 10 × lock-ttl-ms（默认 50 分钟）时日志告警不会触发，页面提示仍生效
`sync.reconcile-interval-ms`	30000	多节点调度对账周期
`sync.content-audit-interval-ms`	900000	内容级校验间隔（15 分钟）；`0` 关闭自动校验，手工触发不受限制
`sync.content-audit-sample-size`	50	每表抽样行数，2–200；按主键范围分桶后每桶取一个键
`sync.content-audit-max-rows`	2000000	源行数估计超过此值的表跳过内容校验；`0` 不设上限
`sync.content-audit-max-tables-per-cycle`	1	每个同步周期最多校验的表数；`0` 不限制
`sync.content-audit-timeout-seconds`	60	单次内容校验查询的语句超时

全量参数不会影响增量

全量专用的 batch、fetch 与并行度只在首次加载与重建基线时生效；增量轮询与 CDC 回放继续使用 `sync.batch-size` 与 `sync.fetch-size`。调大全量参数不会给在线流量带来额外压力。

3.4 增量与 CDC

配置键	默认值	作用
`sync.safety-lag-ms`	1000	时间戳游标高水位回退量，避免漏掉乱序提交
`sync.row-count-audit-interval-ms`	60000	双边行数核对周期；0 关闭
`sync.row-count-repair-max-rows`	1000	行数审计发现的缺口不超过这么多行时，按主键定点补行而不是整表重载；0 关闭定点修补
`sync.full-compare-max-rows`	20000	自动选用全表比对的行数上限；任务级手动选择不受此限制
`sync.delete-compare-max-rows`	0	删除比对改为按主键分块比较的行数上限；0 关闭。开启后每轮有一次目标全表扫描与逐键源库确认
`sync.cdc.reconnect-backoff-ms`	1000	断线重连初始退避

配置键	默认值	作用
`sync.cdc.reconnect-backoff-max-ms`	60000	退避上限
`sync.cdc.server-id-offset`	612000	MySQL 假 replica 的 server id 基数
`sync.cdc.lock-ttl-ms`	30000	CDC 消费租约时长
`sync.cdc.supervise-interval-ms`	10000	节点巡检并抢占过期租约的周期
`sync.cdc.connect-timeout-ms`	10000	binlog 客户端握手超时
`sync.cdc.change-window-ms`	10000	变更日志按表聚合窗口；0 关闭聚合
`sync.cdc.logminer.batch-size`	10000	LogMiner 单批读取行数
`sync.cdc.logminer.idle-poll-ms`	1000	LogMiner 真空闲休眠间隔
`sync.cdc.pg.idle-poll-ms`	500	PostgreSQL 真空闲休眠间隔
`sync.cdc.mssql.idle-poll-ms`	1000	SQL Server 真空闲休眠间隔
`sync.cdc.pg.wal-retention-warn-bytes`	4 GiB	PG 源复制槽滞留 WAL 超过该值在面板标红并写 WARN；0 关闭。openGauss 无 pg_lsn，不显示

空闲休眠不是延迟

各类日志 CDC 引擎在有数据时都会连续读取，只有确认无新变更时才休眠。把空闲间隔调小会提高空转频率，并不会降低有数据时的延迟。

3.5 安全、认证、AI 与更新检查

配置键	默认值	说明
`sync.crypto-password`	内置默认	连接口令加密密钥；新部署留空自动生成
`sync.crypto-salt`	内置默认	加密盐；与上面成对管理
`sync.ai.enabled`	true	关闭后隐藏 AI 菜单并拦截所有 AI 请求
`app.update-check.enabled`	true	关闭后不再访问 GitHub / Gitee，一键升级入口隐藏
`app.update-check.timeout-ms`	5000	单平台检查超时，超时改查另一平台
`app.update-check.match-artifact`	true	版本号相同时再比对安装包（GitHub SHA-256 / Gitee 文件大小）；关闭后只比较版本号

配置键	默认值	说明
`server.servlet.session.cookie.http-only`	true	会话 Cookie 不允许脚本读取
`server.servlet.session.cookie.same-site`	lax	阻止跨站 POST 携带 Cookie；可用 `JYN_COOKIE_SAME_SITE` 覆盖，iframe 跨站嵌入需设为 `None` 并同时开启 Secure
`server.servlet.session.cookie.secure`	false	HTTPS 部署必须设为 true
`app.security.trusted-proxies`	空（仅回环）	可信反向代理 / 网关地址列表；只有直接对端命中时才采信网关头，可用 `JYN_TRUSTED_PROXIES` 覆盖
`app.login.reset-on-startup`	false	应急恢复开关：设为 true 启动一次会重新打开本地登录，修复外部认证后必须去掉；可用 `JYN_RESET_LOGIN` 覆盖
`sync.alert.evaluate-interval-ms`	60000	Webhook 告警的评估间隔：上一次评估结束后再等这么久，不是固定周期；告警关闭或未填地址时不评估
`sync.alert.initial-delay-ms`	45000	启动后首次评估的延迟，避开启动阶段的连接与调度初始化

外部认证与告警的配置不在配置文件里

认证方式、接口地址、字段映射、管理员清单与共享密钥保存在元数据库的 `oa_auth_config` 表中，由管理员在「配置管理 → 外部认证」页面维护；Webhook 地址、告警分类与冷却时间保存在 `alert_config` 表中，入口是「配置管理 → 告警通知」，地址加密保存、页面只回显尾号。配置文件里的 `trusted-proxies` 与 `same-site` 是网关头与 Cookie 的安全边界，两者必须一起核对。

密钥不要跟着升级改

从旧版本升级上来的部署，`crypto-password` 与 `crypto-salt` 是历史密文的解密钥匙。擅自改成新值会让已保存的数据库连接口令全部无法解密，界面上必须重填。新部署留空即可，首次启动自动生成密钥文件。

3.6 日志与 JVM

配置键	默认值	说明
`logging.file.name`	./logs/jync.log	日志文件路径
`logging.logback.rollingpolicy.max-file-size`	10MB	单文件大小上限
`logging.logback.rollingpolicy.max-history`	14	保留天数
`logging.logback.rollingpolicy.total-size-cap`	500MB	日志总量上限

配置键	默认值	说明
`logging.logback.rollingpolicy.clean-history-on-start`	true	启动时清理超限历史与异常退出残留
`logging.level.root`	INFO	排障时可临时调到 DEBUG，事后恢复
`logging.level.org.hibernate.SQL`	WARN	元数据 SQL 不打印，避免日志膨胀

> **日志编码固定为 UTF-8**
>
> JDK 17 在中文 Windows 上的默认字符集是 GBK，若按默认写日志，中文再读回会乱码。应用固定以 UTF-8 写入，系统日志页也用同一编码读取，不随服务器区域设置变化。

JVM 建议使用 G1，堆大小按物理内存的 50%–75% 设置。全量并行度较高时，堆主要被行缓冲与批次占用，4 GB 堆可支撑默认并行度；把并行度提到 4 以上前，先确认内存与数据库连接池都能承受。

```
# 4 GB 内存服务器示例
JAVA_OPTS="-Xms1g -Xmx3g -XX:+UseG1GC -XX:MaxGCPauseMillis=200"

# 容器内按内存限额自动取堆（镜像默认）
JAVA_OPTS="-XX:MaxRAMPercentage=75"
```

3.7 性能采样与指标

v1.10.0 起，应用按固定间隔在本地元数据库落三类指标样本，供控制台的数据库性能页与同步任务性能页回看；Prometheus 端点输出的是同一批最新样本，不额外查询业务库。

配置键	默认值	说明
`metrics.enabled`	true	性能采样总开关。关闭后不再采样，两端性能页显示「采样已关闭」，Prometheus 端点不输出业务指标
`metrics.retention-days`	7	样本保留天数，按天清理；调大会增加元数据库体积
`metrics.task-interval-ms`	60000	同步任务指标采样间隔（1 分钟）
`metrics.db-interval-ms`	300000	数据库指标采样间隔（5 分钟）
`metrics.capacity-interval-ms`	3600000	表数 / 行数 / 占用空间等容量采样间隔（1 小时）
`metrics.prometheus-token`	空	Prometheus 端点的 Bearer token；留空时端点返回 404。可用 `JYN_METRICS_TOKEN` 覆盖

采样是只读的短会话

数据库采样会在被监控库上打开连接执行只读查询，间隔越短、连接越多，源库负载越高。生产环境保持默认的 5 分钟与 1 小时即可；不需要观测时把 `metrics.enabled` 设为 `false` 。

指标端点默认关闭

没有配置 token 时 `/api/metrics/prometheus` 直接返回 404，不暴露任何指标；配置 token 后只接受 `Authorization: Bearer`，不要把它和业务接口一起暴露到公网。

CHAPTER 04

元数据、备份与多实例

Jync 的元数据库保存了所有配置与进度：连接、同步任务、ETL 任务与批次、游标、位点、锁、变更日志与结构快照。它决定了平台能否在重启后接着跑，也决定了备份与多实例的边界。

4.1 元数据库里有什么

表	内容	丢失后果
`database_config`	数据库连接与加密后的口令	需要重建全部连接
`project`	同步任务、对象范围、同步选项与策略	需要重建全部同步任务
`sync_progress`	每表游标与进度	下次从当前游标重来，可能重复加载
`chunk_checkpoint`	全量分块检查点与区间边界	全量从头开始，已写入数据幂等覆盖
`cdc_position`	CDC 位点（SCN / LSN / binlog 位点）	需要重建基线
`sync_task`	运行状态、锁持有者与锁过期时间	锁状态丢失，运行中的任务可能重入
`change_log`	数据变更与结构变更记录	审计链断裂
`metadata_snapshot`	源库结构快照，用于差异比对	下次结构同步重新采集
`app_user`	控制台账号	需要重建账号
`audit_event`	管理操作审计	审计记录丢失
`ai_provider`	AI 供应商配置	需要重新配置
`oa_auth_config`	外部认证方式、接口地址、字段映射与共享密钥	外部认证失效，需要重新配置
`etl_task`	ETL 任务的取数 SQL、目标表、写入方式与批次归属	需要重建 ETL 任务
`etl_batch`	ETL 批次名称、源 / 目标连接与 cron	批次调度丢失，成员任务仍在但不再被批次驱动
`etl_checkpoint`	ETL 断点续跑位置	追加与范围覆盖从当前位置重来，可能重复写入
`etl_run`	ETL 每次运行的成功与否、影响行数与时间	看板 ETL 趋势线缺历史，任务最近结果不受影响
`app_metric_sample`	应用进程指标样本（CPU、堆、线程等）	性能曲线缺应用段历史，不影响运行

表	内容	丢失后果
`db_metric_sample`	数据库连接指标样本（延迟、会话、QPS、缓存命中率、锁等待）	数据库性能页只剩新样本
`task_metric_sample`	同步任务指标样本（行数、延迟、失败次数、变更构成）	任务性能页只剩新样本
`migration_connection` / `migration_task` / `migration_run`	专属迁移的连接（加密凭据）、任务与运行历史	专属连接与任务需要重建，实时同步与 ETL 不受影响
`alert_config` / `alert_state`	Webhook 地址（加密）、告警分类、冷却时间与已通知状态	告警停止外呼，需要重新配置

元数据不是数据备份

元数据库只保存「同步到了哪里」，不保存业务数据本身。它丢失后目标库里的数据仍在，但平台不知道进度，通常需要重建基准或重新全量。

4.2 H2 与外部元数据库

维度	内置 H2 文件库	外部 MySQL 8 / PostgreSQL 16 / Oracle 23
部署复杂度	零配置，随应用启动	需独立实例、库与账号
多副本	不支持	支持
备份方式	停机复制 data 目录	mysqldump / pg_dump / RMAN 等厂商工具
故障恢复	依赖本地磁盘与备份	可复用现有数据库高可用
适用规模	单实例，中小规模	多实例与较大规模

```
# 切到外部 MySQL 8
CREATE DATABASE jync DEFAULT CHARACTER SET utf8mb4;

# application.yml
spring:
  datasource:
    url: jdbc:mysql://10.0.0.20:3306/jync?
useUnicode=true&characterEncoding=utf8&serverTimezone=Asia/Shanghai
    username: jync
    password: ${JYN_META_PASSWORD}
    hikari:
      maximum-pool-size: 10

# PostgreSQL 16 的 URL 形态
# url: jdbc:postgresql://10.0.0.21:5432/jync
# Oracle 23 的 URL 形态（Service Name）
# url: jdbc:oracle:thin:@//10.0.0.22:1521/JYNC
```

> **切换元数据库不是迁移数据**
>
> 从 H2 切到外部库后，新库里没有任何同步任务与连接。要么重新配置，要么按厂商工具把表结构与数据导出再导入。切换前务必确认源库变更能被重新补齐。

4.3 密钥文件

数据库连接口令以 AES 加密后存库。新部署不填写 `sync.crypto-password` 与 `sync.crypto-salt` 时，首次启动会自动生成强密钥并写入元数据库同目录的 `<库名>.key`（默认 `data/jync.key`）。

- **必须一起备份**只备份 `jync.mv.db` 而不备份 `jync.key`，恢复后所有连接口令都无法解密。
- **权限收紧**密钥文件建议 `chmod 600`，属主为运行账号。
- **多实例自管**外部元数据库没有本地密钥文件可依附，多实例部署必须显式设置 `SYNC_CRYPTO_*`，并保证各节点一致。
- **升级不改**从旧版本升级时保持原密钥不变；它是历史密文的解密钥匙。

```
# 检查密钥文件
ls -l /opt/jync/data/jync.key
# 备份时与库文件放在一起
tar czf jync-meta-$(date +%F).tgz -C /opt/jync data
```

4.4 备份与恢复

1 **停机备份**

停止 Jync，复制整个 `data` 目录（含密钥文件）。这是最稳妥的方式，避免 H2 写入过程中复制出不一致文件。

2 **外部库备份**

元数据为 MySQL 时，使用 `mysqldump --single-transaction`，并把密钥另行保存到 Secret 管理系统。

3 **异地保存**

备份文件与密钥分开存放；只拿到库文件拿不到密钥，恢复后口令仍需重填。

4 **定期演练**

每季度至少做一次恢复演练：在隔离环境启动，确认同步任务、连接与进度可见。

5 **升级前必备份**

任何版本变更前先做一次完整备份，并记录当前版本号。

```
# 停机备份
systemctl stop jync
tar czf /backup/jync-$(date +%F-%H%M).tgz -C /opt/jync data
systemctl start jync

# 恢复
systemctl stop jync
mv /opt/jync/data /opt/jync/data.bak.$(date +%F)
tar xzf /backup/jync-2026-10-22-1200.tgz -C /opt/jync
chown -R jync:jync /opt/jync/data
systemctl start jync
```

停机期间不会丢数据

Jync 停机时源库仍在变化。轮询同步任务重启后从游标继续，CDC 同步任务从位点继续；只要源库日志保留时间覆盖停机时长，增量就能自动补齐。停机超过日志保留时间时，需要重建基线。

4.5 多实例部署

多实例的正确性不依赖节点间通信，而依赖共享元数据库。三个机制共同保证「不会双跑、不会漏跑」：调度对账、执行互斥锁、CDC 租约。

机制	作用范围	默认周期
调度对账	把本地 Quartz 调度对齐到数据库里的启用状态	30 秒
执行互斥锁	同一同步任务同一时刻只在一个节点执行	锁 TTL 5 分钟

机制	作用范围	默认周期
CDC 租约	同一 CDC 同步任务只有一个节点在消费	租约 30 秒，巡检 10 秒

1. **切换元数据库**

   把 `SPRING_DATASOURCE_URL` 指向外部 MySQL 8、PostgreSQL 16 或 Oracle 23，各节点使用同一库与同一套密钥。

2. **扩大副本**

   Kubernetes 调整 replicas，或在前端负载均衡后挂多台裸机实例。

3. **处理会话**

   控制台会话默认存在各节点内存中，负载均衡建议开启会话保持；否则刷新到另一节点会要求重新登录。外部认证用户同样只保留会话，切节点后需要重新走一次门户登录。

4. **验证接管**

   停掉持锁节点，确认其他节点在一个锁周期内接管；CDC 同步任务在一个租约加巡检周期内接管。

> **对账只保证存在性**
>
> 在某个节点修改轮询间隔或 Cron 后，其他节点会在该同步任务下次于本节点被调度时对齐。对同步语义没有影响，因为执行锁保证不会双跑，但排障时不要以「某节点上的调度时间」作为唯一依据。

CHAPTER 05

全量性能调优

首次全量往往是整个迁移里最耗时的一步。Jync 的默认值取的是「数据库压力可控」的均衡档；本章说明每个参数影响什么、连接池如何限制并行度，以及怎样用日志判断瓶颈在读、写还是提交。

5.1 先理解吞吐构成

全量加载的每一行都要经过读、清洗、绑定、批量提交四个阶段。每张表、每个并行 worker 结束时都会输出一行吞吐日志，把「慢」拆成读、写、提交三个可操作的占比。调优的第一步不是改参数，而是看这一行。

```
# 典型吞吐日志
table=ORDERS thread=full-load-ORDERS rows=500000 changed=500000 elapsed=2.3s rows/s=216000.0 \
  read=38% write=52% commit=10% batches=250 rowsPerBatch=2000
```

字段	含义	怎么用
`rows` / `changed`	读取行数 / 实际写入行数	与本表行数核对；两者差距大说明源端过滤或映射丢行
`elapsed` / `rows/s`	耗时与吞吐	跨轮次、跨调参对比的主指标
`read%`	源库取数 + 行映射占比	偏高说明源库游标、网络或映射是瓶颈
`write%`	清洗 + 绑定 + `addBatch` + `executeBatch` 占比（不含提交）	偏高说明目标库写入、约束或大字段绑定是瓶颈
`commit%`	目标库 `COMMIT` 等待占比	偏高说明 redo、备库同步或磁盘延迟是瓶颈
`batches` / `rowsPerBatch`	批次数 / 每批实际行数	`rowsPerBatch` 接近批量大小才说明驱动真的在批处理

合计 100%，并有采样门槛

`read% + write% + commit%` 合计 100%。只有 `rows ≥ 1000` 且 `changed > 0` 的轮次打 INFO，小窗口与无变化的轮次降到 DEBUG，避免刷屏。

先记录基线再改

用默认参数跑一张代表性的大表，记录 `rows/s` 与读写占比。每次只改一个参数，复测同一张表，否则无法判断收益来自哪里。

5.2 全量参数矩阵

参数	默认	取值范围（硬限制）	建议	作用与取舍
`sync.full-load-batch-size`	2000	1–20000，超出截断到 20000；任务级 ≤0 视为未设置	1000–5000；宽表或大字段 500–2000	每批行数。批次越大提交越少、越快；超过 5000 后提速趋缓，回滚与内存成本继续上升
`sync.full-load-fetch-size`	2000	1–20000	2000–5000（MySQL 源不生效）	每次向源库取回的行数。调大减少往返，但增加客户端内存与游标压力
`sync.full-load-parallelism`	2	1–8，且受连接池压制	1 → 2 → 4；4 以上收益明显递减	每表并行 worker 数。只对有主键表生效，实际取 `min(配置值, 8, 池能容纳的 worker 数)`
`sync.chunk-size`	100000	≤0 关闭分块，整表退回单次流式读取（此时并行度不生效）	50000–200000	分块粒度。分块越小，单块失败重做范围越小、检查点越密
`sync.max-pool-size`	5	无上限，内部至少按 2 计	5–10，按并行度反算	每个数据库的连接池上限。池不够时 worker 只会排队，吞吐不会随并行度上升

超出范围不会报错

批次与 fetch 超过 20000、并行度超过 8 都会被截断到上限并在日志中提示； `chunk-size ≤ 0` 则关闭分块，整表退回单次流式读取，此时并行度不生效。MySQL 源库按行流式读取，fetch 主要影响客户端缓冲，不会把整表拉进内存。

作用范围

以上参数只影响首次全量与重建基线，不改变增量轮询与 CDC 回放。全局值写在 `application.yml`；批次与并行度可在同步任务详情页的「全量加载性能」卡片按任务覆盖（留空用全局值），fetch、chunk-size 与连接池只读全局配置。

5.3 连接池与并行度

并行不是「配多少就开多少」。每个分块 worker 需要一条源库连接和一条目标库连接，主流程还会占用连接，因此实际并行度由连接池二次封顶：

拓扑	连接池要求	pool=5 时
源库与目标库使用不同连接配置	`max-pool-size ≥ 并行度 + 1`	最多 4
源库与目标库共用同一个连接配置	`max-pool-size ≥ 2 × 并行度 + 2`	最多 1

拓扑	连接池要求	pool=5 时
并行度配置值超过 8	按 8 生效	8

> **同库同配置下并行度被压得很低**
>
> 源库与目标库共用同一个连接配置时，连接池被主连接与 worker 双向占用，pool=5 只能支持 1 个 worker。想在同库内做表到表同步并提高并行度，应先按 `2 × 并行度 + 2` 调大 `max-pool-size`，或为源 / 目标分别建立连接配置。

切分能力在 1.7.8 起覆盖所有有主键的表：**整数前导主键**按区间算术切分，**其余键形态**（字符串、复合键、日期键等）按采样边界切分，由多个 worker 并行分页读取与写入；只有**无主键表**无法安全切分，退回顺序加载。采样边界会校验区间不重叠，检查点仍可续跑。

v1.8.2 起，同一个连接配置上并发运行的多个任务会按活跃任务数分配并行度：系统记录该配置上活跃任务占用的席位（同库两端算两席），新任务按 `（池大小 − 已占）/ 已占` 折算并行度，准入上限为池的一半，另一半留给全量 worker；只有一个任务时并行度与原来一致。席位被占满时，本轮按 `sync.pool-wait-ms` 每 200 毫秒探测一次池计数，仍拿不到位置就退让到下一个调度周期，不再等 Hikari 十五秒超时后整轮失败并累计失败次数。

5.4 Oracle 专项优化

Oracle 是当前客户最常见的源库与目标库。Jync 针对 Oracle 目标做了三项默认开启的优化：

优化	配置	作用
空表 INSERT 优先	`oracle-initial-load-insert-first`	目标表为空时写 INSERT 而不是 MERGE，省掉匹配判断
二级索引延后	`defer-secondary-indexes`	数据灌完后再建非唯一索引，避免逐行维护索引
分块并行	`full-load-parallelism`	整数前导主键按区间切分，其余有主键表按采样边界切分，多 worker 并行写入

Oracle 目标 50 万行基准（10 列，主键加两个二级索引）显示：并行度 1 约 117k 行/秒，并行度 2 约 184k 行/秒，并行度 4 约 216k 行/秒。索引延后使同一批数据的加载时间从约 11.1 秒降到 5.9 秒，事后建索引约 0.5 秒。空表 INSERT 优先在基准中约为 MERGE 的 2.5 倍。

> **主键与唯一索引仍然先建**
>
> 延后只针对非唯一二级索引。主键必须在建表时就有，upsert 依赖它；唯一索引必须立即生效，才能在源端有重复行时及时报错。

不要使用直接路径加载

Jync 不使用 `/*+ APPEND */`、PDML，也不会在加载前删除或重建主键。这些做法收益高但会破坏幂等与并发安全，不属于受支持的调优手段。

5.5 一次完整的调优过程

1. **建立基线**

   用默认参数跑一张代表性大表，记录 `rows/s` 与 `read/write/commit` 占比、源库与目标库的 CPU/IO，以及数据库连接数。

2. **确认并行生效**

   检查表是否有主键，连接池是否满足并行度要求（不同配置为 `pool ≥ 并行度 + 1`，同一配置为 `pool ≥ 2 × 并行度 + 2`）；日志中应出现多个 worker 线程名。实际并行度取 `min(配置值, 8, 池能容纳的 worker 数)`，被压低时日志会提示。

3. **先按最大占比调一侧**

   `commit%` 最大：批量 2000 → 5000 减少提交次数，确认存储与备库同步；`write%` 最大：查目标库约束、索引维护与大字段绑定，写入期间可关闭外键；`read%` 最大：确认游标列有索引，fetch 提到 2000–5000，再考虑并行度 1 → 2 → 4，并错峰执行。

4. **三项都不高但吞吐低**

   先怀疑网络 RTT、批次太小或单线程：批量不低于 1000，并行度 1 → 2（基准约 +57%），并检查链路延迟。

5. **再加并行度**

   把同步任务或全局并行度调到 4，复测。若吞吐提升但数据库压力明显，回退到 2 或 3；每次只改一个变量并重新看日志，4 以上收益明显递减。

6. **确认稳定**

   连续跑两张不同规模的表，吞吐没有明显衰减，错误日志为空，再固化配置；生产库与业务共享时把并行度设为 1，批量保持 1000–2000。

同步任务级覆盖优先

「全量加载性能」卡片可以为单个同步任务设置批次与并行度，优先级高于全局配置。大表同步任务单独提高并行度，小表同步任务保持默认，比全局调大更安全。

5.6 常见反模式

做法	问题	正确做法
把批次调到 20000 以上	被静默收敛，且大事务增加锁与回滚成本	全量 2000–5000，按目标库承受能力微调
并行度调到 8 但 pool 仍是 5	worker 排队抢连接，吞吐不升反降	按拓扑先满足 `pool ≥ 并行度 + 1` 或 `pool ≥ 2 × 并行度 + 2`
业务高峰执行首次全量	源库与目标库同时承压，影响在线业务	安排在低峰期，必要时并行度设为 1
对无主键表期待并行	无法切分区间，只能顺序加载	补主键，或接受顺序加载并延长窗口
只调 JVM 堆不看数据库	瓶颈通常在数据库写入或索引维护	先看读写占比，再决定调哪一侧

CHAPTER 06

日志 CDC 运维

日志 CDC 把延迟从「轮询间隔」降到秒级，并能看到 UPDATE 与 DELETE。代价是源库必须开启日志能力，账号需要额外权限，运维上还要关注位点、租约与日志保留。本章按上线顺序展开。

6.1 日志 CDC 引擎与位点

源库	引擎	最低版本	位点	读取方式
MySQL 兼容家族	binlog	MySQL 5.7（不支持 MariaDB）	binlog 文件 + 偏移，含 GTID	推送协议
Oracle	LogMiner	11g	SCN	按批查询
达梦 DM8	LogMiner	DM8	SCN（最新归档的 NEXT_CHANGE#）	按批查询
PostgreSQL 方言家族	逻辑解码	PostgreSQL 9.4	LSN	流式
SQL Server	CDC	2012	LSN（20 位十六进制）	按批查询
崖山 YashanDB	YStream	23.x	提交 SCN + 实例 + 组 LSN + 组偏移 + 行号	流式

同一家族里的产品按能力探测放行：MySQL 兼容产品（GBase、OceanBase、TiDB、GoldenDB、TDSQL、PolarDB 等）与 PostgreSQL 方言家族（openGauss、Kingbase、HighGo、Vastbase 等）只要具备对应日志能力即可使用，探测不过时会逐项说明缺什么，而不是笼统拒绝。

> **位点是恢复的核心**
>
> 每个 CDC 同步任务只保存一个位点。位点之前的变更已经写入目标库，位点之后的变更等待回放。任何「清位点」的操作都意味着需要重建基线，不能当成普通的重置。

v1.10.0 起，位点同时保存每张表的列序指纹。消费者停机期间源表执行过 DDL 时，重启会停在「列布局漂移」上并给出受影响表名，而不是把日志里的值按位置写进相邻列；MySQL 源开启 `binlog_row_metadata=FULL` 时还会逐份 TABLE_MAP 校验列名与顺序。

6.2 源库准备 SQL

MySQL

```
-- 确认日志能力
SHOW VARIABLES LIKE 'log_bin';            -- 必须 ON
SHOW VARIABLES LIKE 'binlog_format';      -- 必须 ROW
SHOW VARIABLES LIKE 'binlog_row_image';   -- 必须 FULL
SHOW MASTER STATUS;                        -- 记录当前文件与偏移

-- 账号权限
GRANT REPLICATION SLAVE, REPLICATION CLIENT ON *.* TO 'jync'@'%';
FLUSH PRIVILEGES;
```

```
# my.cnf 关键项
[mysqld]
log_bin = mysql-bin
binlog_format = ROW
binlog_row_image = FULL
expire_logs_days = 7
server_id = 1
```

Jync 会以一个「假 replica」的身份连接源库，server id 为 `server-id-offset + 项目ID`。该值不能与源库自身或任何真实从库冲突，否则源库会断开复制连接。同步任务较多时，确认偏移段没有被占用。

Oracle LogMiner

```
-- 归档模式（需停机或重启到 mount 状态）
SHUTDOWN IMMEDIATE;
STARTUP MOUNT;
ALTER DATABASE ARCHIVELOG;
ALTER DATABASE OPEN;

-- 补充日志
ALTER DATABASE ADD SUPPLEMENTAL LOG DATA;
ALTER TABLE app.orders ADD SUPPLEMENTAL LOG DATA (ALL) COLUMNS;

-- 挖掘账号
CREATE USER jync IDENTIFIED BY "******";
GRANT CREATE SESSION TO jync;
GRANT SELECT ANY TRANSACTION TO jync;
GRANT SELECT ANY DICTIONARY TO jync;
GRANT EXECUTE ON DBMS_LOGMNR TO jync;
GRANT SELECT ON V_$LOGMNR_CONTENTS TO jync;
GRANT SELECT ON V_$LOG TO jync;
```

CDB 环境必须在根容器挖掘

多租户 Oracle 中，Jync 需要以公共用户（`C##` 前缀）连接 CDB$ROOT 执行挖掘，再按容器与表名过滤。用普通 PDB 用户连接会看不到跨容器的归档日志。

另外确认：表没有 LONG 列；归档日志保留不少于 24 小时；挖掘账号对目标 schema 有读取权限。归档被清理后，位点指向的 SCN 不再可读，只能重建基线。

PostgreSQL 逻辑解码

```
-- postgresql.conf
wal_level = logical
max_replication_slots = 10
max_wal_senders = 10

-- 修改后重启数据库
-- 账号
CREATE USER jync WITH REPLICATION LOGIN PASSWORD '******';
GRANT CONNECT ON DATABASE appdb TO jync;
GRANT USAGE ON SCHEMA public TO jync;
GRANT SELECT ON ALL TABLES IN SCHEMA public TO jync;

-- 无主键表的 REPLICA IDENTITY
ALTER TABLE public.orders REPLICA IDENTITY FULL;
```

REPLICA IDENTITY

有主键的表默认使用主键作为标识；无主键表必须设置为 `FULL`，否则逻辑解码无法给出完整的旧行，UPDATE 与 DELETE 会被跳过或报错。

SQL Server CDC

```
-- 库级启用（每库一次）
USE appdb;
EXEC sys.sp_cdc_enable_db;

-- 表级启用（每表一次）
EXEC sys.sp_cdc_enable_table
    @source_schema = N'dbo',
    @source_name   = N'orders',
    @role_name     = NULL;

-- 确认 capture instance
EXEC sys.sp_cdc_help_change_data_capture;
```

SQL Server Agent 必须处于运行状态，CDC 的捕获作业由 Agent 驱动。账号需要在 `cdc_admin` 或 `cdc_reader` 角色中。若未建 capture instance，Jync 会提示找不到变更表。

达梦 DM8（LogMiner）

```
-- 归档模式（dm.ini：ARCH_INI=1 并配置归档）
SELECT ARCH_MODE FROM V$DATABASE;          -- 必须 Y

-- 旧镜像写满 redo（静态参数，修改后需重启实例）
SP_SET_PARA_VALUE(1, 'RLOG_APPEND_LOGIC', 2);

-- 补充日志与授权
ALTER TABLE app.orders ADD SUPPLEMENTAL LOG DATA (ALL) COLUMNS;
GRANT CREATE SESSION, SELECT ANY TRANSACTION, SELECT ANY TABLE TO jync;
GRANT EXECUTE ON SYS.DBMS_LOGMNR TO jync;

-- 产生可挖掘的归档
ALTER SYSTEM SWITCH LOGFILE;
```

达梦只能挖掘归档文件，不能挖掘联机日志，CDC 延迟因此取决于归档切换间隔；基线起点取最新归档的 `NEXT_CHANGE#`，没有任何归档时会提示先 `ALTER SYSTEM SWITCH LOGFILE`，不会反复重试。归档保留时间同样需要覆盖停机窗口。

崖山 YashanDB（YStream）

```
-- 归档模式与主库角色
SELECT LOG_MODE, DATABASE_ROLE FROM V$DATABASE;
-- 管理视图可读（探测以此判断 YStream 能力）
SELECT COUNT(*) FROM V$YSTREAM_SERVER;

-- 授权
GRANT SELECT ANY DICTIONARY TO jync;
GRANT YSTREAM_CAPTURE TO jync;

-- 逐表补充日志，缺了 DML 会静默丢失
ALTER TABLE app.orders ADD SUPPLEMENTAL LOG DATA (ALL) COLUMNS;
```

YStream 的源侧服务由 Jync 按同步任务生命周期创建与停止（创建任务、追加表、启动、巡检、停止并删除），位点会回执给源端以回收 redo。不要在 Jync 之外复用同一个 YStream 服务。实测崖山 23.4.7.100 没有 `DBMS_LOGMNR`，崖山源请选 YStream。

6.3 启动与位点纪律

1. **先全量后回放**
   切换到 CDC 时，平台会先执行一次基线全量，再记录当前位点并开始回放。顺序不能颠倒。

2. **确认位点已记录**
   同步任务页应显示「已应用位点」与「落后秒数」，位点随时间推进。

3 **不要直接重置进度**

CDC 同步任务应使用「重建基线」而不是「重置进度」。前者重新记录位点，后者只清游标，可能留下不匹配的旧位点。

4 **停止时保留位点**

停止同步不会清除位点，并会通知正在执行的一轮在下一个已提交批次边界安全退出，随后释放同步锁；恢复后从上次位点继续，停机期间的变更由日志补齐。

5 **核对源库日志保留**

停机时长超过日志保留时间时，位点失效，必须重建基线。

6.4 状态面板与关键指标

指标	正常表现	异常含义
已应用位点	持续前进	不前进说明消费被阻塞或源库无新变更
落后秒数	秒级波动	持续上涨说明目标写入跟不上
吞吐	随业务量波动	突然归零说明连接断开或租约丢失
跳过计数	0 或极少	持续增长说明有变更无法应用，需查日志
错误计数	0	非 0 时先看具体 SQL 与主键
消费节点	仅一个节点显示消费中	多个节点同时消费说明租约或时钟异常
列布局漂移	0	停机期间源表 DDL 导致列序指纹不一致；任务停在漂移上，需要重建基线

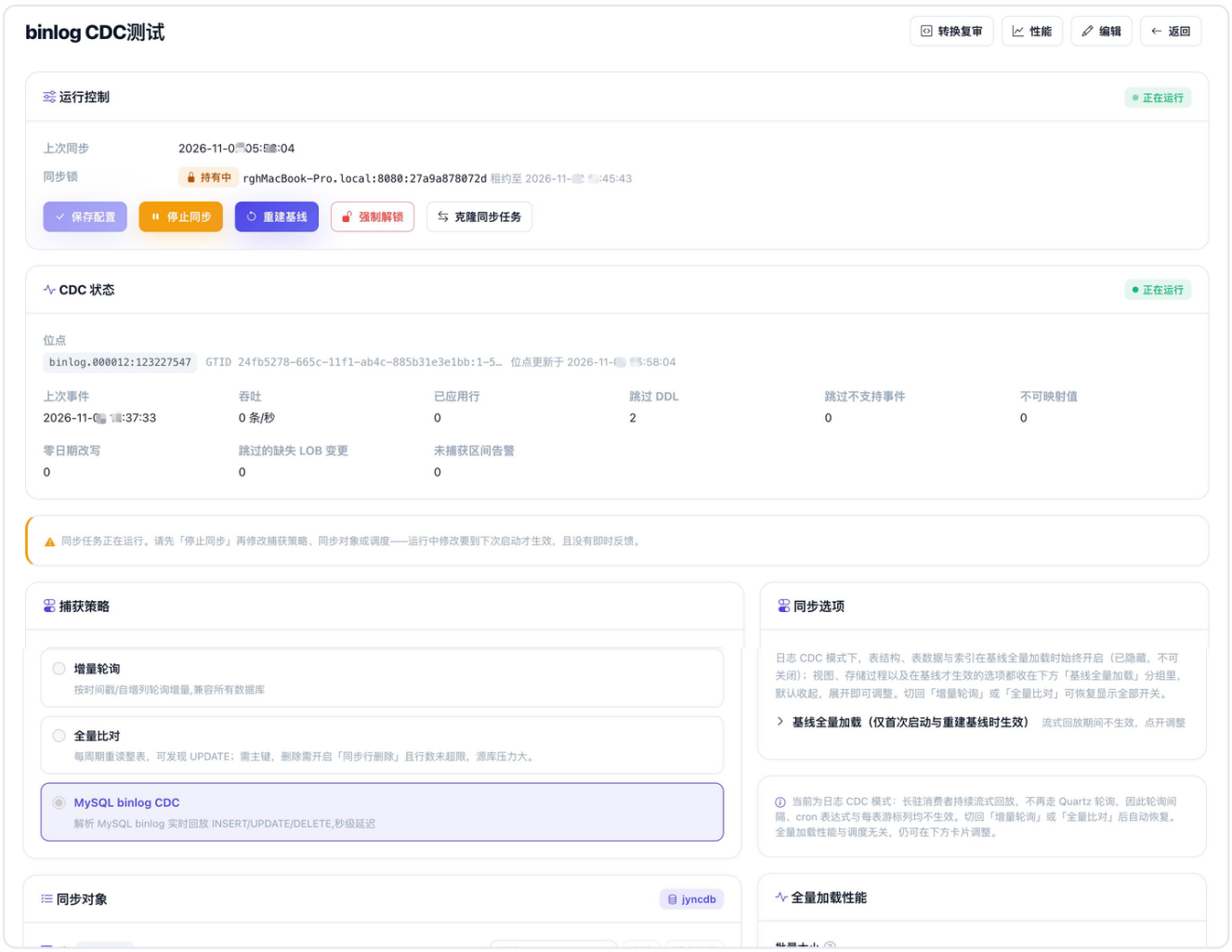


图 6–1　CDC 状态面板：位点、落后秒数、吞吐与错误计数集中呈现（上半部分）

6.5 延迟与积压处理

现象	可能原因	处理
延迟缓慢上涨	目标库写入慢、索引多、锁等待	查看目标库等待事件；降低增量批次、错峰、优化索引
延迟突然跳到小时级	消费线程中断后恢复，正在追赶历史变更	观察位点是否持续推进；不要重启打断追赶
位点不动且无错误	源库没有新变更，或日志读取被阻塞	先在源库制造一条变更验证；再查日志读取权限
跳过计数增长	目标表缺列、主键冲突、类型不兼容	对照日志中的表与主键，修复结构后重建基线
日志被清理	停机或积压超过源库保留时间	无法补读，必须重建基线

先看位点再看延迟

位点不前进时，延迟数字没有意义。先确认消费是否被阻塞，再判断是否只是回放偏慢。

6.6 断线、重连与租约接管

各类日志 CDC 引擎都使用指数退避重连：初始 `reconnect-backoff-ms`（默认 1 秒），逐次翻倍，封顶 `reconnect-backoff-max-ms`（默认 60 秒）。短暂网络抖动会在数秒内恢复；持续失败会在日志中留下明确的连接错误。

CDC 消费线程使用独立短租约，默认 30 秒，各节点每 10 秒巡检一次。持锁节点宕机后，其他节点在「租约到期 + 一个巡检周期」内接管，通常 30–40 秒。接管期间位点不会丢失，恢复后继续回放。

v1.8.1 起，Oracle LogMiner、PostgreSQL 逻辑复制与 SQL Server CDC 只把「应用数据失败」计入连续失败：源库短暂连不上按指数退避无限重连，不再把任务置为错误状态；MySQL 消费者原本就是这个口径。停止 CDC 时两端连接在管理器锁外建立，点「停止」不会被建连过程卡住。

> **强制解锁不终止消费线程**
>
> 强制解锁只释放数据库里的锁行，不会终止已经运行的 CDC 消费线程。只有在确认没有任何节点在执行该同步任务时才使用，否则会造成同一同步任务并发消费。

6.7 重建基线与策略切换

场景	动作	影响
位点失效、日志被清理	重建基线	重新全量并记录新位点，目标库承受一次全量写入
源表结构变更导致跳过	修复结构后重建基线	同上；修复前不要反复重启
从轮询切到 CDC	切换策略	自动重建基线，切换期间延迟归零重算
从 CDC 切回轮询	切换策略	位点保留，但游标清零并先全量对齐
同步任务迁移到新目标库	新建同步任务，不复用旧同步任务	避免旧位点与新目标库不匹配

重建建议在低峰期执行，并先确认目标库磁盘与连接数足够。重建过程中不要同时修改目标表结构，否则全量与回放会看到不一致的元数据。

CHAPTER 07

轮询同步运维

轮询是默认模式：只需要只读账号，不碰日志，不建触发器。它的代价是延迟下限等于轮询间隔，并且看不到 DELETE。理解游标策略与安全滞后，是轮询运维的全部要点。

7.1 五种游标策略

策略	依赖	能看到	适用
TIMESTAMP	更新时间列	新增、更新	有可靠 update_time 的表
ROW_SCN	Oracle ORA_ROWSCN	新增、更新	Oracle 源库
IDENTITY	自增或序列主键	仅新增	只追加的日志、流水表
FULL_COMPARE	主键；自动选择时行数受限	新增、更新；删除需开启「同步行删除」且行数 ≤ 上限	有主键的小表，行数低于 20000
NONE	无	不检测变更	仅做一次全量，或人工触发

策略是同步任务级的

捕获策略按同步任务设置，不能逐表分别指定；逐表只能覆盖游标列。没有可用游标列的表不会自动变成全表比对，而是按行数上限决定是否允许 FULL_COMPARE：超过上限时自动解析为 NONE（跳过并记录原因）。在同步任务级手动选择「全量比对」不受该上限约束；详情页与启动日志会列出超限、或按当前轮询间隔负担过重的表，并给出建议间隔（按约 1,000 行/秒估算）。

7.2 游标列的选择与维护

- **单调性**游标列必须随写入单调递增。时间列在并发事务下可能乱序提交，这正是安全滞后存在的原因。
- **索引**游标列应有索引，否则每轮轮询都会全表扫描，源库压力随数据量线性上升。
- **精度**时间列精度至少到秒；毫秒精度更稳妥，但要注意数据库默认精度可能截断。
- **不可回填**业务回填历史时间会导致游标回头，新数据可能被跳过；需要回填时先停止同步任务并重建基线。

```
-- MySQL：确认游标列有索引
SHOW INDEX FROM app.orders WHERE Column_name = 'update_time';

-- Oracle：确认列与索引
SELECT column_name, data_type FROM all_tab_columns
 WHERE owner = 'APP' AND table_name = 'ORDERS';
SELECT index_name, column_name FROM all_ind_columns
 WHERE table_owner = 'APP' AND table_name = 'ORDERS';
```

7.3 轮询间隔与源库负载

默认间隔 2 秒。间隔越短，延迟越低，但每轮都要执行一次游标查询；表多、数据量大时，间隔过短会让源库持续处于查询压力下。判断依据是源库的 CPU、慢查询与连接数，而不是「感觉不够快」。

v1.8.1 起，常规任务不再每个轮询周期为每张表查询一次目录估算行数：只有开启「同步行删除」或显式全量对比的任务才预取估算，其余任务在「没有任何游标候选」的兜底判断里才现取。表很多的库因此每轮少掉一轮目录往返（2000 张表、20 ms 链路约 40 秒），这类任务的元数据快照也不再因行数变化而每轮重写。

场景	建议间隔	理由
核心业务表，要求近实时	1–2 秒	延迟可接受，查询压力可控
普通报表/同步表	5–30 秒	降低源库查询频率
超大表、弱索引	30–60 秒	优先保证源库稳定，或改用 CDC
仅每日一次	Cron 调度	无需持续轮询

按同步任务设置间隔

不同同步任务可以有不同的轮询间隔与 Cron。把核心表与报表表分开建同步任务，比全局调小间隔更可控。

7.4 安全滞后与漏数判定

时间戳游标读取的是「高水位减去安全滞后」之前的行。默认滞后 1 秒，用来覆盖那些已经在事务中写入、但提交时间略晚于其他事务的变更。滞后太小可能漏行，太大则增加延迟。

- **漏行迹象**行数审计持续报警、目标表行数少于源表、业务反馈旧数据未更新。
- **排查顺序**先确认游标列是否被回填或时区不一致，再检查安全滞后是否被调成 0。
- **补救**停止同步任务，重置进度并重新全量；仅补某几张表时，可在同步任务中重新勾选对象后重建。

不要用轮询追删除

轮询看不到 DELETE。需要同步删除时，用 FULL_COMPARE（有主键的小表，并开启「同步行删除」）或切换到日志 CDC。用「目标库定期清空重建」来模拟删除，代价远高于 CDC。

v1.8.2 起，行数审计发现缺口时先尝试定点修补：按主键把区间切段、只对比计数偏短的段，再只回源取缺失的行幂等写入，游标保持不动，一次修补只记一条变更日志。缺口超过 `sync.row-count-repair-max-rows`（默认 1000 行）、表没有主键或游标列无法在目标侧比较时，仍退回清空游标整表重载。删除比对超过内存上限时也可以改为按主键键序分页比较（`sync.delete-compare-max-rows`，默认 0 关闭），内存从「整表所有键」降到「一页键」，代价是每轮一次目标全表扫描与逐键回源确认。

7.5 何时切换到日志 CDC

信号	说明
需要看到 DELETE	轮询无法感知删除；有主键的小表可开「同步行删除」后用全表比对，长期方案是 CDC
延迟要求进入秒级以内	轮询下限就是间隔，无法再低
源库无法承受频繁轮询	CDC 读日志，对业务表查询压力更小
变更量大且集中	CDC 按日志顺序回放，避免大窗口查询
源库具备日志能力	否则先做源库改造，不能直接切换

切换前先完成 6.2 的源库准备，再在同步任务页切换策略。切换会自动重建基线，期间目标库会有一次全量写入，应安排在维护窗口内。

轮询与日志 CDC 对照

维度	轮询	日志 CDC
延迟下限	等于轮询间隔	秒级
删除感知	不支持（小表可开「同步行删除」+ 全表比对）	支持
源库侵入	仅只读查询	需开启日志与复制/挖掘权限
运维关注点	游标列、索引、轮询间隔	位点、日志保留、租约与回放延迟
适用场景	报表、弱实时、无删除同步	实时同步、需要同步删除与更新

> **可以混用**
> 同一套部署里，核心实时表用 CDC，报表表继续轮询。两种模式共享元数据库与看板，互不影响；切换只作用于单个同步任务。

CHAPTER 08

ETL 任务与批次运维

同步任务保证源与目标持续一致，ETL 任务只按操作者写的 SQL 取数，再把结果写入目标表。两者共用控制台与元数据库，但调度、连接、进度和失败语义完全不同。本章讲生产环境里的调度、观测与排障；页面上的新建与维护操作见《用户操作手册》第 8 章。

8.1 ETL 与同步任务的分工

维度	同步任务	ETL 任务
目标	结构与数据持续一致，删除与 DDL 都会跟随	按取数 SQL 复制结果行；不比对两侧，不保证与源表一致
调度	项目级轮询或日志 CDC，运行在同步引擎的线程池	独立 cron，运行在 ETL 专用线程池
连接	任务内共享连接池，受 `sync.max-pool-size` 限制	单表任务每次运行打开一对连接；批次内所有成员共用一对
进度	游标或 CDC 位点，存在 `sync_progress` / `cdc_position`	断点续跑位置，存在 `etl_checkpoint`
失败	按调度周期重试，可进入内置告警	记录失败，不自动重试；等下一次 cron 或手动运行
目标结构	自动建表并持续同步结构	只在目标表不存在时按结果列建一次，之后不再改
元数据	`project` / `sync_progress` / `cdc_position`	`etl_batch` / `etl_task` / `etl_checkpoint` / `etl_run`

ETL 不接入同步引擎

ETL 不读日志、不写 CDC 位点，也不会触发同步任务的重试；同步任务的结构同步同样不会修改 ETL 已建好的目标表。两套任务只在元数据库和控制台层面共用。

内置告警目前不覆盖 ETL

告警分类只有同步失败、轮询卡死、项目未运行、CDC 停摆、数据漂移、连接健康与 PG slot 积压七类，ETL 失败不在其中。ETL 巡检需要靠页面上的最近运行状态、`etl_run` 或外部监控。

8.2 调度与线程池

ETL 使用自己的调度器和执行器，与同步引擎的 Quartz 线程池隔离：一条执行数小时的 ETL 不会占用轮询或 CDC 的线程。两组线程的规模固定，不能通过配置调大。

线程池	规模	作用
`jync-etl-trigger-*`	2	只负责到点触发 cron，不执行取数
`jync-etl-run-*`	核心 2 / 最大 2，队列 50	真正执行 ETL 任务或批次

- **启动时注册**应用启动后加载所有启用的单表任务与批次，按 cron 注册触发；停用或修改 cron 后自动重新注册。
- **同一目标不叠加**触发时若任务或批次还在运行，本次触发直接跳过并写一条 INFO 日志，不会排队等待，也不会启动第二份。
- **同时最多两个**执行器只有两个槽位，其余运行请求进入队列（最多 50 个）；队列满时提交被拒绝，日志写 WARN，本次触发不会补跑。
- **错过的触发不补**cron 按墙钟触发，不会像消息队列那样回放。上一轮跑过了下一个时间点，就等再下一个时间点。

> **不要把大批次排在同一个整点**
> 多个批次同时到点，只有一两个能真正运行，其余在执行队列里等待，页面看起来像「没有按时跑」。需要串行的大批次请把 cron 错开，或合并成一个批次按顺序执行。

8.3 批次：共享连接与顺序执行

同源同目标的一组表建议放进一个 ETL 批次。批次持有唯一一条 cron 和一对源 / 目标连接，成员按创建顺序逐个执行；单个成员失败或被占用不会中断后面的成员。批次创建后源 / 目标连接锁定，需要换库时新建批次。

运行情况	判定
全部成员成功	批次最近运行记为 `SUCCESS members=N read=… written=…`
有成员失败或被跳过	记为 `PARTIAL members=N ok=… failed=… skipped=…`，即使没有失败也不算成功
成员正在被手动运行	该成员被跳过（skipped），其余成员照常执行，下一轮再补
批次本身还在运行	新的触发或「运行批次」被拒绝，日志写「still running; the scheduled run was skipped」



图 8–1　ETL 批次：一个批次共享一对源 / 目标连接与一条 cron

- **顺序执行**批次内不并行，几十张大表可能跑很久；需要并行请拆成多个批次，但全机同时最多只有两个 ETL 执行槽。
- **目标表去重**同一批次内两个成员不允许写同一张目标表，避免顺序执行时互相覆盖。
- **成员不单独调度**成员的启用状态与 cron 由批次统一管理；在成员页面只能「立即运行」，单独启停会被拒绝。
- **移出与删除**移出成员会清空它的检查点；移出最后一张表时批次自动停用。删除批次会连带删除成员任务与检查点，运行历史保留。

> **互斥只在一个进程内**
>
> 批次与任务的「正在运行」判断是进程内标记。多实例部署且多个节点都启用了同一个 ETL 批次时，它们可能同时执行并写同一张目标表。ETL 批次应只在一个节点启用，或把实例拆成「同步节点」与「ETL 节点」。

8.4 断点续跑与检查点

只有「追加」和「范围覆盖」写入方式、且配置了断点续跑 key 的任务才有检查点。每写入 500 行并提交目标库后，Jync 把最后一条记录的 key 写入 `etl_checkpoint`；下次运行从该 key 之后继续。覆盖模式每轮先清空目标表，不保留检查点。

操作	对检查点的影响
任务运行失败	保留已提交批次对应的检查点；下次从最后一个已提交位置继续，已提交的行不重复写
修改取数 SQL、源 / 目标、目标表、写入方式、范围条件或 resume key	检查点被清空，下次从结果集开头重来；追加模式下可能产生重复行
切到覆盖模式	每次运行前清空检查点，目标表先清空再写
移出批次成员，或删除任务 / 批次	检查点一并删除

操作	对检查点的影响
从旧备份恢复元数据库	检查点回到备份时点，之后写入的行会被再次读取和写入

> **续跑语义是「至少一次」**
>
> 目标库已提交但检查点尚未落库时进程崩溃，重启后可能重写最后一批。目标表没有主键或唯一约束时会出现重复行；对重复敏感的表请建唯一索引，或改用覆盖 / 范围覆盖。

8.5 运行历史与故障排查

每次运行都会在 `etl_run` 留一条记录（任务、成功与否、影响行数、时间），看板的 ETL 趋势线由此绘制；`etl_task` 只保留最近一次结果。删除任务不会删除历史行，`etl_task_id` 允许为空。当前没有内置的历史清理，长期运行需要定期归档或清理 `etl_run`。

任务页的最近结果只有两种形态：成功为 `SUCCESS read=… written=…`，失败为 `FAILED side=SOURCE|TARGET|PROGRAM error=…`。批次结果是 `SUCCESS`，或带 `ok / failed / skipped` 计数的 `PARTIAL`。

日志关键字	含义	处理方向
`ETL task … scheduled with cron`	任务已注册触发	没有这条说明 cron 为空、任务被停用，或它是批次成员
`still running; the scheduled run was skipped`	上一次还没结束，本次触发跳过	看上一轮是否卡在慢 SQL、锁等待或大表
`could not be started`	执行队列满或执行器正在关闭	错开 cron，减少同时到点的大批次
`ETL task … copied … rows into … (… read)`	一次成功运行及读写行数	读多写少说明目标端有过滤或合并，核对写入方式
`failed on the source side`	读取源库失败（SQL、权限、连接）	先在源库执行同一条 SQL，再查账号权限与网络
`failed on the target side`	目标库建表、删除或写入失败	查目标表权限、主键 / 唯一约束、字段类型与锁等待
`failed in the program`	配置或程序侧失败（缺连接、resume key 不合法等）	按页面错误提示修正配置后重跑
`failed inside batch`	批次某成员异常退出，已记一条失败历史	打开该成员的最近结果，按 side 定位

现象	常见原因	处理
到点没有运行	任务 / 批次未启用；cron 为空或非法；把批次成员当成有独立 cron 的任务	确认启用状态与批次 cron；成员只看批次
触发了但页面一直「运行中」	SQL 长时间不返回、目标库锁等待、网络挂起	查源 / 目标库会话与锁；必要时在数据库侧终止会话
批次显示 PARTIAL、skipped 不为 0	成员被手动运行占用，或上一轮同成员尚未结束	等当前运行结束；避免 cron 与手动运行重叠
目标表结构与预期不符	目标表已存在，Jync 不再改结构	按需手工改表或换目标表；结构演进仍由同步任务负责
追加模式出现重复行	修改取数定义导致检查点清空；恢复旧元数据；崩溃发生在提交与写检查点之间	按 resume key 去重，或改用覆盖 / 范围覆盖

CHAPTER 09

监控、日志与告警

Jync 没有引入独立的监控组件，可观测性来自四处：控制台看板、数据库与同步任务性能页、日志文件，以及可选的 Prometheus 指标端点。把这几处的关键信号接入现有监控体系，就能覆盖绝大多数故障。

9.1 日志文件与轮转

项	默认	说明
文件	`./logs/jync.log`	相对工作目录
单文件上限	10 MB	超过即滚动
保留天数	14 天	按天滚动保留
总量上限	500 MB	超过后删除最旧历史
启动清理	开启	清理超限历史与异常退出残留的临时文件

三重上限是 v1.7.7 的默认行为。旧版本只按天数保留且没有总量限制，长期运行会把日志目录撑到数十 GB；升级后自动生效，无需额外配置。

9.2 关键日志与判读

关键字	含义	处理方向
`rows/s=`	分块吞吐统计	与基线比较，判断调优效果
`StopRequestedException`	收到停止请求，正常中断	不是错误，确认同步任务已按预期停止
`LockLostException`	执行锁被其他节点接管	检查节点存活与时钟，避免双跑
`has made no progress for`	一轮超过 `sync.stuck-threshold-ms` 没有推进（同一轮只告警一次）	调度已冻结，等待不会恢复；确认后重启本实例。强制解锁只清锁行，不结束线程
`cannot see every object in the schema`	源库清单缺项，但账号只能看见部分对象，无法区分删表与授权收回；目标对象已保留	以表属主或库级授权同步后，源端删表才会联动；每个同步任务每类对象只告警一次

关键字	含义	处理方向
`SQLException`	数据库返回错误	看完整 SQL 与错误码，区分权限、超时、语法
`reconnect`	CDC 断线重连	短暂抖动可忽略，持续出现查网络与源库日志
`audit`	管理操作审计	用于追查配置变更与启停操作

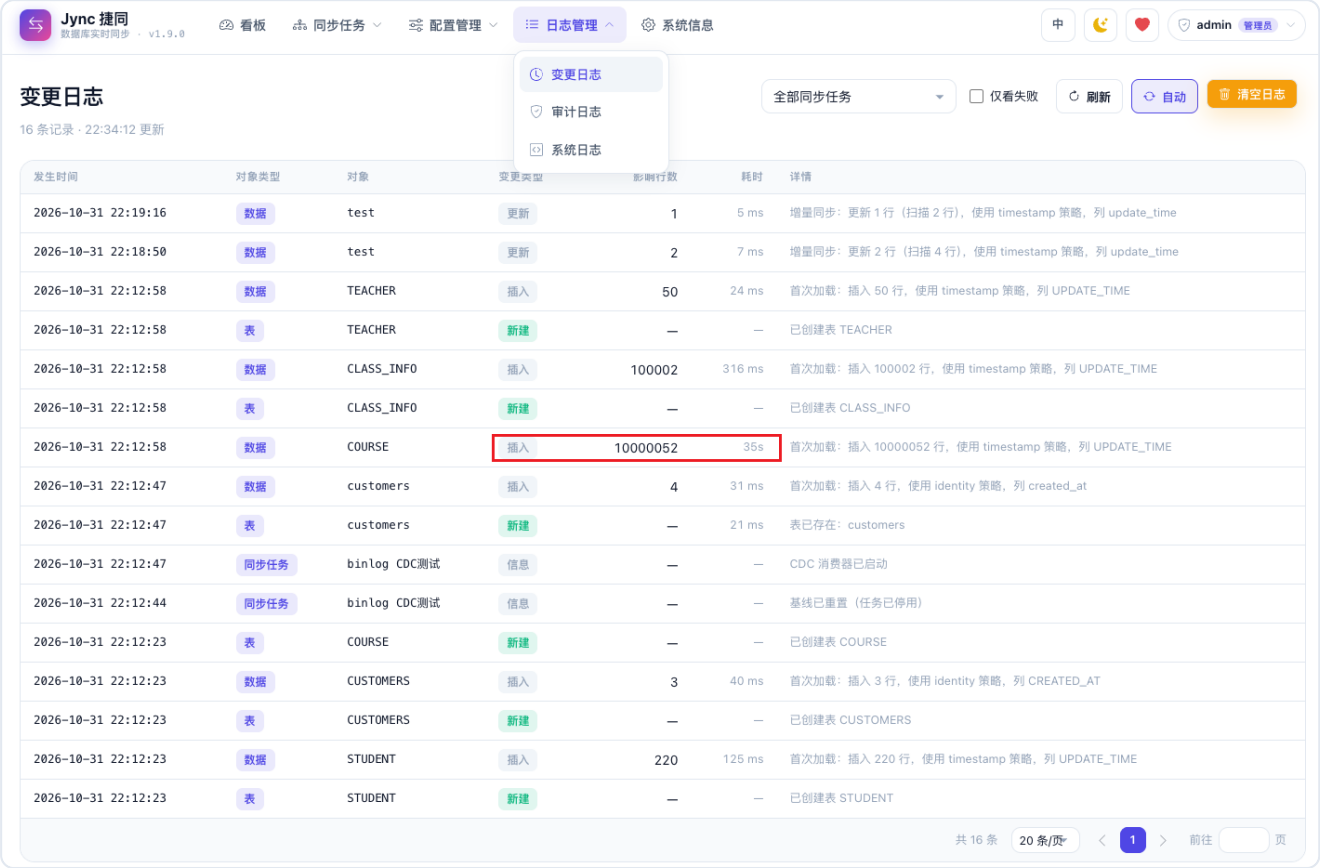


图 9–1　控制台日志：按同步任务与时间查看同步过程与异常

```
# 常用检索
grep -n "ERROR\|WARN" logs/jync.log | tail -50
grep -n "rows/s=" logs/jync.log | tail -20
grep -n "table=ORDERS" logs/jync.log | tail -20
grep -n "LockLost\|StopRequested" logs/jync.log | tail -20
```

9.3 指标采集与探活

- **探活**应用未引入 actuator，使用 `GET /login`；返回 200 代表 Web 层与元数据库可用。Kubernetes 清单已内置 startup / liveness / readiness 三个探针。
- **进程**监控 Java 进程存活、CPU、堆使用与线程数；堆持续接近上限时先看是否有大表全量在跑。

- **数据库**监控源库与目标库的连接数、慢查询、锁等待；CDC 同步任务还要监控日志空间与复制槽。
- **磁盘**监控安装目录与日志目录；日志有 500 MB 上限，但元数据库会随同步任务、变更日志与指标样本增长。
- **业务**用行数审计与抽样比对作为最终校验，监控指标正常不代表数据一定一致。

内置性能页

数据库性能页与同步任务性能页直接读取本地样本，不需要额外组件。数据库采样默认 5 分钟、同步任务 1 分钟、容量 1 小时，历史保留 7 天；采样开关与间隔见 3.7 节。

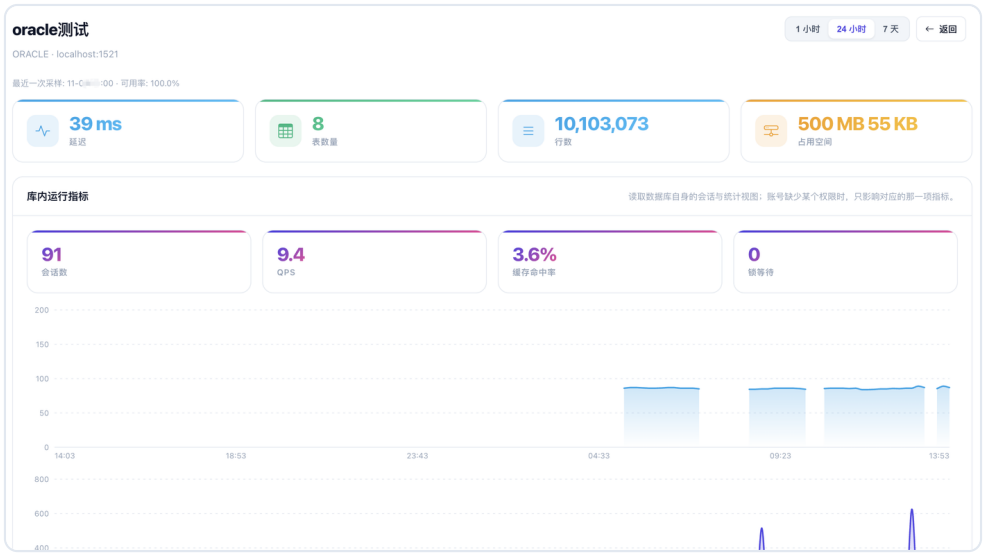


图 9–2　数据库性能：连接延迟、会话 / QPS / 缓存命中率 / 锁等待与容量

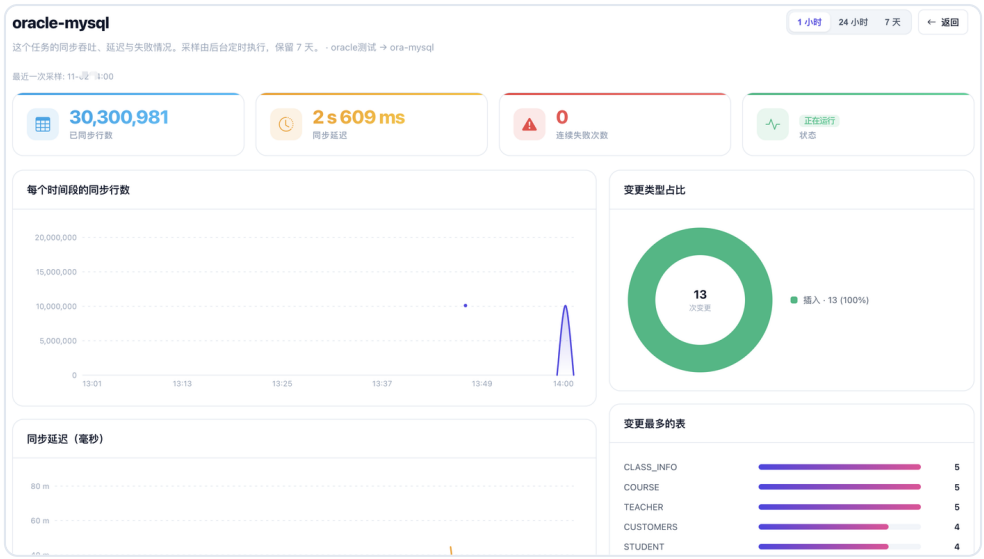


图 9–3　同步任务性能：同步行数、延迟、失败次数与变更构成

Prometheus 端点

配置 `metrics.prometheus-token`（或环境变量 `JYN_METRICS_TOKEN`）后，抓取 `GET /api/metrics/prometheus` 并带上 `Authorization: Bearer <token>`，即可拿到进程、任务与连接三组最新样本。端点输

出的是采样表中已落库的样本，不会在被监控库上触发额外查询；应让抓取间隔不小于采样间隔，否则同一批样本会被重复抓取。

```
# prometheus.yml 片段
scrape_configs:
  - job_name: jync
    scrape_interval: 60s
    authorization:
      credentials: <JYN_METRICS_TOKEN>
    static_configs:
      - targets: ["jync.example.com:8080"]
```

未配置 token 时端点不存在

token 留空时 `/api/metrics/prometheus` 返回 404，这是默认关闭状态，不是配置错误。启用后只接受 Bearer token，抓取账号与业务账号应使用不同的网络边界。

9.4 告警建议

下表用于已经接入外部监控（Prometheus、Zabbix、云监控）的部署。Jync 1.9.0 起还内置 Webhook 告警，可在「配置管理 → 告警通知」把运行异常直接推到钉钉、企业微信、飞书或自建接口。两者不冲突：内置告警负责业务侧事件，外部监控负责进程、磁盘与主机指标。

告警条件	级别	动作
进程退出或探针连续失败	紧急	查日志与元数据库锁，确认是否被重复实例抢占
CDC 落后秒数持续 10 分钟上涨	重要	查目标库写入压力与源库日志读取
位点 5 分钟未推进且源库有变更	重要	查租约、连接与错误日志
行数审计报差异	重要	确认差异表，排查漏行或目标库被外部修改
ERROR 日志在 5 分钟内超过阈值	重要	按错误类型聚合，优先处理持续出现的
磁盘使用超过 80%	提醒	清理旧备份、检查日志与元数据库增长
源/目标库连接数接近上限	提醒	降低全量并行度或调整连接池

内置告警覆盖的七类事件

同步失败、轮询卡死、项目未运行、CDC 停摆、数据漂移、连接健康、PG slot 积压。它只读取已有运行状态，不写任务、不改位点、不触发重试；同一事件在重复提醒间隔（5–1440 分钟）内不重复发送，故障恢复时补发一条恢复消息。进程退出、磁盘水位与主机指标仍需外部监控。

9.5 Webhook 告警运维

1.9.0 起，应用内置 Webhook 告警：管理员在「配置管理 → 告警通知」启用后，Jync 按固定节奏检查已有的运行状态，把命中的异常 POST 到钉钉、企业微信、飞书或自建接口。告警只读，不写任务、不改位点、不触发重试。



图 9–4　告警通知：Webhook 地址、消息格式、分类与重复提醒间隔

配置与存储

配置项	行为
开关	关闭时不评估规则、不产生任何外呼，系统行为与未配置时一致
Webhook 地址	只接受 `http` / `https` 且必须带主机名，最长 2048 字符；AES 加密后存 `alert_config`，页面只回显尾号。编辑其他项时地址留空即保留原地址，清除地址有单独开关
消息格式	通用 JSON（默认）/ 钉钉 / 企业微信 / 飞书，按接收端选择
告警分类	多选；启用时至少要选一类。历史行没有分类字段时按全部类别处理
重复提醒	5–1440 分钟，默认 30；同一条件在间隔内不重复发送

设置只有一行

`alert_config` 是单行设置（`id = 1`）；`alert_state` 按「分类 + 任务 + 对象」记录是否已通知、首次发现与最近发送时间。两个节点共享同一份状态，冷却起点因此一致。

评估节奏与外呼行为

- **评估间隔**启动后 45 秒做第一次评估，之后每次评估结束再等 60 秒；可用 `sync.alert.initial-delay-ms` 与 `sync.alert.evaluate-interval-ms` 调整。开关关闭或地址为空时本轮直接返回。
- **发送不阻塞**评估线程只负责生成消息，真正 POST 在单线程 `jync-alert-webhook` 上排队；连接与请求超时都是 3 秒，失败不重试，也不持久化队列。
- **2xx 才算成功**「发送测试消息」同步发送并立即返回结果；非 2xx、超时、证书错误都按失败处理。
- **地址不落日志**URL 本身是凭证，日志从不打印完整地址；非 2xx 只在 DEBUG 记状态码，连接失败只记异常类名。
- **不保证送达**接收端不可达、被限流或应用在队列清空前重启，消息就丢了；关键异常仍需外部监控兜底。

七类事件与恢复语义

告警分类	判定条件	恢复消息
同步失败	同一张表连续多轮同步失败	有
轮询卡死	一轮超过 `sync.stuck-threshold-ms` 没有进展；该值设 0 表示关闭	有
项目未运行	已启用的轮询项目没有注册调度，且当前不在运行	有
CDC 停摆	消费者因致命错误停驻，或长时间断开	有
数据漂移	行数审计发现目标少于源，并已开始修补	无（事件类）
连接健康	连接连续多次健康探测失败	有
PG slot 积压	PostgreSQL 逻辑复制槽滞留的 WAL 超过阈值	有

> **条件类与事件类**
> 条件类在出现时发一次，持续期间每过一个重复提醒间隔再发，恢复时补发一条恢复消息；数据漂移是事件类，每条新事件只发一次，没有恢复消息。判定阈值仍由各自子系统持有（例如卡死阈值、PG WAL 上限、审计周期），告警页不提供设置。ETL 失败不在七类之中。

报文格式

格式	请求体
通用 JSON	`{"event":"jync.alert","text":"…"}`
钉钉	`{"msgtype":"text","text":{"content":"…"}}`
企业微信	与钉钉相同的报文结构
飞书	`{"msg_type":"text","content":{"text":"…"}}`

请求头固定为 `Content-Type: application/json; charset=UTF-8`，正文中的引号、换行与控制字符会被转义。自建接口按通用格式解析即可。

投递失败排查

现象	常见原因	处理
「发送测试消息」失败	DNS / 端口 / 证书 / 代理 / 3 秒超时 / 接收端返回非 2xx	在服务器上用 `curl -m 5` 向同一地址 POST 一条测试报文，逐层排除网络与 token
测试成功，真实告警收不到	开关关闭、未选分类、当前没有命中事件，或仍在重复提醒间隔内	逐项核对；临时把间隔调到 5 分钟验证，确认后改回
某类事件从未出现过	判定条件未满足，例如卡死阈值设为 0、CDC 没停、PG 未超上限	按对应章节确认阈值与实际状态，不要只盯告警页
日志里看不到投递失败	失败只写 DEBUG，默认日志级别是 INFO	临时设 `logging.level.com.qqmu.jync.service.alert=DEBUG`，检索「Alert webhook answered HTTP」与「could not be reached」
短时间收到大量重复消息	重复提醒间隔过短、多个分类同时命中，或多节点同时评估	把间隔调到 30 分钟以上；多节点部署时接受极端竞态下的重复，不要用很短的冷却
HTTPS 握手失败	接收端使用内部 CA，JDK 信任库不包含该证书	把 CA 导入运行账号使用的 truststore，或改用受信任地址
接收端限流	官方机器人有频率限制，例如钉钉自定义机器人约 20 条/分钟	提高重复提醒间隔，只启用真正需要的分类

出网与安全

- **默认不出网**不填地址或关闭开关时，Jync 不产生任何告警请求；启用后由服务器直接向该地址发起 POST。
- **Webhook 是 SSRF 面**地址由管理员填写，服务端会主动请求它。只使用可信内网接口或官方机器人地址，不要填公网或不可信地址；不需要时保持关闭。
- **加密依赖全局密钥**地址与数据库口令使用同一套 `sync.crypto-password` / `sync.crypto-salt` 与密钥文件。密钥丢失或更换后地址无法解密，告警会静默停止，需要重新填写。
- **需要代理时显式配置**Java HTTP 客户端读取 JVM 的 `-Dhttps.proxyHost` / `-Dhttps.proxyPort`，不自动继承 shell 的 `HTTPS_PROXY`；内网出口受限时在启动参数里补上。
- **正文含业务标识**告警正文包含任务名、表名与错误摘要，会离开本机；接收端必须与数据敏感级别匹配。

9.6 审计与留痕

管理操作写入 `audit_event`，数据与结构变更写入 `change_log`。审计记录用于回答「谁在什么时候改了什么」，变更日志用于回答「哪张表在哪个时间窗口同步了多少行」。

> **保留策略要匹配合规要求**
> 元数据库会随变更日志增长。若合规要求长期保留，应定期导出 `audit_event` 与 `change_log` 到归档库，而不是无限期留在业务库。

9.7 系统日志页面

1.8.0 起，应用日志可以在控制台的「日志管理 → 系统日志」直接检索，所有登录用户都可访问。值班人员不必先登录服务器，就能判断故障发生在源库读取、目标库写入还是锁与连接层。

能力	行为
文件范围	只读日志目录中由滚动策略生成的白名单文件；文件名白名单加真实路径校验，拒绝 `../` 与符号链接逃逸
级别	DEBUG / INFO / WARN / ERROR，或「全部」
关键字	不区分大小写模糊匹配，命中片段高亮；日志文本逐段转义后再插入标记
读取上限	单次最多扫描 16 MB；普通文件从尾部读，gzip 归档从头部解压读；返回最近 1000 条匹配行并提示是否截断
编码	写入与读取都固定 UTF-8，不受服务器区域设置影响

> **页面检索不是日志采集**
> 16 MB 与 1000 条是交互式排障的边界，不替代集中日志平台。需要长期留存、跨节点聚合或告警时，仍应把 `logs/` 接入现有采集链路。

> **文件权限与白名单**
> 系统日志页只认日志目录内的滚动文件。若修改了 `logging.file.name`，确认新路径仍在应用可读范围内且属主正确；否则页面会提示「日志文件不可用」，但应用本身仍正常写日志。

9.8 诊断包与支持取证

需要厂商支持时，不必手工收集日志、配置与版本信息：管理员打开「设置 → 诊断包」，点「下载诊断包」，浏览器会下载 `jync-diagnostics-YYYYMMDD-HHmmss.zip`。导出动作写入审计日志（动作 `DIAGNOSTICS_EXPORT`，记录条目数与操作者）。

条目	内容	用途
`README.txt`	包说明、脱敏边界与大小上限	转交前给接收方看的处理提示
`manifest.json`	其余每个条目的文件名、大小与 SHA-256	校验传输中是否被改动
`summary.json`	版本、生成时间、项目数、连接数、日志行数	第一时间确认现场版本
`config/sync-settings.json`	轮询、批次、全量、锁、超时、校验等运行参数	复核是否被改过参数
`config/app-settings.json`	版本、端口、时区、语言、profile	排除环境差异
`connections.json`	连接的产品类型、方言族、角色、主机、端口、库 / schema、用户名	判断驱动与网络路径
`projects.json` 、`projects/{id}/`	同步任务的配置摘要、各表进度、CDC 位点摘要	定位卡在哪张表、哪个位点
`changelog/recent.json` 、`audit/recent.json`	最近 200 条变更记录与管理员操作审计，详情截断 500 字	还原最近发生了什么
`logs/jync.log.tail.txt`	日志尾部，最多 2000 行 / 2 MB	看异常栈与告警上下文
`jvm.json`	JVM 版本、堆、线程与系统属性摘要	排除资源与版本问题

脱敏是尽力而为

包内不含密码与 `enc:` 密文、API Key、Webhook 地址、带账号信息的连接串、自定义驱动路径、游标原值、CDC 布局签名、`jync.key` 、`data/` 与原始 `application.yml` 。日志、变更详情与审计详情仍会经过正则脱敏，但业务标识、SQL 片段与日志正文可能保留。转交外部支持前按组织的数据外发流程审批，并把压缩包当作敏感文件保管。

上限

单 JSON 条目 1 MB、日志尾部 2 MB、整包压缩前 10 MB。超过 10 MB 时接口返回明确错误，不会生成残缺包；先清理超大日志或减少变更记录再导出。

9.9 内容级校验的运维开销

内容级校验在每张表增量同步成功后按主键抽样比对行值，默认每 15 分钟最多一次、每表 50 行、每个项目每周期只校验 1 张表。它给源库与目标库各带来一次按主键的批量读，不会全表扫描：抽样通过分桶取键，查询带游标上界与语句超时。

顾虑	实际行为与建议
大表成本	源行数估计超过 `content-audit-max-rows`（默认 200 万）直接跳过；确实需要校验超大表时再调高，并配合加大间隔
无索引表	游标列没有前导索引且表较大时跳过，受 `max-unindexed-scan-rows` 保护，避免退化成全表扫描
误报	LOB / JSON / XML / 几何列不参与比较；数字、时间、布尔先规范化再比，时间按墙钟对齐。目标端若在同步之外改写数据，差异是真实的，属于需要排查的外部修改
告警噪声	同一任务、同一张表、同一组差异列在短时间内只记一次；恒等游标（IDENTITY）看不到已有行更新，这类表建议改用时间戳游标或 CDC
完全关闭	`sync.content-audit-interval-ms=0`；管理员仍可对单表手工触发「校验内容」

与行数审计的分工

行数审计发现「目标比源少」，内容级校验发现「行数相同但值不同」。两者都只报告不修复；行数缺口在阈值内可走定点补行，内容差异必须人工判断来源。

CHAPTER 10

专属迁移运维

专属迁移是独立于实时同步与 ETL 的第三条链路。运维上要记住三点：它有自己的连接与三张元数据表；任务只在手动或 cron 触发时运行；失败重跑没有断点，按写入方式决定是否先清理目标对象。

10.1 组件与数据模型

表	内容	丢失后果
`migration_connection`	专属连接的产品类型、端点、库 / schema、额外参数与加密凭据	需要重建全部专属连接
`migration_task`	任务类型、源 / 目标连接、对象范围、选项与 cron	需要重建专属任务
`migration_run`	每次运行的状态、触发方式、计数、错误摘要与时间	运行历史丢失，任务与连接不受影响

三张表由 `ddl-auto=update` 首次启动自动创建，随 `./data` 一起备份。删除任务会一并清理它的运行记录；连接仍被任务引用时不能修改产品类型，删除前需先删除引用它的任务。

10.2 连接与凭据

- **凭据存储**：密码 / API Key 只存加密字段，页面不回显；MongoDB URI 与 Kingbase JDBC URL / 额外参数中禁止写密码（Kingbase 还禁止 user / password 参数），保存与测试时直接拒绝。
- **驱动**：MongoDB 使用随包内置的原生驱动（约 2.2 MB），不需要另配 jar；MongoDB（Atlas SQL）走 JDBC 的旧路径仍需自备厂商驱动。KingbaseES 驱动随包提供。
- **最小权限**：搜索连接需要索引读、写入目标索引与别名管理权限；MongoDB 需要 listCollections、createIndex、insert 与目标集合的 drop 权限；KingbaseES 需要目标 schema 的建表与写入权限。
- **TLS**：搜索连接沿用 `verifyTls`；MongoDB 的 TLS 由 URI 或额外参数控制，KingbaseES 由驱动参数控制，这两类连接不显示 `verifyTls` 开关。
- **日志**：测试连接与运行日志不输出 URI、密码或 API Key，只保留产品、版本与错误摘要。

10.3 调度、运行与恢复

- **触发方式**：手动运行与 cron 定时。启用 / 停用任务后立即收敛调度，不需要重启。

- **并发**：同一任务在单个实例内只有一个执行槽；定时触发时若上一轮仍在运行，本轮直接跳过。专属迁移没有跨实例分布式锁，多实例共用同一元数据库时不要让多个实例同时承载定时任务，或统一使用 OVERWRITE 收敛重复写入。
- **取消**：取消在批间检查，已写入的对象保留；正在提交的一批会完成后再停。
- **重启**：实例启动时把遗留的 RUNNING 记录标记为 INTERRUPTED，不会自动续跑。
- **历史**：每个任务保留最近 50 次运行，含触发方式、对象计数与错误摘要。

10.4 监控与排障

现象	原因与处理
连接测试失败，提示 URI 含密码	MongoDB URI 或 Kingbase URL 中写了密码；把密码移到密码框，连接串只保留主机、端口与库名
预检报「源与目标为同一数据库」	同库复制会被拒绝，避免 OVERWRITE 先删目标；跨库复制请建两条不同 `databaseName` 的连接
FAIL_IF_EXISTS 预检失败	目标已存在同名索引 / 集合 / 表；改用 OVERWRITE，或先手工清理目标对象
MongoDB APPEND 出现重复键告警	这是预期行为：_id 冲突跳过并告警，不覆盖目标文档；需要以源为准时改用 OVERWRITE
KingbaseES 连接即失败	确认版本不低于 V9R1C1B25；V8R3 缺少 PG_CATALOG 兼容视图，与内置 9.x 驱动不兼容
运行失败后目标留下部分数据	没有跨对象事务；用 OVERWRITE 重跑收敛，或按失败对象手工清理后重跑
任务一直未按 cron 运行	确认任务已启用且 cron 合法；上一轮仍在运行时会跳过本轮，查看运行历史与日志

排障时先看运行历史的错误摘要与同一时间段的 `logs/jync.log`；连接类问题用「测试连接」复核，权限类问题按连接类型检查源 / 目标账号的最小权限。

10.5 限制与升级

- **一次性复制**：只复制运行那一刻的数据，没有断点续传；需要持续跟随变化请使用实时同步或 ETL。
- **对象边界**：view 与 timeseries 集合不复制；分片 key、用户 / 角色 / 认证信息不迁移；MongoDB 普通索引不会翻译成 SQL 索引；GridFS 按普通集合复制。
- **嵌套数据**：MongoDB → KingbaseES 只把文档与数组写成 TEXT 的 JSON，不展开为列。
- **升级**：v1.12.0 不新增专属迁移表，也没有破坏性变更，替换 `jync.jar` 重启即可，容器部署把镜像 tag 更新为 `jync:1.12.0`。

CHAPTER 11

升级与回滚

Jync 的升级很轻：替换同名 JAR，重启，元数据表结构自动演进。真正决定升级是否安全的，是升级前的备份、升级后的验证，以及出问题时能否快速回到旧版本。

11.1 版本检查与一键升级

版本检查有两条路径：版本卡的后台检查优先访问 GitHub，失败或超时自动改查 Gitee；管理员页面的同步检查同样优先 GitHub，再查 Gitee。成功结果缓存 10 分钟，失败缓存 1 分钟。单次请求超时默认 5 秒。

场景	行为
发现新版本	系统信息右上角出现提示，管理员可进入升级入口
版本号相同、安装包被重新发布	再比对一次发布制品：GitHub 用附件 SHA-256，Gitee 退化为下载地址 `HEAD` 取 `Content-Length`。确认不同才显示「同一版本号的安装包已重新发布」并允许同版本一键升级；证明不了差异时按已是最新处理。管理员可「忽略此提示」，发布方再上传新文件后自动重新提醒；`app.update-check.match-artifact=false` 可关闭比对
一键升级	下载并校验新 JAR，改名替换，旧版本保留为 `jync.jar.bak`；新进程启动失败时可用备份回滚
匿名 API 限流	平台按客户端 IP 限制匿名调用，超限返回 403（Gitee 报 `Rate Limit Exceeded`），此时弹窗提示「拒绝了匿名请求……请切换下载源，或稍后重试」。升级仍严格使用所选下载源，不会静默换源；版本卡的后台检查照常 GitHub 优先、失败回退 Gitee
下载失败	保留当前版本运行，日志记录失败原因；可改用离线升级
纯内网	两个平台都不可达，建议关闭更新检查，入口与提示一并隐藏

> **一键升级仍要备份**
> 自动升级只负责替换程序，不负责备份数据。升级前应确保 `data` 目录与密钥有可用备份，并记录当前版本号。

11.2 离线手工升级

1. **停止服务**
   确认没有同步任务在运行，再停止应用。
2. **备份**
   复制 `data` 目录与当前 `jync.jar`，保留可回退版本。
3. **替换文件**

把新包重命名为 `jync.jar`，覆盖旧文件；注意文件属主与执行权限。

4 **启动并验证**

启动后访问登录页，检查系统信息中的版本号；查看日志确认没有 ERROR。

5 **抽查同步**

运行一次「立即同步」，确认变更日志正常写入，同步任务状态正常。

```
# 离线升级
systemctl stop jync
cp -a /opt/jync/data /backup/jync-data-$(date +%F)
cp /opt/jync/jync.jar /backup/jync-$(date +%F).jar
cp jync-1.9.0.jar /opt/jync/jync.jar
chown jync:jync /opt/jync/jync.jar
systemctl start jync
tail -f /opt/jync/logs/jync.log
```

11.3 回滚

1 **停止服务**

与升级相同的停机步骤。

2 **换回程序**

用备份的旧 JAR 覆盖当前文件。

3 **评估元数据**

新版本可能已经演进表结构。旧版本通常能打开新结构；若启动失败，使用升级前的 data 备份整体恢复。

4 **启动验证**

确认版本号、同步任务状态与连接可用，再恢复同步。

> **回滚会丢失升级后的配置变更**
>
> 整体恢复 data 备份意味着升级后新增的同步任务、连接与进度都会回到备份时点。回滚前先导出或记录这些变更，避免二次返工。

11.4 升级检查清单

- ✓ 当前版本号、JDK 版本与元数据形态已记录。
- ✓ `data` 目录与密钥文件已完整备份，并验证备份可读。

- ✓ 当前 JAR 已另存，回退路径明确。
- ✓ 升级窗口已通知业务方，CDC 同步任务停机时长不超过日志保留时间。
- ✓ 升级后登录页正常，系统信息版本号正确。
- ✓ 日志无持续 ERROR，同步任务列表与连接测试正常。
- ✓ 至少一个同步任务完成一次手动同步，变更日志有记录。
- ✓ CDC 同步任务的位点与落后秒数正常，无跳过计数增长。
- ✓ 如启用告警：配置页的地址仍显示尾号，点一次「发送测试消息」确认接收端可达；不需要告警时确认开关为关闭。
- ✓ ETL 批次的启用状态与 cron 保持升级前配置；升级前确认没有 ETL 正在运行，避免重启时留下未结束的批次。
- ✓ 如使用专属迁移：升级前确认没有迁移任务正在运行；升级后专属连接测试正常，运行记录中不应出现新的 INTERRUPTED。

11.5 兼容性注意事项

项	说明
包名	发行包固定为 `jync.jar`，版本只体现在标签与页面信息中
元数据演进	`ddl-auto: update` 自动加表加列，无需手工迁移
加密密钥	从旧版本升级保持原值，不要改成新默认值
旧品牌数据	从 SyncTool 1.2.x 及更早版本升级，旧库文件会自动改名接管
会话	升级重启会使控制台会话失效，需要重新登录
浏览器缓存	静态资源每次带条件请求，通常无需强制刷新；异常时清除缓存重试
v1.8.0 新增表	`oa_auth_config` 由 `ddl-auto=update` 首次启动自动创建，并随 `./data` 一起备份；`auth_mode` 列名避开 Oracle / 达梦 / 崖山 / 神通等库的保留字
v1.8.0 默认行为	外部认证默认关闭，访问 `/login` 仍 302 到本地账号表单；旧书签与脚本不受影响
v1.8.0 旧地址	`/oa/settings` 保留为 302 → `/external-auth`（仅 GET）；旧表单字段已改名，POST 不兼容
v1.8.0 新增环境变量	`JYN_RESET_LOGIN`（应急恢复，平时不生效）、`JYN_COOKIE_SAME_SITE`（默认 lax）、`JYN_TRUSTED_PROXIES`（网关头模式的可信代理列表）
v1.8.0 界面前端	导航按「配置管理」「日志管理」分组，界面统一把「项目」改称「同步任务」；路由、数据表与 API 不变

项	说明
v1.8.1 结构	无数据库结构变更、无新增配置项；替换 `jync.jar` 重启即可。批量创建只在管理员打开「批量创建」页面时才读取源库目录统计
v1.8.2 结构	无数据库结构变更；替换 `jync.jar` 重启即可，容器部署把镜像 tag 更新为 `jync:1.8.2`。新增四个可选配置项，不配置即保持原行为
v1.8.2 日志 CDC	新增崖山 YStream 采集策略，达梦 DM8 与 PostgreSQL 方言家族、MySQL 兼容家族按能力探测放行；已配置的采集策略与位点不受影响，已有 CDC 任务升级后从原位点继续
v1.8.2 安装包	移除四个未使用的传递依赖（Bean Validation 校验器、AspectJ weaver、Quartz 的 c3p0 支持、Tomcat WebSocket），安装包从 84.0 MB 降到 79.6 MB；日志 CDC 所需的 zstd 解压与全部数据库驱动保留
v1.8.2 崖山源	崖山 23.x 没有 `DBMS_LOGMNR`，日志 CDC 请选 YStream；YStream 源侧服务由 Jync 托管，删除同步任务或切换策略时会尽力停止并删除
v1.9.0 结构	首次启动自动创建 `etl_task`、`etl_batch`、`etl_checkpoint`、`etl_run`、`alert_config`、`alert_state` 六张新表，其余表无结构变更；没有破坏性变更，替换 `jync.jar` 重启即可，容器部署把镜像 tag 更新为 `jync:1.9.0`
v1.9.0 默认行为	内置告警默认关闭，未填 Webhook 地址时不产生任何外呼；已有同步、CDC 与单表 ETL 任务的调度、位点与检查点不变，升级后按原计划继续
v1.9.0 数据库类型	内置类型扩到 34 种（含 7 种 ETL 分析型引擎与 MongoDB 原生读取），其中 25 种的驱动随包提供、9 种需自备官方 jar；已保存的连接配置无需改动
v1.9.0 安装包	安装包约 80 MB，日志 CDC 的 zstd 解压与全部内置驱动保留；新增的 ETL 与告警不引入额外外部依赖
v1.10.0 结构	首次启动自动创建 `app_metric_sample`、`db_metric_sample`、`task_metric_sample` 三张指标表，其余表无结构变更；没有破坏性变更，替换 `jync.jar` 重启即可，容器部署把镜像 tag 更新为 `jync:1.10.0`
v1.10.0 默认行为	性能采样默认开启，但只写本地元数据库、不外呼；Prometheus 端点默认关闭（token 为空时返回 404）。不需要采样时把 `metrics.enabled` 设为 `false`
v1.10.0 CDC	位点新增每张表的列序指纹。升级后已有 CDC 任务从原位点继续；若消费者停机期间源表执行过 DDL，重启会停在「列布局漂移」并提示重建基线，这是有意的保护而不是故障
v1.10.0 数据库类型	SAP HANA 支持实时同步；Doris、ClickHouse、Hive、Impala、Elasticsearch、MongoDB（Atlas SQL）可作为实时同步的源（仅轮询 / 全表比对，按无主键读取），不作目标。已保存的连接与同步任务无需改动
v1.10.0 安装包	安装包约 82 MB；指标采样与 Prometheus 端点不引入新的第三方依赖，指标表随 `./data` 一起备份
v1.11.0 结构	首次启动自动创建 `migration_connection`、`migration_task`、`migration_run` 三张专属迁移表，其余表无结构变更；没有破坏性变更，替换 `jync.jar` 重启即可，容器部署把镜像 tag 更新为 `jync:1.11.0`
v1.11.0 专属迁移	新增 MongoDB → MongoDB 与 MongoDB → KingbaseES；MongoDB 原生驱动随包内置，ES / OpenSearch → Easysearch 的选项与行为不变。专属任务只在手动运行或启用 cron 后执行

项	说明
v1.11.0 凭据	MongoDB URI 与 Kingbase JDBC URL / 额外参数中禁止写密码（Kingbase 还禁止 user / password 参数）；专属连接仍被任务引用时不能修改产品类型
v1.11.0 安装包	安装包约 82 MB，与 v1.10.0 基本持平：新增 MongoDB 原生驱动约 2.2 MB，未引入其他运行时依赖
v1.12.0 结构	不新增元数据表，无数据库结构变更；替换 `jync.jar` 重启即可，容器部署把镜像 tag 更新为 `jync:1.12.0`
v1.12.0 默认行为	内容级校验在增量成功后自动运行，默认 15 分钟间隔、每表 50 行、单表 200 万行上限、每项目每周期 1 张表；可用 `sync.content-audit-*` 调整，设间隔为 `0` 关闭。同步前检查与诊断包只在用户点击时执行，没有调度器
v1.12.0 诊断包	设置页新增管理员可见的「诊断包」卡片，导出动作写入 `DIAGNOSTICS_EXPORT` 审计；包内不含密码、密文、连接串凭据、游标值与 `data/`，但可能含业务信息，按敏感文件处理
v1.12.0 方言	MariaDB 改用正数流式读取并默认 `useAffectedRows=true`；DB2 目标 MERGE 参数统一加 CAST（修复 SQL0418）；视图 / 存储过程转换新增多项改写，无法保证等价的写法改为明确阻断。已有连接与任务无需改动
v1.12.0 安装包	安装包约 82 MB，与 v1.11.0 基本持平：预检、校验与诊断包均为应用内代码，ClickHouse JDBC 仅用于测试范围，不进发行包

CHAPTER 12

安全加固

Jync 保存数据库连接口令、能看到业务数据、还能建表与写入。它的安全边界不只是控制台登录，还包括传输、密钥、数据库权限与出网行为。

12.1 账号与口令

- 首次登录立即修改 admin 与 view 的默认密码 `123456`。
- view 账号只用于查看；不要让值班人员共用 admin。
- 控制台账号与数据库账号分离，避免同一口令跨系统复用。
- 人员离场时及时停用账号，并检查审计日志中的近期操作。

12.2 传输与 Cookie

项	建议	说明
HTTPS	必须	口令与业务元数据都经控制台传输
Secure Cookie	开启	`JYN_COOKIE_SECURE=true`
HttpOnly	默认开启	脚本无法读取会话 Cookie
SameSite	默认 Lax	阻止跨站 POST，保留正常导航；iframe 跨站嵌入时设为 None，并同时开启 Secure
代理头	正确传递	`X-Forwarded-Proto` 影响重定向与 Cookie 判断

12.3 密钥管理

- 新部署留空 `crypto-password` 与 `crypto-salt`，让系统自动生成 `jync.key`。
- 密钥文件权限设为 600，属主为运行账号，不进入代码仓库与普通备份盘。
- 多实例部署显式设置 `SYNC_CRYPTO_*`，各节点一致；使用 Secret 管理，不写明文。
- 密钥轮换需要重新录入所有连接口令，属于高风险操作，应制定专门流程。

12.4 数据库最小权限

角色	最小权限	不要授予
源库只读	SELECT、读取元数据	INSERT / UPDATE / DELETE、DDL

角色	最小权限	不要授予
目标库写	建表、索引、DML	DROP DATABASE、管理账号权限
MySQL CDC	REPLICATION SLAVE / CLIENT	SUPER、全局写权限
Oracle 挖掘	SELECT ANY TRANSACTION、LOGMNR	SYSDBA、DBA
PostgreSQL	REPLICATION、SELECT	SUPERUSER
SQL Server	cdc_admin / cdc_reader	sysadmin

按同步任务划分账号

不同同步任务访问不同库时，为每个库单独建账号，并在 Jync 中按同步任务绑定。一个同步任务出问题不会波及其他库。

12.5 网络隔离与数据出网

Jync 默认只在三种情况下访问外网：版本检查、AI 起草与 Webhook 告警。版本检查访问 GitHub / Gitee，AI 起草访问你配置的模型端点，告警只在管理员填了 Webhook 地址后才会外呼。三者都可以关闭。

开关	关闭后	适用
`app.update-check.enabled=false`	不再访问代码托管平台，升级入口隐藏	纯内网、等保环境
`sync.ai.enabled=false`	AI 菜单隐藏，请求在发出前拦截	不允许源码出网的场景
不配置 AI 供应商	功能保留但不产生任何外部请求	暂时不用 AI，但保留入口
不配置 Webhook 地址，或关闭告警开关	不评估规则、不发送任何告警请求	默认不出网的内网环境

Webhook 是服务端主动外呼

告警地址由管理员填写，Jync 会从服务器直接 POST JSON 到该地址，属于 SSRF 面。只填可信内网接口或官方机器人地址，不要填公网或不可信地址；地址加密保存，页面只回显尾号。不需要告警时保持开关关闭。

AI 会把过程源码发到端点

启用 AI 后，转换审阅页的「用 AI 起草」会把存储过程源码发送给配置的模型端点。内网或合规部署应使用本地推理端点，或保持 AI 关闭。

12.6 外部认证与单点登录

外部认证把控制台登录交给企业已有身份体系，配置入口在「配置管理 → 外部认证」。三种方式同一时刻只启用一种，安全边界各不相同：

方式	信任来源	加固要点
账号密码（OA 接口）	企业 OA 登录接口的校验结果	只填可信内网地址；只接受 http(s)；明文 HTTP 必须显式开启；不跟随重定向；响应体上限 64 KB
网关头	直接 TCP 对端位于回环或可信代理列表内的请求头	Jync 必须部署在网关之后；配置 `app.security.trusted-proxies`；网关剥离客户端伪造的身份头；可选共享密钥头
票据校验	票据校验接口的校验结果	地址只填可信内网；票据一次性、不写审计、不在页面回显；落地页一律 302

- **外部用户不是本地账号**不写入 `app_user`，没有本地密码，服务端拒绝 `/account/password`；账号生命周期由门户管理。
- **管理员来源**用户名清单、认证响应中的管理员标识字段，或网关头模式下的管理员标识头；其余外部用户为只读角色。
- **共享密钥**网关头模式的共享密钥加密存储、常量时间比较；密钥无法解密时一律拒绝请求。
- **失败锁定**本地账号与 OA 共用「用户名 + 客户端 IP」计数，5 次失败锁 15 分钟；票据入口按 IP 独立限流，20 次失败暂停 15 分钟。
- **不记录敏感内容**失败审计只记录用户名、来源与错误类别；密码、票据与外部响应正文都不写日志与审计。
- **关闭本地登录前先验证外部认证**关闭后 `POST /login` 与 `/login/local` 都不再放行；应急恢复用 `JYN_RESET_LOGIN=true` 启动一次。

> **外部认证属于 SSRF 面**
>
> OA 登录地址与票据校验地址由管理员填写，服务器会主动外呼。只允许可信内网地址，不要填公网地址或其他可被诱导访问的内部服务；修改后先用页面上的测试按钮验证身份与来源，再启用。

CHAPTER 13

故障排查

排查从「现在哪一层不工作」开始：进程、元数据、网络、权限、数据、性能。下面的表按层组织，每行给出可执行的检查动作。先取证，再修复。

13.1 启动与元数据

现象	原因	处理
启动即退出，提示 JDK 版本	使用了 JDK 8 或 11	安装 JDK 17，确认 `java -version`
H2 文件锁冲突	同一数据目录被两个进程/容器占用	确保单实例，或多实例改用外部 MySQL
登录页显示 `??key_zh_CN??`	用 `spring.config.location` 替换了内置配置	改用 `additional-location`
全新部署不建表	覆盖配置丢失了 `ddl-auto`	恢复内置配置，确认使用叠加方式
启动很慢后超时	元数据库较大，首启建表或迁移耗时	调大 startupProbe 的 failureThreshold
磁盘满导致启动失败	日志或元数据增长	清理旧日志与备份，确认 500 MB 日志上限生效

13.2 连接与权限

现象	原因	处理
连接测试超时	网络不通、防火墙、端口错误	在服务器上用客户端工具直连验证
SQL Server 命名实例超时（约 15 秒）	URL 固定拼了 `:1433`，命名实例的端口由 SQL Server Browser 动态分配	端口留空，主机写 `主机名\实例名` 或附加参数填 `instanceName`；1.7.7 起空端口不拼 `:port`
认证失败	口令错误、账号被锁、认证插件不兼容	核对账号；MySQL 8 注意 `caching_sha2_password`
库不存在	URL 中的库名拼写错误	Oracle 用 Service Name，不是 SID
SSL 相关错误	驱动默认要求 TLS，源库未启用	在附加参数中显式关闭或配置信任
容器内全部超时	`localhost` 指向容器	改用宿主机 IP 或 `host.docker.internal`
时间差 8 小时	容器缺时区	设置 `TZ=Asia/Shanghai`，并检查数据库时区

13.3 结构与全量

现象	原因	处理
建表失败，类型不兼容	源类型在目标库无对应	在类型映射中调整，或改用通用类型
视图/过程创建失败	引用了未同步的对象或方言不支持	先补齐依赖，再进入转换审阅手工覆盖
全量中途停止	连接中断、锁丢失、手动停止	检查错误类型；续跑从检查点继续，不需要从头
全量很慢	并行未生效、索引维护、源库读取慢	看吞吐日志的读写占比，按第 5 章调优
全量后行数不一致	源表在加载期间持续变化，或部分分块失败	看行数审计；必要时重建基线
检查点被丢弃、从头加载	记录下来的区间布局不完整、前后矛盾或来自更早版本	属于保护行为，避免跳段丢行；正常情况下源表增长、改并行度都会按原布局续跑。日志出现 `Resuming the ... split` 表示已续跑
千万级大表首次全量反复失败、每次重来	v1.11.0 及更早版本在源表持续增长时无法对上上次的切分	升级到 v1.12.0；新版本按记录的区间续跑，新增行只在尾部追加。升级前不要反复手动重跑同一张大表
MariaDB 报 `invalid fetch size`	驱动不接受负的 fetch size（流式标记）	v1.12.0 改用正数流式读取；旧版本升级或把该连接的 fetch 调成正数
MariaDB 空转 upsert 被记成更新、行数审计偏差	驱动默认按 matched 而非 changed 返回受影响行数	v1.12.0 预设 URL 自动带 `useAffectedRows=true`；旧连接可在附加参数中手工补上
DB2 目标报 SQL0418 / SQLSTATE=42610	MERGE 的无类型参数标记在三处解析出不同类型	v1.12.0 按目标列统一加 CAST；全量、增量与 CDC 均生效
源库删了表，目标表还在	源库账号看不见整个 schema，权限收回与删表无法区分	属于保护行为；以表属主或库级 `SELECT` 同步后删表才会联动。日志关键字 `cannot see every object in the schema`

13.4 增量与 CDC

现象	原因	处理
轮询看不到更新	游标列未更新或策略选错	确认 update_time 由业务维护，或改用 ROW_SCN / CDC
「下次自动运行」时间停在过去、数据不再增长	上一轮没有返回，Quartz 已跳过它之后的所有触发（疑似卡死）	详情页会显示「上一轮仍在执行 / 疑似卡死」；等待不会恢复，重启本实例。强制解锁只清锁记录，不结束线程

现象	原因	处理
轮询看不到删除	轮询机制不感知 DELETE	有主键的小表开「同步行删除」并用 FULL_COMPARE，大表切 CDC
CDC 启动失败	日志未开启、权限不足、版本不符	按 6.2 逐项核对
位点不推进	日志读取被阻塞、租约丢失、无新变更	先制造一条变更验证，再查租约与错误日志
落后秒数上涨	目标写入跟不上	降低增量批次、错峰、优化目标索引
跳过计数增长	目标表结构或主键冲突	按日志修表，再重建基线
多节点同时消费	租约失效或时钟偏差	校时；确认元数据库连通；必要时停止后重启同步任务
源端报同一 server_id 冲突	同一同步任务被两个实例或两份元数据库同时消费	只保留一个实例消费；被踢下线后按 6.7 重建基线恢复

13.5 性能与资源

现象	原因	处理
全量吞吐远低于基准	并行度为 1、批次过小、目标索引多	确认表有主键、连接池满足并行度要求，按第 5 章调参
数据库连接被打满	并行度或连接池配置过高	按 `pool - 1` 限制并行，降低 pool
堆内存持续增长	大表全量、fetch 过大、并发同步任务过多	降低 fetch 与并行度，错开大同步任务
查询长时间不返回	锁等待或网络黑洞	查询超时默认 300 秒；用 jstack 看线程栈
GC 频繁	堆过小或批次过大	增大堆，或降低全量批次与并行度

```
# 线程与进程取证
jcmd <pid> Thread.print > /tmp/jync-threads.txt
jcmd <pid> GC.heap_info
ss -tnp | grep 8080
df -h /opt/jync
```

13.6 页面、升级与升级后异常

现象	原因	处理
登录后立即掉线	Secure Cookie 配了但走 HTTP，或代理未传协议头	统一 HTTPS，传递 `X-Forwarded-Proto`
iframe 嵌入时登录态丢失	跨站 iframe 下 SameSite=Lax 不发送会话 Cookie	设 `JYN_COOKIE_SAME_SITE=None` 并同时开启 `JYN_COOKIE_SECURE=true`，且必须走 HTTPS
登录页只有门户入口	已启用外部认证并关闭了本地登录	从门户进入；紧急时以 `JYN_RESET_LOGIN=true` 启动一次恢复本地登录，修好配置后去掉
网关头登录后仍是未登录	直接对端不在可信代理列表内，或网关没有注入身份头	核对 `app.security.trusted-proxies` 与网关配置；确认 Jync 没有直接暴露给客户端
OA / 票据登录一直失败	接口地址、字段映射或成功标识不匹配；或已触发失败锁定	用页面测试按钮看实际响应；核对字段名与响应路径；等待 15 分钟或从审计确认锁定来源
系统日志页提示文件不可用	`logging.file.name` 指向白名单外，或文件权限不足	恢复默认 `./logs/jync.log`，确认运行账号可读日志目录
页面样式错乱	浏览器缓存了旧静态资源	强制刷新；静态资源默认每次条件请求
一键升级下载失败	外网不可达，或平台的匿名 API 限流（HTTP 403 / 429）	限流时切换下载源或稍后重试；外网不可达改用离线升级，纯内网关闭更新检查
升级后连接口令失效	误改了加密密钥	恢复原 `crypto-*` 配置与密钥文件
升级后启动失败	元数据结构与旧程序不兼容	回滚到旧 JAR，或恢复升级前 data 备份
升级后 CPU 升高	新版本首次重建缓存或索引	观察 10 分钟；持续升高时查线程栈

CHAPTER 14

附录

14.1 参数速查

配置键	默认	一句话说明
`server.port`	8080	Web 端口
`spring.datasource.url`	H2 文件	元数据库地址
`spring.datasource.hikari.maximum-pool-size`	10	元数据连接池
`sync.poll-interval`	2000	轮询间隔（毫秒）
`sync.max-retries`	3	同步单元失败重试次数
`sync.batch-size`	500	增量与 CDC 批次
`sync.fetch-size`	1000	增量读取批量
`sync.full-load-batch-size`	2000	全量写入批次（1–20000）
`sync.full-load-fetch-size`	2000	全量读取批量（1–20000）
`sync.full-load-parallelism`	2	全量并行度（1–8，受连接池约束）
`sync.chunk-size`	100000	分块行数（小于等于 0 关闭分块）
`sync.max-pool-size`	5	每库连接池上限
`sync.pool-wait-ms`	10000	池满排队退让的等待上限（0 = 立即退让）
`sync.oracle-initial-load-insert-first`	true	空表 INSERT 优先
`sync.defer-secondary-indexes`	true	二级索引延后
`sync.safety-lag-ms`	1000	时间戳高水位回退
`sync.row-count-audit-interval-ms`	60000	行数审计周期
`sync.row-count-repair-max-rows`	1000	审计缺口的定点补行上限（0 关闭）
`sync.full-compare-max-rows`	20000	自动选用全表比对的行数上限（手动选择不受限）
`sync.delete-compare-max-rows`	0	删除比对分块的行数上限（0 关闭）
`sync.lock-ttl-ms`	300000	同步任务锁有效期
`sync.stuck-threshold-ms`	1800000	无进展判定卡死的阈值（0 关闭）
`sync.reconcile-interval-ms`	30000	多节点对账周期
`sync.query-timeout-seconds`	300	数据查询超时

配置键	默认	一句话说明
`sync.metadata-query-timeout-seconds`	60	元数据查询超时
`sync.shutdown-wait-ms`	20000	停机等待在途批次
`sync.cdc.reconnect-backoff-ms`	1000	CDC 重连初始退避
`sync.cdc.reconnect-backoff-max-ms`	60000	CDC 退避上限
`sync.cdc.server-id-offset`	612000	MySQL 假 replica 基数
`sync.cdc.lock-ttl-ms`	30000	CDC 租约时长
`sync.cdc.supervise-interval-ms`	10000	租约巡检周期
`sync.cdc.connect-timeout-ms`	10000	binlog 握手超时
`sync.cdc.change-window-ms`	10000	变更日志聚合窗口
`sync.cdc.logminer.batch-size`	10000	LogMiner 单批行数
`sync.cdc.logminer.idle-poll-ms`	1000	LogMiner 空闲休眠
`sync.cdc.pg.idle-poll-ms`	500	PG 空闲休眠
`sync.cdc.mssql.idle-poll-ms`	1000	SQL Server 空闲休眠
`sync.cdc.pg.wal-retention-warn-bytes`	4 GiB	PG 复制槽 WAL 滞留告警阈值（0 关闭）
`sync.ai.enabled`	true	AI 功能总开关
`app.update-check.enabled`	true	版本检查开关
`app.update-check.timeout-ms`	5000	单平台检查超时
`app.update-check.match-artifact`	true	同版本重发时比对安装包；关闭后只比较版本号
`app.security.trusted-proxies`	空（仅回环）	网关头模式的可信代理列表
`app.login.reset-on-startup`	false	应急恢复开关，启用一次即恢复本地登录
`server.servlet.session.cookie.same-site`	lax	会话 Cookie SameSite 策略
`server.servlet.session.cookie.secure`	false	HTTPS 部署设为 true
`metrics.enabled`	true	性能采样总开关
`metrics.retention-days`	7	指标样本保留天数
`metrics.task-interval-ms`	60000	同步任务采样间隔
`metrics.db-interval-ms`	300000	数据库采样间隔
`metrics.capacity-interval-ms`	3600000	容量采样间隔
`metrics.prometheus-token`	空（端点 404）	Prometheus Bearer token

配置键	默认	一句话说明
`logging.file.name`	./logs/jync.log	日志路径
`logging.logback.rollingpolicy.max-file-size`	10MB	单文件上限
`logging.logback.rollingpolicy.max-history`	14	保留天数
`logging.logback.rollingpolicy.total-size-cap`	500MB	日志总量上限

14.2 环境变量速查

变量	对应配置	典型值
`TZ`	容器时区	Asia/Shanghai
`JAVA_OPTS`	JVM 参数	-XX:MaxRAMPercentage=75
`SERVER_PORT`	server.port	8080
`JYN_COOKIE_SECURE`	会话 Cookie Secure	true
`JYN_COOKIE_SAME_SITE`	会话 Cookie SameSite	lax（iframe 跨站用 None，需同时 Secure）
`JYN_TRUSTED_PROXIES`	app.security.trusted-proxies	10.0.0.10,10.0.0.11
`JYN_RESET_LOGIN`	app.login.reset-on-startup	true（仅在应急恢复时临时使用一次）
`SPRING_DATASOURCE_URL`	元数据库地址	jdbc:mysql://.../jync
`SPRING_DATASOURCE_USERNAME`	元数据库账号	jync
`SPRING_DATASOURCE_PASSWORD`	元数据库口令	来自 Secret
`SYNC_CRYPTO_PASSWORD`	sync.crypto-password	多实例必填
`SYNC_CRYPTO_SALT`	sync.crypto-salt	多实例必填
`SYNC_POLL_INTERVAL`	sync.poll-interval	2000
`SYNC_AI_ENABLED`	sync.ai.enabled	false
`APP_UPDATE_CHECK_ENABLED`	app.update-check.enabled	false
`APP_UPDATE_CHECK_MATCH_ARTIFACT`	app.update-check.match-artifact	true
`JYN_METRICS_TOKEN`	metrics.prometheus-token	留空关闭 Prometheus 端点；启用时用足够长的随机串

14.3 运维检查清单

每日

- ✓ 看板无红色异常，同步任务运行状态符合预期。
- ✓ CDC 同步任务落后秒数正常，位点持续推进。
- ✓ 日志无持续 ERROR，行数审计无新差异。
- ✓ 数据库与同步任务性能页的延迟曲线没有持续上涨；启用 Prometheus 时抓取状态为 UP。
- ✓ 磁盘与数据库连接数在阈值内。

每周

- ✓ 抽查一个同步任务的目标表行数与关键字段。
- ✓ 检查备份任务执行结果，确认备份文件可读。
- ✓ 检查源库日志保留时间是否覆盖最长的计划停机窗口。
- ✓ 复核新增同步任务的游标策略与全量参数。

每月

- ✓ 做一次恢复演练或备份校验。
- ✓ 检查账号与权限，清理离场人员账号。
- ✓ 评估全量吞吐趋势，必要时调整并行度与批次。
- ✓ 检查版本更新，评估升级窗口。

14.4 术语表与支持

术语	含义
全量	把源表当前全部数据写入目标表，是增量的起点
增量	按游标或日志捕获变化并写入目标库
基线	CDC 开始回放前的一次全量与位点记录
位点	CDC 已应用到的日志位置（SCN / LSN / binlog 偏移）
游标	轮询模式记录的业务进度（时间、SCN 或主键）
检查点	全量分块完成后记录的进度，用于中断续跑
租约	CDC 消费线程的短锁，保证同一同步任务只有一个节点消费

术语	含义
安全滞后	时间戳游标回退量，用于覆盖乱序提交
幂等	重复执行不会产生重复数据；upsert 是典型实现

渠道	地址	说明
官网	jync.qqmu.com	产品介绍、文档与下载
GitHub	github.com/vfaner/jync	源码、Issue、Release
Gitee	gitee.com/super_rgh/jync	国内镜像与 Release
邮箱	vfaner@88.com	商务与合作
QQ 群	426669837	沐沐开源工坊

配套文档

日常操作步骤见《Jync 用户操作手册》；产品能力与选型见《Jync 产品介绍》。三本手册均随版本发布，版本号与发行标签一致。

Jync
捷同 · 异构数据库同步

让数据在异构库之间稳定地流动

Jync 捷同 · 异构数据库同步平台

部署与运维手册 · v1.12.0

jync.qqmu.com · v1.12.0 · 2026 年 11 月

技术交流 QQ 群 426669837 · 邮箱 vfaner@88.com

© Jync 捷同 · 沐沐开源工坊