

Cargue sus datos en Adabas sin escribir a mano.

nsadaloader carga registros desde JSON, CSV, TSV o XML directamente en sus ficheros Adabas: STORE, UPDATE y DELETE, con campos MU, grupos PE y control por lotes.

VER FUNCIONALIDADES →

STORE · UPDATE · DELETE · JSON · CSV · XML · Lotes configurables



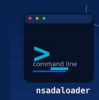
JSON · CSV · XML

autodetección por extensión



Fichero Adabas

STORE, UPDATE o DELETE



Cargar datos a mano no escala

Cada fichero nuevo que hay que cargar significa el mismo trabajo: un programa, una transacción, una lista de errores y dos días de pruebas.

1

Carga caso a caso

Un desarrollo por cada fichero, cada formato y cada casuística, sin reutilizar nada.

2

Errores que sólo aparecen en producción

Campos truncados, MU que no encajan, transacciones que se quedan a medias.

3

Sin trazabilidad

Un fichero cargado a medias y ni un log que explique qué pasó.

nsadaloader convierte la carga en un proceso repetible, con simulación previa y traza de cada registro.

QUÉ ES

El cargador masivo Adabas, de Nacarsoft

nsadaloader lee un fichero de entrada, lo valida contra el DDM y escribe los registros en Adabas dentro de transacciones por lotes, con la traza de cada operación.

Maneja STORE, UPDATE y DELETE, campos MU, grupos PE y nombres DDM lógicos. Con -dry-run ve exactamente lo que va a hacer antes de tocar el núcleo.

4

formatos de entrada

3

operaciones

1

traza completa



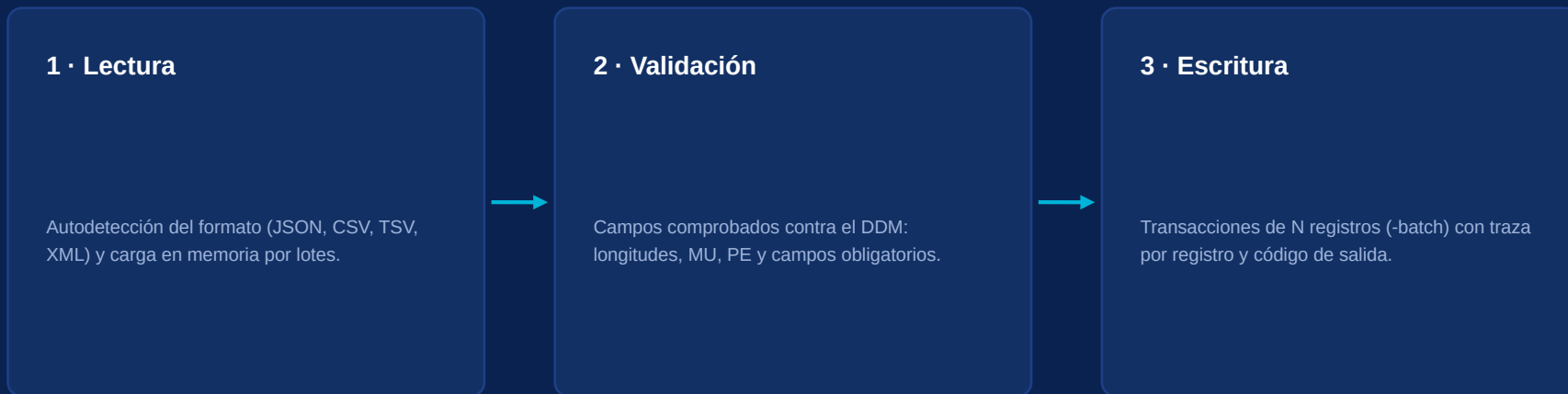
nsadaloader

Fichero → Adabas, con control

- JSON, CSV, TSV y XML con autodetección
- STORE, UPDATE y DELETE en lote
- Campos MU y grupos PE nativos
- -dry-run: simula antes de escribir

Del fichero de entrada al núcleo, con control

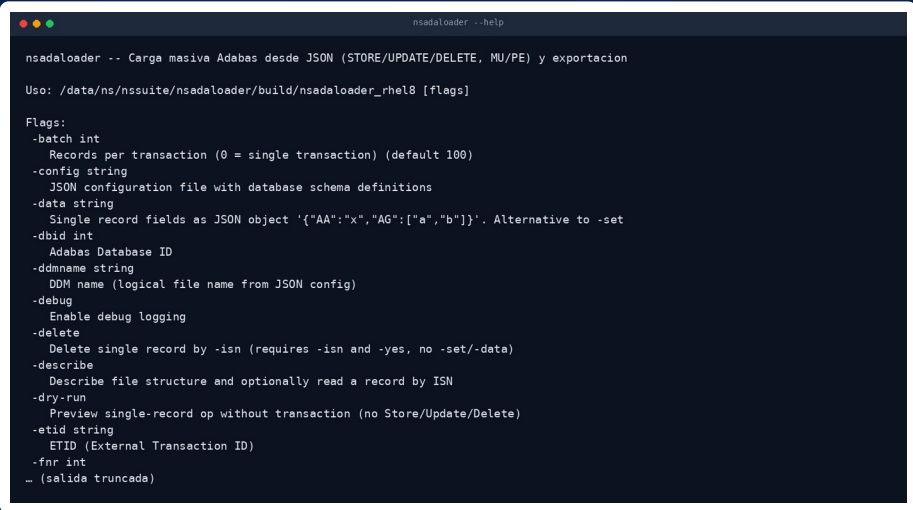
Validación previa, transacción por lote y traza: nada se escribe a ciegas.



Con -dry-run no se abre ninguna transacción: ve el plan antes de ejecutarlo.

Un cargador, todas las entradas

El mismo binario para JSON, CSV, TSV y XML, con autodetección por extensión y validación contra el DDM.



```
nsadaloader --help

nsadaloader -- Carga masiva Adabas desde JSON (STORE/UPDATE/DELETE, MU/PE) y exportacion

Uso: /data/ns/nssuite/nsadaloader/build/nsadaloader_rhel8 [flags]

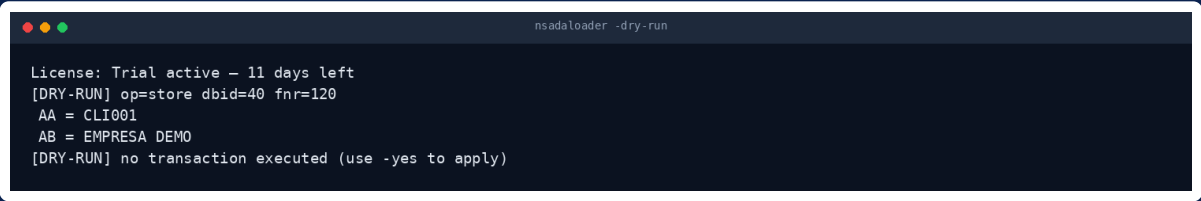
Flags:
  -batch int
    Records per transaction (0 = single transaction) (default 100)
  -config string
    JSON configuration file with database schema definitions
  -data string
    Single record fields as JSON object '{"AA":"x","AG":["a","b"]}'. Alternative to -set
  -dbid int
    Adabas Database ID
  -ddmname string
    DDM name (logical file name from JSON config)
  -debug
    Enable debug logging
  -delete
    Delete single record by -isn (requires -isn and -yes, no -set/-data)
  -describe
    Describe file structure and optionally read a record by ISN
  -dry-run
    Preview single-record op without transaction (no Store/Update/Delete)
  -etid string
    ETID (External Transaction ID)
  -fnr int
    ... (salida truncada)
```

Salida real de nsadaloader --help

- Sin desarrollo a medida
- Validación contra el DDM
- Transacciones por lote

Primero se simula, luego se carga

-dry-run muestra la operación, los valores y el fichero afectado sin abrir ninguna transacción contra el núcleo.

A terminal window titled 'nsadaloader -dry-run' with a dark background and light text. The output shows a license status, a dry-run operation for storing data, and the specific values for the operation.

```
nsadaloader -dry-run

License: Trial active - 11 days left
[DRY-RUN] op=store dbid=40 fnr=120
AA = CLI001
AB = EMPRESA DEMO
[DRY-RUN] no transaction executed (use -yes to apply)
```

Salida real de nsadaloader con -dry-run

- Plan antes de ejecutar
- Sin transacciones
- Traza por registro

Lo que hace, punto por punto

Pensado para cargas migraciones y pruebas, con el mismo control que en producción.



Cuatro formatos de entrada

JSON, CSV, TSV y XML con autodetección por extensión, sin configuración.



Campos MU y grupos PE

Soporte nativo de elementos repetidos y de grupos periódicos, sin trucos.



Transacciones por lotes

-batch controla el tamaño de cada transacción (100 registros por defecto).



Plantillas con -describe

Describe la estructura del fichero y lee un registro por ISN, en sólo lectura.



STORE, UPDATE y DELETE

Inserción, actualización y borrado por ISN en la misma carga.



Simulación (-dry-run)

Veja la operación y sus valores sin ejecutar ninguna transacción.



Nombres DDM lógicos

Con la configuración de nsddm, los campos se nombran como en el DDM.



Exportación a JSON

-output vuelca el fichero a JSON: también sirve para extraer y para probar.

Los parámetros que se usan todos los días

Combinables entre sí; el resto de banderas están en --help.

-db id <id>

Base de datos Adabas de destino. Obligatorio.

-fnr <n> | -ddm name

Fichero por número o por nombre lógico DDM (requiere -config).

-input <fichero>

Fichero de entrada: JSON, CSV, TSV o XML.

-set | -data

Un solo registro como pares o como objeto JSON.

-dry-run

Simula la operación sin abrir transacción.

-batch <n>

Registros por transacción (0 = una sola transacción).

-describe

Describe la estructura del fichero y lee un registro por ISN.

-output <fichero>

Exporta el fichero a JSON en lugar de escribir en Adabas.

Dónde se usa en el día a día

Los mismos casos que resolvemos con clientes en banca, sanidad y sector público.

01

Migración de un fichero

Carga inicial de un fichero Adabas desde el export de otro sistema.

02

Corrección de datos

UPDATE masivo de campos concretos sobre miles de registros, con dry-run antes.

03

Pruebas de entorno

Carga un subconjunto de datos en el entorno de pruebas y lo borra después.

04

Carga de referencias

Alta de catálogos y tablas de referencia con validación contra el DDM.

Instalar y cargar en cinco minutos

Requisitos

- Linux x86-64 (RHEL 8 o RHEL 9)
- Cliente nativo Adabas (acj) en el PATH
- Acceso de escritura a la base de datos
- Licencia del producto o modo trial de 30 días
- DDM en JSON (nsddm) para nombres lógicos

Instalación y primer uso

```
# Instalar la suite
./install.sh --prefix /opt/ns

# Simular una carga
nsadaloader -dbid 40 -input clientes.json \
  -fmr120 -dry-run

# Cargar de verdad (por lotes)
nsadaloader -dbid 40 -input clientes.json \
  -fmr120 -batch 500 -yes
```

La licencia se busca en \$NS_LICENSE_FILE, junto a los datos del producto o en /opt/ns/licenses/. Sin licencia: trial de 30 días.

Lo que siempre preguntan

¿Escribe directamente en el núcleo?

Sí, en transacciones por lotes, y con -dry-run puede ver beforehand exactamente lo que va a escribir.

¿Qué pasa si un registro falla?

Queda registrado en la traza con el ISN y el motivo, y el proceso devuelve un código de salida distinto de cero.

¿Admite campos MU y PE?

Sí, de forma nativa: elementos repetidos y grupos periódicos con la misma semántica del DDM.

¿Necesita el DDM?

Para trabajar por número de fichero, no. Con el JSON de nsddm puede usar nombres de campo lógicos y validar la carga.

SIGUIENTE PASO

Cargue sus datos sin una sola línea de código propia.

Pídanos una demo con su DDM y su fichero de entrada: le enseñamos la simulación y la carga real sobre su base.

PEDIR UNA DEMO →

nacarsoft.com · soporte@nacarsoft.com · +34 910 038 502



NacarSoft Soluciones Informáticas S.L. · nacarsoft.com · nsadaloader

NSADALoader en un vistazo

- Entrada JSON, CSV, TSV o XML
- STORE, UPDATE y DELETE en lote
- Campos MU y grupos PE nativos
- -dry-run antes de tocar el núcleo