

# JCTNT - Jflex Cheap Table Name Translator

Applicazione web per la traduzione bidirezionale dei nomi di tabelle e campi JAS/JFlex tra nomenclatura fisica (database) e logica (applicativa).

## PARTE 1: Architettura e Installazione

### 1.1 Stack Tecnologico

| Componente     | Tecnologia                           |
|----------------|--------------------------------------|
| Backend        | Flask 3.0 (Python)                   |
| Database       | Oracle (oracledb driver, thick mode) |
| Frontend       | HTML5 + CSS3 + JavaScript vanilla    |
| Storage locale | File JSON                            |

### 1.2 Struttura Cartelle

|                           |                                   |
|---------------------------|-----------------------------------|
| JCTNT/                    |                                   |
| — app.py                  | # Backend Flask + API REST        |
| — templates/              |                                   |
| — index.html              | # Markup HTML (SPA)               |
| — static/                 |                                   |
| — css/                    |                                   |
| — style.css               | # Stili (~350 righe)              |
| — js/                     |                                   |
| — app.js                  | # Logica applicativa (~250 righe) |
| — Data/                   |                                   |
| — connection_data.json    | # Ultima connessione (auto)       |
| — connection_history.json | # Storico connessioni (auto)      |
| — search_history.json     | # Storico ricerche (auto)         |
| — requirements.txt        | # Dipendenze Python               |

### 1.3 Endpoint API

| Metodo | Endpoint | Descrizione      |
|--------|----------|------------------|
| GET    | /        | Serve index.html |

| Metodo | Endpoint                | Descrizione                        |
|--------|-------------------------|------------------------------------|
| GET    | /api/connection-data    | Ultima connessione salvata         |
| GET    | /api/connection-history | Lista connessioni salvate          |
| GET    | /api/search-history     | Lista ricerche salvate             |
| POST   | /api/add-search-history | Aggiunge ricerca alla history      |
| POST   | /api/connect            | Connette al DB e carica dizionario |

### 1.4 Query di Caricamento

Dizionario campi (~37.000 record):

```

sql

SELECT
  TAB.TABLEDBNAME  AS TABELLA_FISICA,
  FIE.DBFIELDNAME  AS CAMPO_FISICO,
  TAB.TABLENAME    AS TABELLA_LOGICA,
  FIE.TABLEFIELDNAME AS CAMPO_LOGICO,
  FIE.TYPE         AS TIPO,
  FIE.WIDTH        AS AMPIEZZA,
  FIE.DECIMALS     AS DECIMALI
FROM FW_TABLES TAB
JOIN FW_TABLE_FIELDS FIE ON (FIE.TABLENAME = TAB.TABLENAME)

```

Indici e colonne:

```

sql

SELECT table_name, index_name, uniqueness
FROM user_indexes

SELECT table_name, index_name, column_name, column_position
FROM user_ind_columns
ORDER BY index_name, column_position

```

### 1.5 Requisiti

- Python 3.10+
- Oracle Instant Client (nel PATH di sistema)
- Browser moderno (Chrome, Firefox, Edge)

## 1.6 Installazione

### 1. Crea la struttura

```
bash

mkdir JCTNT && cd JCTNT
mkdir templates
mkdir -p static/css static/js
mkdir Data
```

### 2. Copia i file

| File       | Destinazione      |
|------------|-------------------|
| app.py     | JCTNT/            |
| index.html | JCTNT/templates/  |
| style.css  | JCTNT/static/css/ |
| app.js     | JCTNT/static/js/  |

### 3. Crea requirements.txt

```
Flask==3.0.0
oracledb==2.0.0
```

### 4. Installa dipendenze

```
bash

pip install -r requirements.txt
```

## 1.7 Avvio

```
bash

cd JCTNT
python app.py
```

Output atteso:

\* Running on http://127.0.0.1:5000

\* Debug mode: on

Apri il browser su: <http://localhost:5000>

---

## PARTE 2: Guida all'Utilizzo

### 2.1 Flusso di Lavoro

CONNESSIONE → RICERCA → RISULTATI

### 2.2 Step 1: Connessione

All'apertura dell'applicazione ti trovi nella schermata di connessione.

**Se hai già usato l'app:**

- Seleziona una connessione dal dropdown "Connessioni salvate"
- I campi si compilano automaticamente

**Se è la prima volta:**

- Compila manualmente: Host, Porta, SID, Username, Password

Clicca **Connetti**. Se la connessione va a buon fine, il box diventa verde e compare il pulsante **Avanti**.

### 2.3 Step 2: Ricerca

Nella schermata di ricerca hai due campi:

**Nome Tabella** (obbligatorio)

- Puoi inserire il nome fisico (es. `MD_ARTI`) oppure il nome logico (es. `Articoli`)
- La ricerca non distingue maiuscole/minuscole

**Campi da cercare** (opzionale)

- Lascia vuoto per vedere tutti i campi della tabella
- Oppure inserisci i nomi dei campi che ti interessano
- Puoi separarli con virgola, spazio o andando a capo

Clicca **Traduci** per vedere i risultati.

## 2.4 Step 3: Risultati

I risultati sono divisi in due tab:

### Tab "Tabella"

- Mostra l'elenco dei campi con: nome fisico, nome logico, tipo, ampiezza, decimali
- I campi non trovati appaiono in rosso con "NOT FOUND"

### Tab "Indici"

- Mostra gli indici della tabella
- Clicca sul triangolino a sinistra per espandere e vedere le colonne dell'indice

## 2.5 Copiare i Valori

### Copia singola cella:

- Clicca su qualsiasi cella della tabella
- Appare il feedback "Copiato!" e il valore è nella clipboard

### Copia tutto:

- Usa il pulsante "Copia tutto" in alto a destra
- Nel tab Tabella: copia in formato TSV (incollabile in Excel)
- Nel tab Indici: copia in formato leggibile con tutte le colonne

## 2.6 Navigazione

- Le card "Ricerca" e "Risultati" sono espandibili/collassabili cliccando sull'intestazione
- Per fare una nuova ricerca sulla stessa connessione: espandi la card Ricerca
- Per cambiare database: clicca **Nuova Connessione**

## 2.7 Gestione Automatica della History

L'applicazione salva automaticamente:

- **Connessioni:** ogni connessione riuscita viene memorizzata per uso futuro
- **Ricerche:** ogni tabella cercata appare nel dropdown "Ricerche recenti"

I file di history si trovano nella cartella Data/ e vengono gestiti automaticamente.

---

