

V O L V O

## COMMON POLSKA

14-17 June 2026, Lyon

Art of sharing

# The Next Level of Integration: Building Hybrid AI-Powered Workflows with n8n and IBM i

MARCIN OGOŃ





MARCIN OGOŃ



Doświadczenie



**Owner**

Alpinsoft - Samozatrudnienie  
 maj 2015 – obecnie - 9 lat 11 mies.  
 Poland

Software Development / IT Consulting

Projects for automotive:

- Service contracts
- Excel files importing and parsing tool
- IBMi REST integration with JWT
- Angular Front-End application
- Azure and Openshift
- SQLRPGLE, Java, Apache Camel, Spring Boot, JavaScript, TypeScript, Angular

Projects for banking:

- Pricing and billing application
- Card services
- IBMi administration tools



**Contractor**

Fiserv  
 cze 2015 – maj 2021 - 6 lat  
 Warsaw, Masovian District, Poland

ICBS 24/7 - online transaction processing  
 JSON extract for debt collection...



**ING Bank Śląski**

7 lat 2 mies.  
 Katowice, Silesian District, Poland

● **IT Development Specialist**

maj 2012 – lut 2015 - 2 lata 10 mies.

ICBS (International Comprehensive Banking System) modifications - RPG and CL.  
 Customer Screening - Java, VBS...

● **IT Developer**

maj 2008 – maj 2012 - 4 lata 1 mies.

ICBS (International Comprehensive Banking System) modifications - RPG and CL.  
 Customer Screening - Java, VBS...

● **Programmer**

sty 2008 – maj 2008 - 5 mies.

ICBS (International Comprehensive Banking System) modifications - RPG and CL.



# n8n (n-eight-n)

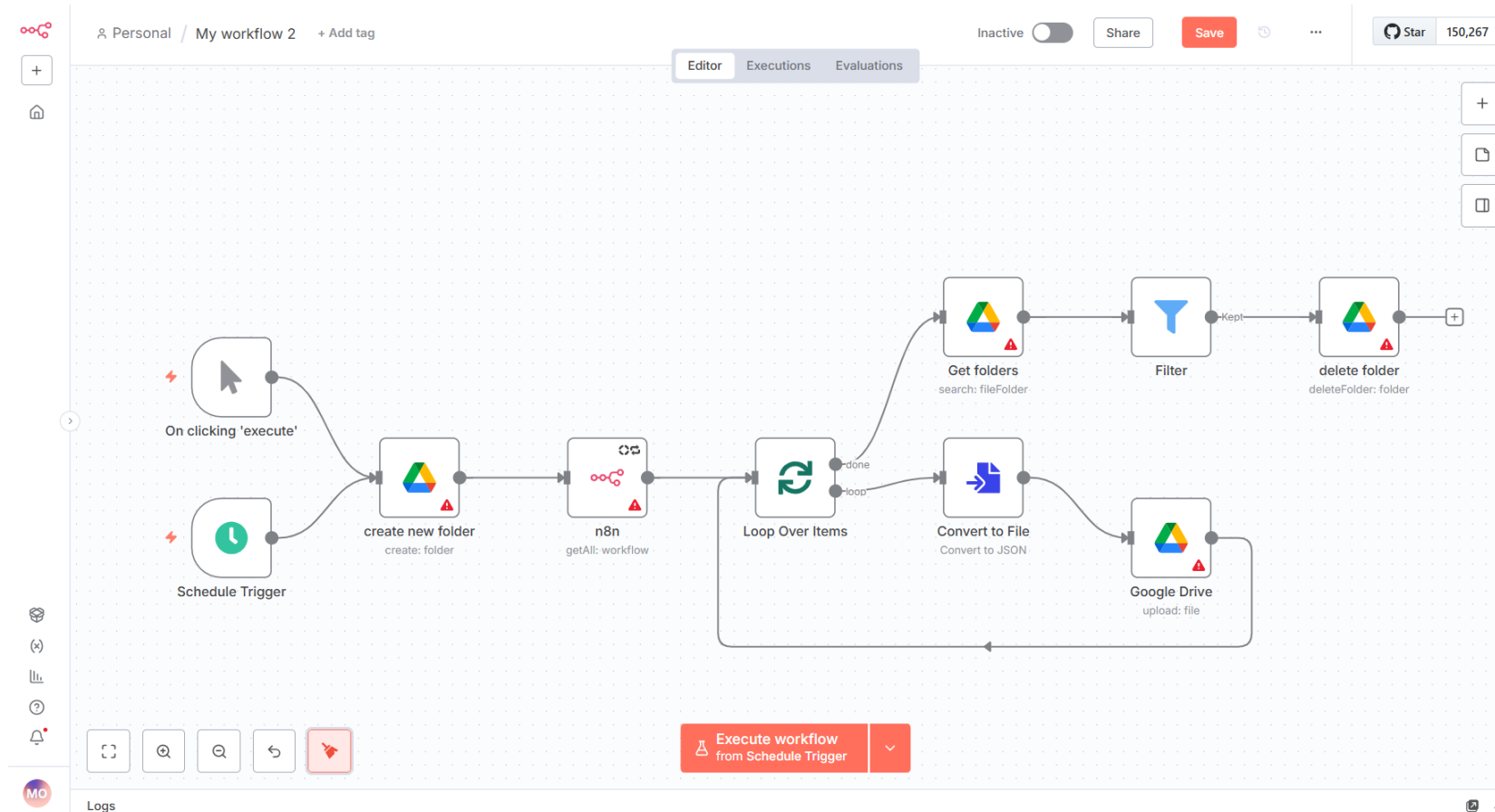
## Process automation without coding and limitations

- Visual tool for task and process automation ("low-code").
- Allows connecting hundreds of different applications and services.
- Works based on "nodes" – each node is one step in your process.
- It is an open-source solution, which means you can host it on your own server.
- Giving AI "hands": The AI model itself can only generate text. n8n allows it to act.
- Rapid prototyping: You can quickly build and test ideas for AI automations.



# n8n

## Process automation without coding and limitations



# Mapepire

## Modern database access layer

- Easy installation and management ( runs as a simple Daemon).
- Optimized for access speed.
- Allows applications written in .NET Core, Node.js or Python to directly access data on IBM i.
- Facilitates the creation of new applications (e.g., Web or microservices) using the existing "legacy" database without complex configuration.



# Mapepire

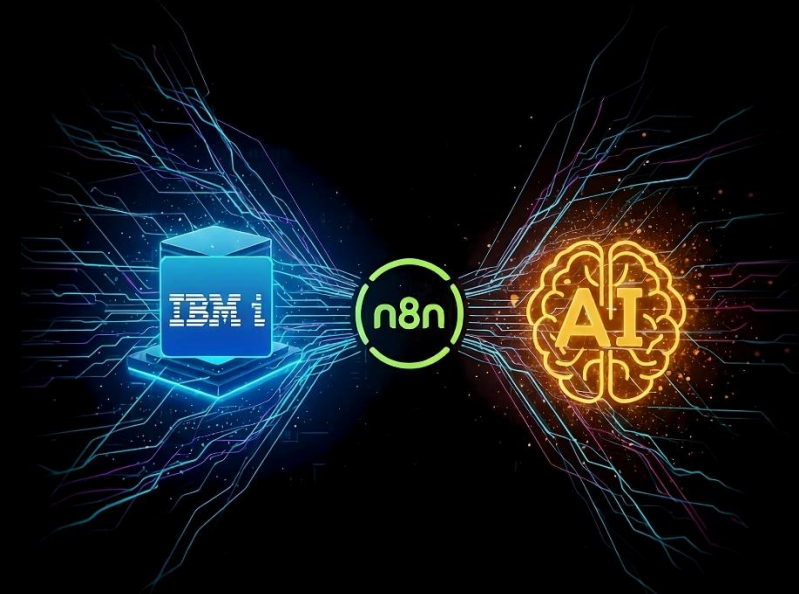
Modern database access layer

	JDBC	ODBC	Mapepire
Needs only a single port			✓
Data is always encrypted			✓
Manageable via system exit points	✓	✓	✓
Enhanced CCSID support	✓		✓
Runs in WatsonX.ai Jupyter notebooks			✓
Runs in lightweight containers (for instance Alpine Linux)	✓		✓
Directly supports multiple client languages			✓

# Integration with IBM i: n8n-nodes-ibmi-mapepire

Easy integration and automation of tasks on the IBM i system.

- This is an additional, community node for n8n.
- It was created specifically for communication with IBM i systems.
  - Running SQL commands
  - Executing CL commands
- Acts as a bridge (client) connecting n8n with the Mapepire server (@ibm/mapepire-js), which has direct access to the IBM i system.
- Can automatically paginate results.
- A simple tool for "exposing" logic and data from the IBM i system to modern applications and processes





# Installation

AI + n8n + IBM i

The screenshot shows the n8n web interface. On the left is a sidebar with navigation options: Settings, Usage and plan, Personal, Users, n8n API, External Secrets, Environments, SSO, LDAP, Log Streaming, and Community nodes (highlighted with a red box). The main area displays 'Community nodes' with an 'Install' button in the top right. A modal dialog titled 'Install community nodes' is open in the center. It contains a search prompt 'Find community nodes to add on the npm public registry. [More info](#)' with a 'Browse' button. Below is a text input field for 'npm Package Name' containing 'n8n-nodes-ibmi-mapepire'. At the bottom, there is a checked checkbox for 'I understand the risks of installing unverified code from a public source' with a 'More info' link, and an 'Install' button.

# Credentials

AI + n8n + IBM i

 Mapepire Server account 1  
Mapepire Server 

---

Connection **Sharing** Details

Host \*

Port \*

User \*

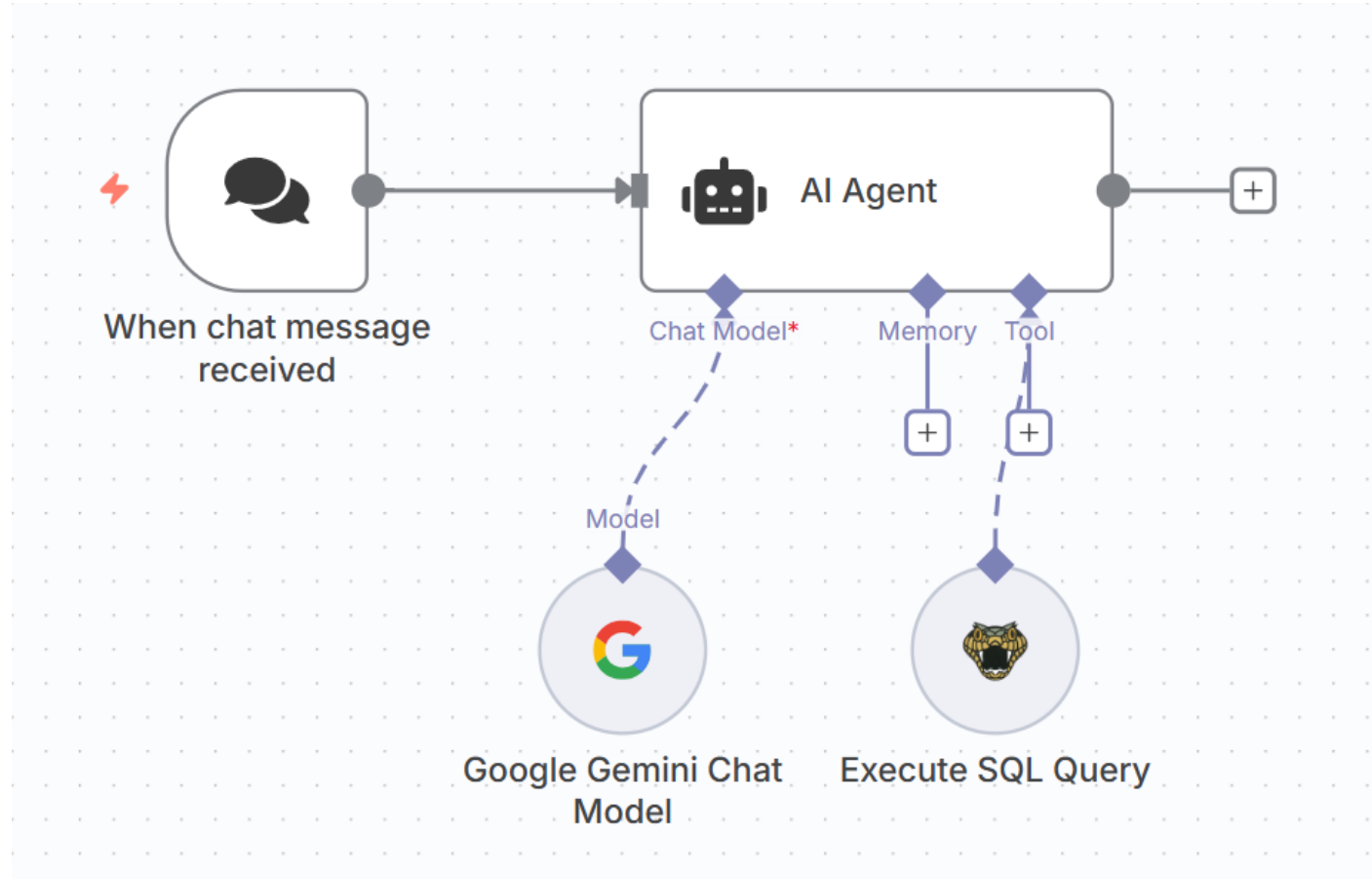
Password \*

Ignore Unauthorized TLS

CA Certificate (PEM)

# Workflow - querying the database

AI + n8n + IBM i



# Prompt

AI + n8n + IBM i




The screenshot shows the configuration interface for an AI Agent. At the top left, it says "AI Agent" with a robot icon. To the right is a red button labeled "Execute step". Below this are two tabs: "Parameters" (selected) and "Settings". A "Docs" link with an external icon is also present. The main content area is titled "Source for Prompt (User Message)" and has a dropdown menu set to "Define below". Below this is a text area titled "Prompt (User Message)" containing the following text:

```
You are end user helper. You need to write and run complex queries in DB2 on IBMi aligned with user requests to reduce query number run. Basing on available tools aggregate data and response to user in easy to read manner. Make sure every table uses MOGON1 schema prefix. Don't share schema name with user.
```

At the bottom of the text area, there is a placeholder for a user request: `User request:{{ $json.chatInput }}`. A small icon in the bottom right corner of the text area suggests a copy or paste function.

# Model and tools

AI + n8n + IBM i


 Google Gemini Chat Model

**Parameters** Settings

Credential to connect with  
Google Gemini(PaLM) Api account

Model  
models/gemini-2.5-flash

Options  
No properties  
Add Option

 Execute SQL Query Execute step

**Parameters** Settings Docs

Credential to connect with  
Mapepire Server account 1

Description  
Get all the data from DB2 on IBM i, make sure you append the tables in MOGON1 schema. Don't use ';' at the end.

Mode  
SQL

SQL  
`fx {{ $fromAI("sql_query", "SQL Query") }}`

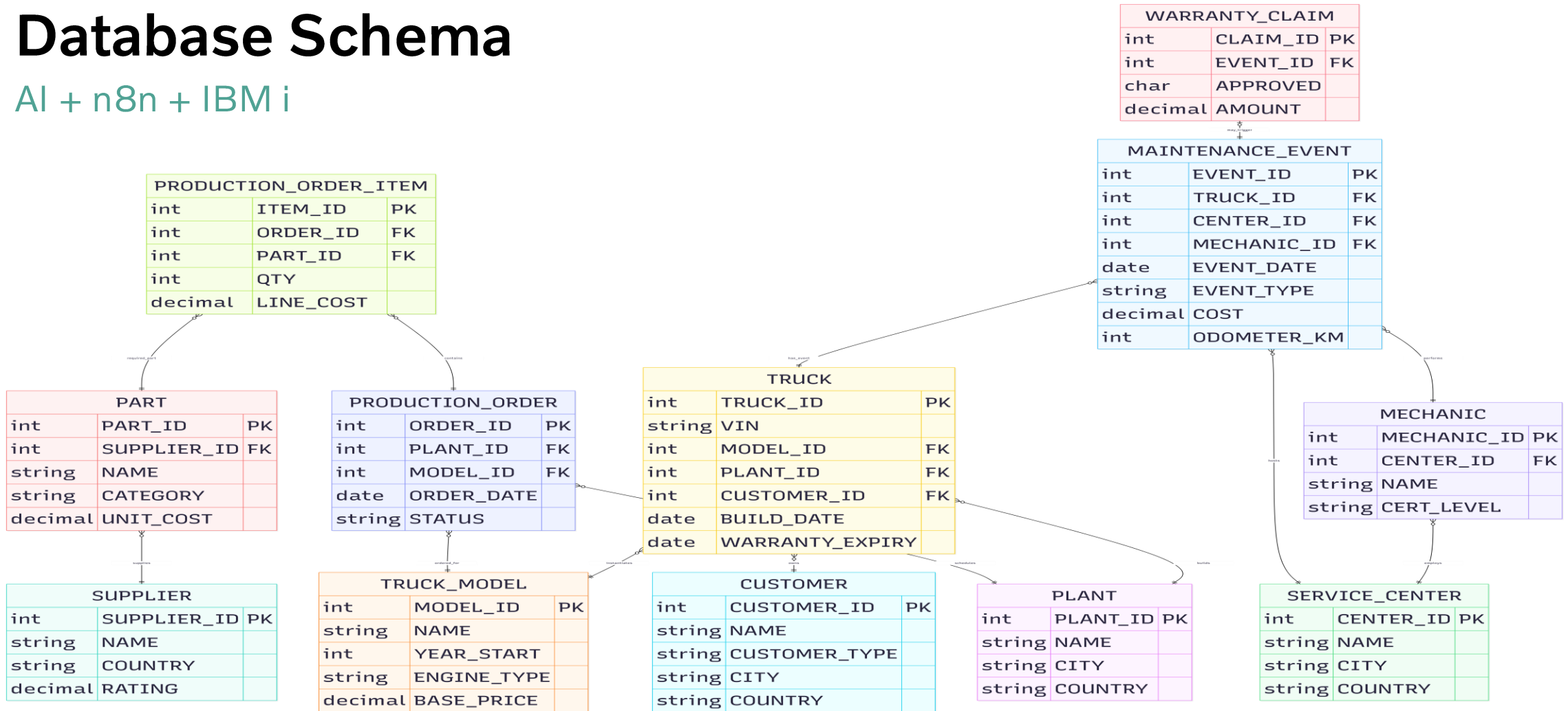
undefined

Fetch Size  
100

Additional Fields  
No properties  
Add Field

# Database Schema

AI + n8n + IBM i



# Implementation

AI + n8n + IBM i

ChatConnect [Truck Database](#) Admin Assistant

### Truck Database Chat

Ask questions and get information from the truck database.

**Hi there!** 🙌

Start a chat. We're here to help you 24/7.

Hi there! 🙌

My name is Nathan. How can I assist you today?

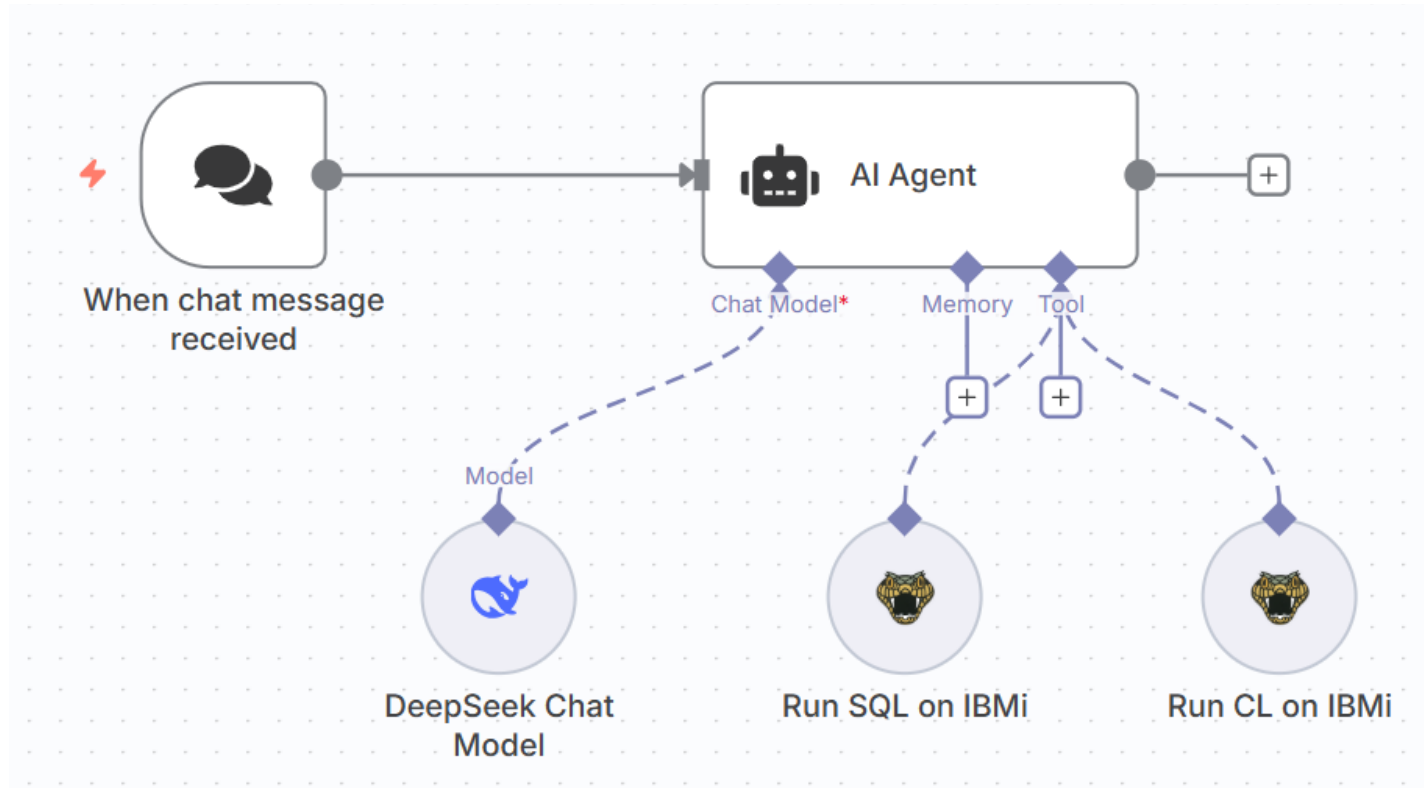
For the last 3 years, show the top 10 truck models by number of maintenance events — return model name and event count.

For the last 3 years, the top 10 truck models by number of maintenance events are:

- Model 221: 8 events
- Model 238: 8 events
- Model 214: 8 events
- Model 783: 8 events
- Model 989: 8 events
- Model 463: 8 events


# Workflow - running commands

AI + n8n + IBM i



# Tool Configuration

AI + n8n + IBM i

 Run CL on IBMi [Execute step](#)

[Parameters](#) [Settings](#) [Docs](#)

Credential to connect with  
Mapepire Server account 2

Description  
Run CL command on IBM i

Mode  
CL Command

CL Command  
`fx {{ $fromAI("cl_command", "CL Command") }}`

undefined

Additional Fields

Terse Results

Include Metadata

Add Field

# Query Analysis

AI + n8n + IBM i

Logs Clear execution When chat message received Success in 0s

Success in 36.673s | 33,556 Tokens

- When chat message received
- AI Agent
  - DeepSeek Chat Model
  - Run SQL on IBMi
  - DeepSeek Chat Model
  - Run SQL on IBMi
  - DeepSeek Chat Model

OUTPUT 1 item

sessionId	action	chatInput
a4b08483b47e4aa882d721b5c8487e3b	sendMessage	List all active job

# Query Analysis

AI + n8n + IBM i

Logs 🗑️ Clear execution
DeepSeek Chat Model Success in 4.636s | 410 Tokens

Input Output ⋮ ▼

Success in 36.673s | 33,556 Tokens

- 🗨️ When chat message received
- 🤖 AI Agent
  - 🦋 DeepSeek Chat Model
  - 🦋 Run SQL on IBMi
  - 🦋 DeepSeek Chat Model
  - 🦋 Run SQL on IBMi
  - 🦋 DeepSeek Chat Model

INPUT
🔍 ☰ 🔊 📄 {} 1 item

Human: You are supporting tool. Choose best CL commands and SQL statements for IBM i release: 7.5 to finalize task and execute like on ACS Run SQL (or STRSQL / ODBC) on user with no \*ALLOBJ. Optimize and minimize SQL and command output if not needed. For every table function use TABLE(table function) to get data like SELECT \* FROM TABLE(QSYS2.OBJECT\_STATISTICS('MJATST','JRN')) AS X.\n\nQuestion: List all active job

OUTPUT
🔍 ☰ 🔊 📄 {} 1 item

```

[
  {
    "response": {
      "generations": [
        [
          {
            "text": "",
            "generationInfo": {
              "finish_reason": "tool_calls"
            }
          }
        ]
      ]
    },
    "tokenUsage": {
      "completionTokens": 41,
      "promptTokens": 369,
      "totalTokens": 410
    }
  }
]

```

# Query Analysis

AI + n8n + IBM i

The screenshot displays a workflow execution interface for an AI agent. On the left, a 'Logs' sidebar shows a sequence of events: 'When chat message received', 'AI Agent', and a list of 'DeepSeek Chat Model' and 'Run SQL on IBMi' actions. The main area is titled 'Run SQL on IBMi' and shows a successful execution in 6.593s. The 'INPUT' section contains a SQL query: `SELECT * FROM TABLE(QSYS2.ACTIVE_JOB_INFO(JOB_NAME _FILTER => '%')) AS X`. The 'OUTPUT' section shows a JSON response with metadata for the query, including column count (125), job name ('431323/QUSER/QZDASOINIT'), and column details for 'ORDINAL\_POSITION'.

Logs Clear execution **Run SQL on IBMi** Success in 6.593s Input Output ...

Success in 36.673s | 33,556 Tokens

When chat message received

AI Agent

- DeepSeek Chat Model
- Run SQL on IBMi**
- DeepSeek Chat Model
- Run SQL on IBMi
- DeepSeek Chat Model

**INPUT** 1 item

```
SQL SELECT * FROM
TABLE(QSYS2.ACTIVE_JOB_INFO(JOB_NAME
_FILTER => '%')) AS X
```

**OUTPUT** 1 item

```
{
  "response": [
    {
      "rows": [
      ],
      "metadata": {
        "column_count": 125,
        "job": "431323/QUSER/QZDASOINIT",
        "columns": [
          {
            "name": "ORDINAL_POSITION",
            "type": "INTEGER",
            "display_size": 11,
            "label": "ORDINAL_POSITION",
            "precision": 10,
            "scale": 0,
            "autoIncrement": false,
            "nullable": 1,
            "readOnly": true,
            "writeable": false
          },
          {
            "name": "JOB_NAME"
          }
        ]
      }
    }
  ]
}
```

# Query Analysis

AI + n8n + IBM i

Logs Clear execution
DeepSeek Chat Model Success in 5.365s | 7,622 Tokens
Input
Output
⋮

Success in 36.673s | 33,5
Overview Details

- When chat message received
- AI Agent
  - DeepSeek Chat Model
  - Run SQL on IBMi
  - DeepSeek Chat Model ✎
  - Run SQL on IBMi
  - DeepSeek Chat Model

INPUT	1 item
<p>Human: You are supporting tool. Choose best CL commands and SQL statements for IBM i release: 7.5 to finalize task and execute like on ACS Run SQL (or STRSQL / ODBC) on user with no *ALLOBJ. Optimize and minimize SQL and command output if not needed. For every table function use TABLE(table function) to get data like SELECT * FROM TABLE(QSYS2.OBJECT_STATISTICS('MJATST','JRN')) AS X.\n\nQuestion: List all active job\nAI: \n Tool: [{"rows": [], "metadata": [{"column_count": 125, "job": "431323/QUSER/QZDASOINIT", "col": [{"name": "ORDINAL_POSITION", "type": "INTEGER", "display_size": 10, "label": "ORDINAL_POSITION", "display_size": 10}, {"name": "JOB_NAME", "type": "VARCHAR", "display_size": 28, "label": "JOB_NAME", "display_size": 28}, {"name": "JOB_NAME_SHORT", "type": "VARCHAR", "display_size": 10, "label": "JOB_NAME_SHORT", "display_size": 10}, {"name": "JOB_USER", "type": "VARCHAR", "display_size": 10, "label": "JOB_USER", "display_size": 10}, {"name": "JOB_NUMBER", "type": "VARCHAR", "display_size": 6, "label": "JOB_NUMBER", "display_size": 6}, {"name": "INTERNAL_JOB_ID", "type": "BINARY", "display_size": 10, "label": "INTERNAL_JOB_ID", "display_size": 10}, {"name": "SUBSYSTEM", "type": "VARCHAR", "display_size": 10, "label": "SUBSYSTEM", "display_size": 10}, {"name": "SUBSYSTEM_LIBRARY_NAME", "type": "VARCHAR", "display_size": 10, "label": "SUBSYSTEM_LIBRARY_NAME", "display_size": 10}, {"name": "AUTHORIZATION_NAME", "type": "VARCHAR", "display_size": 10, "label": "AUTHORIZATION_NAME", "display_size": 10}, {"name": "JOB_TYPE", "type": "VARCHAR", "display_size": 3, "label": "JOB_TYPE", "display_size": 3}, {"name": "FUNCTION_TYPE", "type": "VARCHAR", "display_size": 10, "label": "FUNCTION_TYPE", "display_size": 10}, {"name": "FUNCTION", "type": "VARCHAR", "display_size": 10, "label": "FUNCTION", "display_size": 10}, {"name": "JOB_STATUS", "type": "VARCHAR", "display_size": 4, "label": "JOB_STATUS", "display_size": 4}, {"name": "MEMORY_POOL", "type": "VARCHAR", "display_size": 9, "label": "MEMORY_POOL", "display_size": 9}, {"name": "RIIPRIORITY", "type": "INTEGER", "display_size": 11, "label": "RIIPRIORITY", "display_size": 11}]}]</p>	<p>OUTPUT</p> <p style="font-size: 0.8em;">1 item</p> <p>The query returned an empty result set, which suggests there might be an issue with the filter or the table function. Let me try a more direct approach using a simpler SQL query:</p>

# Query Analysis

AI + n8n + IBM i

Logs
Clear execution

DeepSeek Chat Model
Success in 5.365s
7,622 Tokens

Input
Output
...

Success in 36.673s | 33,556 Tokens

- When chat message received
- AI Agent
  - DeepSeek Chat Model
  - Run SQL on IBMi
  - DeepSeek Chat Model
  - Run SQL on IBMi
  - DeepSeek Chat Model

**INPUT** 1 item

Human: You are supporting tool. Choose best CL commands and SQL statements for IBM i release: 7.5 to finalize task and execute like on ACS Run SQL (or STRSQL / ODBC) on user with no \*ALLOBJ. Optimize and minimize SQL and command output if not needed. For every table function use TABLE(table function) to get data like SELECT \* FROM TABLE (QSYS2.OBJECT\_STATISTICS('MJATST ',JRN') ) AS X.\n\n Question: List all active job\n\n AI: \n\n Tool: [{"rows": [], "metadata": {"column\_count": 125, "job": "431323/QUSER/QZDASOINIT", "columns": [{"name": "ORDINAL\_POSITION", "type": "INTEGER", "display\_size": 6, "label": "ORDINAL\_POSITION", "display\_size": 6}, {"name": "JOB\_NAME", "type": "VARCHAR", "display\_size": 28, "label": "JOB\_NAME", "display\_size": 28}, {"name": "JOB\_NAME\_SHORT", "type": "VARCHAR", "display\_size": 10, "label": "JOB\_NAME\_SHORT", "display\_size": 10}, {"name": "JOB\_USER", "type": "VARCHAR", "display\_size": 10, "label": "JOB\_USER", "display\_size": 10}, {"name": "JOB\_NUMBER", "type": "VARCHAR", "display\_size": 6, "label": "JOB\_NUMBER", "display\_size": 6}, {"name": "INTERNAL\_JOB\_ID", "type": "BINARY", "display\_size": 10, "label": "INTERNAL\_JOB\_ID", "display\_size": 10}, {"name": "SUBSYSTEM", "type": "VARCHAR", "display\_size": 10, "label": "SUBSYSTEM", "display\_size": 10}, {"name": "SUBSYSTEM\_LIBRARY\_NAME", "type": "VARCHAR", "display\_size": 10, "label": "SUBSYSTEM\_LIBRARY\_NAME", "display\_size": 10}, {"name": "AUTHORIZATION\_NAME", "type": "VARCHAR", "display\_size": 10, "label": "AUTHORIZATION\_NAME", "display\_size": 10}, {"name": "JOB\_TYPE", "type": "VARCHAR", "display\_size": 3, "label": "JOB\_TYPE", "display\_size": 3}, {"name": "FUNCTION\_TYPE", "type": "VARCHAR", "display\_size": 10, "label": "FUNCTION\_TYPE", "display\_size": 10}, {"name": "JOB\_STATUS", "type": "VARCHAR", "display\_size": 4, "label": "JOB\_STATUS", "display\_size": 4}, {"name": "MEMORY\_POOL", "type": "VARCHAR", "display\_size": 9, "label": "MEMORY\_POOL", "display\_size": 9}, {"name": "RUN\_PRIORITY", "type": "INTEGER", "display\_size": 11, "label": "RUN\_PRIORITY", "display\_size": 11}, {"name": "THREAD\_COUNT", "type": "INTEGER", "display\_size": 11, "label": "THREAD\_COUNT", "display\_size": 11}]}]

**OUTPUT** 1 item

```
[
  {
    "response": {
      "generations": [
        [
          {
            "text": "The query returned an empty result set, which suggests there might be an issue with the filter or the table function. Let me try a more direct approach using a simpler SQL query:",
            "generationInfo": {
              "finish_reason": "tool_calls"
            }
          }
        ]
      ]
    },
    "tokenUsage": {
      "completionTokens": 84,
      "promptTokens": 7538,
      "totalTokens": 7622
    }
  }
]
```

# Query Analysis

AI + n8n + IBM i

The screenshot displays the n8n workflow execution interface. On the left, the 'Logs' panel shows a sequence of steps: 'When chat message received', 'AI Agent', and a series of 'DeepSeek Chat Model' and 'Run SQL on IBMi' steps. The 'Run SQL on IBMi' step is currently selected and highlighted. The main workspace is divided into three sections: 'INPUT', 'OUTPUT', and 'EXECUTION LOGS'. The 'INPUT' section shows the SQL query: `SELECT JOB_NAME, JOB_USER, JOB_NUMBER, JOB_STATUS FROM TABLE(QSYS2.ACTIVE_JOB_INFO()) AS X`. The 'OUTPUT' section displays the JSON response from the query, which is a list of three rows of job information. The 'EXECUTION LOGS' section shows the execution status as 'Success in 1.793s'.

Logs Clear execution Run SQL on IBMi Success in 1.793s Input Output ...

Success in 36.673s | 33,556 Tokens

When chat message received

AI Agent

- DeepSeek Chat Model
- Run SQL on IBMi
- DeepSeek Chat Model
- Run SQL on IBMi
- DeepSeek Chat Model

INPUT 1 item

```
SQL SELECT JOB_NAME, JOB_USER,
JOB_NUMBER, JOB_STATUS FROM
TABLE(QSYS2.ACTIVE_JOB_INFO()) AS X
```

OUTPUT 1 item

```
[
  {
    "response": [
      {
        "rows": [
          {
            "JOB_NAME": "000000/QSYS/SCPF",
            "JOB_USER": "QSYS",
            "JOB_NUMBER": "000000",
            "JOB_STATUS": "EVTW"
          },
          {
            "JOB_NAME": "128669/QSYS/QSYSARB3",
            "JOB_USER": "QSYS",
            "JOB_NUMBER": "128669",
            "JOB_STATUS": "EVTW"
          },
          {
            "JOB_NAME": "128675/QSYS/QDBSRV01",
            "JOB_USER": "QSYS",
            "JOB_NUMBER": "128675",
            "JOB_STATUS": "EVTW"
          }
        ]
      }
    ]
  }
]
```

# Query Analysis

AI + n8n + IBM i

Logs Clear execution
DeepSeek Chat Model Success in 17.36s | 25,524 Tokens

Input Output ... ▼

Success in 36.673s | 33,556 Tokens

- When chat message received
- AI Agent
  - DeepSeek Chat Model
  - Run SQL on IBMi
  - DeepSeek Chat Model
  - Run SQL on IBMi
  - DeepSeek Chat Model

**INPUT** 1 item

Human: You are supporting tool. Choose best CL commands and SQL statements for IBM i release: 7.5 to finalize task and execute like on ACS Run SQL (or STRSQL / ODBC) on user with no \*ALLOBJ. Optimize and minimize SQL and command output if not needed. For every table function use TABLE(table function) to get data like SELECT \* FROM TABLE (QSYS2.OBJECT\_STATISTICS('MJATST ','JRN') ) AS X.\n\n Question: List all active job\n AI: \n Tool: [{"rows": [], "metadata": {"column\_count": 125, "job": "431323/QUSER/QZDASOINIT", "col": [{"name": "ORDINAL\_POSITION", "type": "INTEGER", "display\_size": 10, "label": "Ordinal Position", "display": "Ordinal Position"}, {"name": "JOB\_NAME", "type": "VARCHAR", "display\_size": 28, "label": "Job Name", "display": "Job Name"}, {"name": "JOB\_NAME\_SHORT", "type": "VARCHAR", "display\_size": 10, "label": "Job Name Short", "display": "Job Name Short"}, {"name": "JOB\_USER", "type": "VARCHAR", "display\_size": 10, "label": "Job User", "display": "Job User"}, {"name": "JOB\_NUMBER", "type": "VARCHAR", "display\_size": 6, "label": "Job Number", "display": "Job Number"}, {"name": "INTERNAL\_JOB\_ID", "type": "BINARY", "display\_size": 11, "label": "Internal Job ID", "display": "Internal Job ID"}, {"name": "SUBSYSTEM", "type": "VARCHAR", "display\_size": 10, "label": "Subsystem", "display": "Subsystem"}, {"name": "SUBSYSTEM\_LIBRARY\_NAME", "type": "VARCHAR", "display\_size": 10, "label": "Subsystem Library Name", "display": "Subsystem Library Name"}, {"name": "AUTHORIZATION\_NAME", "type": "VARCHAR", "display\_size": 10, "label": "Authorization Name", "display": "Authorization Name"}, {"name": "JOB\_TYPE", "type": "VARCHAR", "display\_size": 3, "label": "Job Type", "display": "Job Type"}, {"name": "FUNCTION\_TYPE", "type": "VARCHAR", "display\_size": 10, "label": "Function Type", "display": "Function Type"}, {"name": "FUNCTION", "type": "VARCHAR", "display\_size": 10, "label": "Function", "display": "Function"}, {"name": "JOB\_STATUS", "type": "VARCHAR", "display\_size": 4, "label": "Job Status", "display": "Job Status"}, {"name": "MEMORY\_POOL", "type": "VARCHAR", "display\_size": 9, "label": "Memory Pool", "display": "Memory Pool"}, {"name": "RUN\_PRIORITY", "type": "INTEGER", "display\_size": 11, "label": "Run Priority", "display": "Run Priority"}, {"name": "THREAD\_COUNT", "type": "INTEGER", "display\_size": 11, "label": "Thread Count", "display": "Thread Count"}]}

**OUTPUT** 1 item

I've successfully retrieved the list of all active jobs on your IBM i system. Here's a summary of what was found:

### Active Jobs Summary

The query returned **333 active jobs** currently running on your IBM i 7.5 system. The results show:

- Job Name** (format: job\_number/user/job\_name)
- Job User** (user profile that owns the job)
- Job Number** (system-assigned job identifier)
- Job Status** (current status of each job)

**Key Observations:**

- System Jobs:** Many system-level jobs from QSYS, QUSER, and other system users
- Batch Jobs:** Several QBATCH jobs in DEQW status (waiting on job queue)
- Interactive Jobs:** Multiple QINTER jobs and user sessions
- Database Jobs:** QDBSRV\* jobs for database services
- Network Services:** TCP/IP jobs (QTCP), web services (QTMHHTTP), and other network services
- User Sessions:** Various user interactive sessions (QPAD\* jobs) in DSPW status (display wait)

# References

- [AI Workflow Automation Platform & Tools - n8n](#)
- [Docker | n8n Docs](#)
- [Mapepire: cloud-friendly IBM i database access layer, built with simplicity and performance in-mind](#)
- [marcin-ogon/n8n-nodes-ibmi-mapepire: n8n community nodes to run IBM i SQL and CL commands via Mapepire](#)
- [PUB400.COM - Your public IBM i server](#)

**V O L V O**