Microsoft SQL ML Services with Intel Optimized AI on Lenovo ThinkSystem SR650 V3

Solution Brief

In-database ML

In-database machine learning integrates machine learning directly into SQL databases, allowing all related processes such as selecting data, training, and model evaluation to occur within the database itself on the same node. This configuration allows organizations to handle complex analyses directly in their databases without transferring data between systems, avoiding delays and security risks associated with exposed data movement.

Lenovo Solutions for Microsoft SQL Server on ThinkSystem SR650 V3 are optimized for AI workloads, Online Transaction Processing (OLTP) and Data Warehouse (DW) which are **Accelerated by Intel** offerings. This technical brief features Microsoft SQL Server 2022 running on a high-performance Lenovo dual-socket 2U rack mount enterprise server. The server can support both 4th and 5th Generation Intel® Xeon® Scalable processors, TruDDR5 4800MHz memory and P5620 NVMe drives among a variety of storage options, including support for the PCIe 5.0 standard devices for I/O. The Intel 5th generation processors support up to 64 cores, 385 watts and 96GB 5600 speed memory.

The SR650 V3 server is a storage dense offering, with up to 40 2.5" drive bays in the front, middle and rear of the server and 5 different slot configurations at the rear of the server. Onboard NVMe PCIe ports allow direct connections to 16 NVMe SSDs, which frees up PCIe slots and lowers NVMe solution acquisition costs.

Microsoft SQL Server and in-database ML

Microsoft SQL Server Machine Learning Services (MSSQL ML Services) provides a platform for developing and deploying machine learning models directly within the SQL Server database environment. This integration allows for simple execution of Python and R scripts with relational data, enabling predictive analytics and machine learning capabilities directly on the data without the need for data movement. [ML Services docs]

To start the journey with in-database ML, there are simple instructions to follow for installation of MSSQL ML Services on MSSQL Serves 2022 with Python - Install MSSQL ML .

After installation, you can see that next to SQL Server service there is SQL Launchpad service. It is responsible for managing Python runtime next to SQL database functions. When a user submits a script for execution, MSSQL Server communicates with the Launchpad service and the latter initiates the appropriate runtime in an isolated process. That helps to secure database functionality and integrity in case of script execution failure.

Integration with Intel optimizations

Integrating MSSQL ML Services with Intel optimizations on Windows platforms can enhance the performance of machine learning workflows on Intel CPUs. These optimizations leverage Intel's hardware and software technologies to accelerate data analytics and machine learning tasks, making them more beneficial for enterprises that rely on SQL Server for their data management and analytics needs.

The evaluation of use cases was processed on the following setup:

- Windows Server 2022 Datacenter Evaluation system,
- Lenovo SR650 V3 server with 2 processors INTEL(R) XEON(R) GOLD 6548Y+ (code name Emerald Rapids) with AMX technology
- MSSQL Server 2022 CU15

To introduce Intel optimizations into ML workloads we decided to use Optimum-Intel. This is a tool from Hugging Face designed to optimize and accelerate the performance of machine learning models, specifically transformer models, on Intel hardware. It enhances the efficiency of these models, making them faster and more cost-effective to run on Intel devices. The installation process is in Appendix 2. From Optimum-Intel, we use Openvino which is open-source toolkit that enables high performance inference capabilities for Intel CPUs, GPUs, and special DL inference accelerators.

Use case 1 - Product brochure (advertisement) creation from product details

The idea of this use case is to get the product and its description from data that might be stored in SQL database and use all that information as an input for a text-generation AI model. Based on the requirements that should be included in the prompt for the gen AI model we expect to receive an advertisement brochure plan for the specific product.

The use case is based on the Amazon Sales Dataset from Kaggle. It contains a solid package of Amazon products with their descriptions and details as well as with reviews from users. The dataset was uploaded to a newly created MSSQL database.

The model chosen for this case is Qwen2-7B-Instruct from well-known and well tested Qwen2 family of Gen AI models. The model is stored on the same machine as the SQL Server. To download and save the Qwen2 model we used the script written in a T-SQL (Transact SQL) language and we ran it in SQL Server Management Studio (SSMS) – see Figure 1.

```
EXECUTE sp_execute_external_script
  @language = N'Python',
  @script = N'
import os
os.environ["HF_HOME"] = r"C:\\ProgramData\\Python310\\cache_model"

from optimum.intel import OVModelForCausalLM

model_id = "Qwen/Qwen2-7B-Instruct"
model = OVModelForCausalLM.from_pretrained(model_id, export=True, load_in_8bit=False)
model.save_pretrained("C:\\ProgramData\\Python310\\cache_model\\openvino\\Qwen2-7B-Instruct-
BF16")
'
```

Figure 1 Load model in Openvino format with Intel optimizations

The analytics procedure is written in a T-SQL script and can be run from SSMS. The script permits access to required data from the database and runs directly on the gen AI analytics using the Python script – see Figure 2. The results could also be loaded into another table in the database within the same T-SQL script.

```
EXECUTE sp_execute_external_script
  @language = N'Python',
  @input_data_1 = N'SELECT TOP (3) [product_name],[about_product] FROM
[amazon_products].[dbo].[amazon] ORDER BY NEWID()',
  @script = N'
import os
os.environ["ONEDNN_VERBOSE"] = "1"
os.environ["HF_HOME"] = r"C:\ProgramData\Python310\cache_model"

import pandas as pd
from transformers import AutoTokenizer, pipeline
from optimum.intel import OVModelForCausalLM

model_id = "Qwen/Qwen2-7B-Instruct"
model_path = "C:\\ProgramData\\Python310\\cache_model\\openvino\\Qwen2-7B-Instruct-BF16"

model = OVModelForCausalLM.from_pretrained(model_path)
tokenizer = AutoTokenizer.from_pretrained(model_id)
gen_pipeline = pipeline("text-generation", model=model, tokenizer=tokenizer)

df = pd.DataFrame(InputDataSet)
products = df.values.tolist()

for product in products:
    product_name = product[0]
    product_description = product[1]
    prompt = "You are marketing expert. Craft a brochure for the following product " +
        product_name + ". You can use in the brochure product qualities and advantages from
        description: "+ product_description
    messages = [{"role": "user", "content": prompt}]
    text = tokenizer.apply_chat_template(messages, tokenize=False, add_generation_prompt=True)
    device = "cpu"
    model_inputs = tokenizer([text], return_tensors="pt").to(device)

    generated_ids = model.generate(model_inputs.input_ids, max_new_tokens=512, do_sample=True)

    generated_ids = [output_ids[len(input_ids):] for input_ids, output_ids in
                     zip(model_inputs.input_ids, generated_ids)]

    response = tokenizer.batch_decode(generated_ids, skip_special_tokens=True)[0]
    print(response)
'
```

Figure 2 Script to use gen AI with transformers with Openvino optimizations within SQL database

Use case 2 - Sentiment analysis of reviews

Another example of AI analytics to showcase, is sentiment analysis of product reviews. The idea is based on the reviews of users, we want to determine what the general perception of a given product is, whether positive or negative.

The use case is based on the same Amazon Sales Dataset from Kaggle, as it contains the Amazon products list and their descriptions, and a good portion of user reviews. The dataset was uploaded to a newly created MSSQL database.

The model chosen for this case is state-of-the-art for sentiments analysis DistilBERT base uncased finetuned SST-2 . The model is stored on the same machine as the SQL Server. To download and save the DistilBERT model we used the script written in a T-SQL (Transact SQL) language and we run it in SQL Server Management Studio (SSMS) – see Figure 3.

```
EXECUTE sp_execute_external_script
  @language = N'Python',
  @script = N'
import os
os.environ["ONEDNN_VERBOSE"] = "1"
os.environ["HF_HOME"] = r"C:\\ProgramData\\Python310\\cache_model"

from optimum.intel import OVModelForSequenceClassification

model_id = "distilbert/distilbert-base-uncased-finetuned-sst-2-english"
model = OVModelForSequenceClassification.from_pretrained(model_id, export=True)
model.save_pretrained("C:\\ProgramData\\Python310\\cache_model\\openvino\\distilbert-base-un-
cased-finetuned-sst-2-english")
'
```

Figure 3 Download and load DistilBERT model

The whole analytics procedure is also written in a T-SQL script, like the previous example, and can be run directly from SSMS. The script permits access to required data from the database and then runs the NLP analytics using the Python script embedded in the T-SQL structure. The results of sentiments are displayed in the terminal preview window and, as a future development, they can be loaded into another table in the database within the same T-SQL script – see Figure 4.

```
EXECUTE sp_execute_external_script
  @language = N'Python',
  @input_data_1 = N'SELECT TOP (3) [product_name],[review_title],[review_content] FROM
[amazon_products].[dbo].[amazon] ORDER BY NEWID()',
  @script = N'
import os
os.environ["ONEDNN_VERBOSE"] = "1"
os.environ["HF_HOME"] = r"C:\ProgramData\Python310\cache_model"

import pandas as pd
from transformers import AutoTokenizer, pipeline
from optimum.intel import OVModelForSequenceClassification

model_id = "distilbert/distilbert-base-uncased-finetuned-sst-2-english"
model_path = "C:\\ProgramData\\Python310\\cache_model\\openvino\\distilbert-base-uncased-
finetuned-sst-2-english"

model = OVModelForSequenceClassification.from_pretrained(model_path)
tokenizer = AutoTokenizer.from_pretrained(model_id, model_max_length=1024)

sentiments_pipeline = pipeline("sentiment-analysis", model=model, tokenizer=tokenizer)

df = pd.DataFrame(InputDataSet)
reviews = df.values.tolist()
for review in reviews:
    product_name = review[0]
    review_title = review[1]
    review_content = review[2]
    sentiment = sentiments_pipeline(review_title + " " + review_content)
    print("Product: " + product_name + "\n")
    print("Title: " + review_title + "\n")
    print("Review: " + review_content + "\n")
    print(sentiment)
'
```

Figure 4 Script to use gen AI with transformers and Openvino optimizations within SQL database

Benchmarking

The team attempted a basic testing platform for optimum-intel package - the optimum-benchmark library to evaluate and compare models used in both use cases. After preliminary tests of optimum-benchmark library we concluded that the benchmarking tool was not yet mature for our test purposes. Additional testing could be done at a later date and results posted here.

Nonetheless, we present official Intel benchmarking results of 5th Gen CPUs and Qwen2 models that show the Intel Xeon delivering latency numbers that meets production use cases.

Appendix 1

Python set up

Install the latest version of Python 3.10, using the proper Windows installation package. It is important, according to instructions, to install it with access for all system users. After Python is installed, you can open Windows Terminal.

Enter home python directory, in this example it is: `C:\ProgramData\Python310`

Upgrade pip to its latest version:

```
python -m pip install -t "C:\ProgramData\Python310\Lib\site-packages"  --upgrade pip
```

So next, install ***revoscalepy*** library and packages required for that installation.

```
python -m pip install -t "C:\ProgramData\Python310\Lib\site-packages" dill
numpy==1.22.4 pandas patsy python-dateutil
```

```
python -m pip install -t "C:\ProgramData\Python310\Lib\site-packages"
https://aka.ms/sqlml/python3.10/windows/revoscalepy-10.0.1-py3-none-any.whl
```

The reason, we have to use Python 3.10 is the revoscalepy package, it is accessible only for that version of Python.

Access for SQL Launchpad

Once all the libraries are in place, we need to manage access to folders with python packages. We use Integrity Control Access Control List command that can display and modify an access-control list of permissions to specified folder for Windows Server.

```
icacls "C:\ProgramData\Python310\Lib\site-packages" /grant "NT
Service\MSSQLLAUNCHPAD":(OI)(CI)RX /T
```

```
icacls "C:\ProgramData\Python310\Lib\site-packages" /grant *S-1-15-2-1:(OI)(CI)RX /T
```

And then we need to configure and connect our Python runtime with MSSQL Server through revoscalepy registration app.

```
cd "C:\ProgramData\Python310\Lib\site-packages\revoscalepy\rxLibs"
```

```
.\RegisterRext.exe /configure /pythonhome:"C:\ProgramData\Python310"
/instance:"MSSQLSERVER"
```

After installation and registration, you need to restart MSSQL Server instance and Launchpad instance, using the SQL Server Configuration Manager.

Appendix 2

Installation of Intel optimized libraries

The commands below install the Intel Optimized libraries needed to run the use cases.

```
python -m pip install -t "C:\ProgramData\Python310\Lib\site-packages" "optimum-
intel"@git+https://github.com/huggingface/optimum-intel.git
```

```
python -m pip install -t "C:\ProgramData\Python310\Lib\site-packages" "optimum-
intel[openvino]"@git+https://github.com/huggingface/optimum-intel.git
```

```
python -m pip install -t "C:\ProgramData\Python310\Lib\site-packages" "optimum-intel[nncf]"@git+https://github.com/huggingface/optimum-intel.git

python -m pip install -t "C:\ProgramData\Python310\Lib\site-packages" --force-reinstall git+https://github.com/huggingface/optimum-benchmark.git fsspec[http]==2024.5.0 transformers==4.41.2 numpy==1.23.5 torch==2.2.0
```

Why Lenovo

Lenovo is a US$70 billion revenue Fortune Global 500 company serving customers in 180 markets around the world. Focused on a bold vision to deliver smarter technology for all, we are developing world-changing technologies that power (through devices and infrastructure) and empower (through solutions, services and software) millions of customers every day.

For More Information

To learn more about this Lenovo solution contact your Lenovo Business Partner or visit: https://www.lenovo.com/us/en/servers-storage/solutions/database/

References:

Lenovo ThinkSystem SR650 V4: SR650 V3 Detailed Information

Microsoft SQL Server 2022: https://learn.microsoft.com/en-us/sql/sql-server/what-s-new-in-sql-server-2022? view=sql-server-ver16

Related product families

Product families related to this document are the following:

- Microsoft Alliance
- Microsoft SQL Server
- **Lenovo Servers and Storage**

Trademarks

Lenovo and the Lenovo logo are trademarks or registered trademarks of Lenovo in the United States, other countries, or both. A current list of Lenovo trademarks is available on the Web at https://www.lenovo.com/us/en/legal/copytrade/.

The following terms are trademarks of Lenovo in the United States, other countries, or both:
Lenovo®
AnyBay®
ThinkSystem®
XClarity®

The following terms are trademarks of other companies:

Intel®, Intel Optane™, and Xeon® are trademarks of Intel Corporation or its subsidiaries.

Azure®, Hyper-V®, Microsoft®, SQL Server®, Windows Server®, and Windows® are trademarks of Microsoft Corporation in the United States, other countries, or both.

TPC, TPC-C, and TPC-H are trademarks of Transaction Processing Performance Council.

Other company, product, or service names may be trademarks or service marks of others.