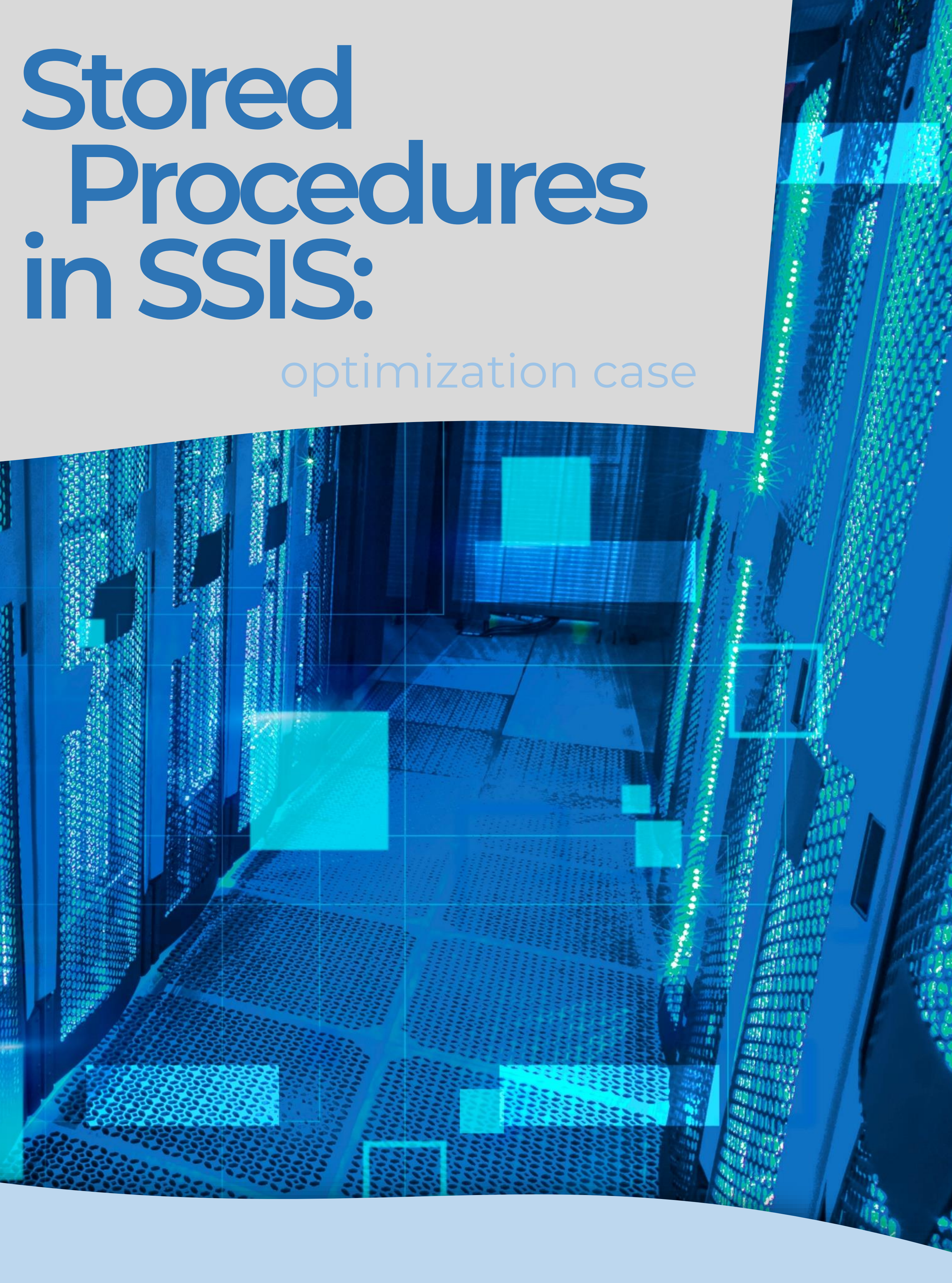
Stored Procedures in SSIS:

optimization case



Details →

By Ivan Sokolov with ❤️

THE TASK

automate the data processing using large legacy code logic

STARTING POINT

optimize the script to reduce its complexity and runtime

ORIGINAL SCRIPT		REFACTORED SCRIPT
1275	Lines of code	731
25	Temp tables	18
10	Number of outputs	6
53	Runtime, mins	31

⚠

THE PROBLEM

even after optimization, the script can be passed via neither SSRS nor SSIS

❌

SQL Server Reporting Services (SSRS)

Out of memory error

❌

SQL Server Integration Services (SSIS)

Cannot use Temp tables

THE CONCEPT

turn the refactored legacy script into a stored procedure to save its results into a staging table on the SQL server to be used as a source for the SSIS package

1

STORED PROCEDURE

Create a procedure that replicates the legacy script logic. No need to get rid of temp tables as it is executed on the SQL server side

2

STAGING TABLE

make the procedure save results into a staging table

3

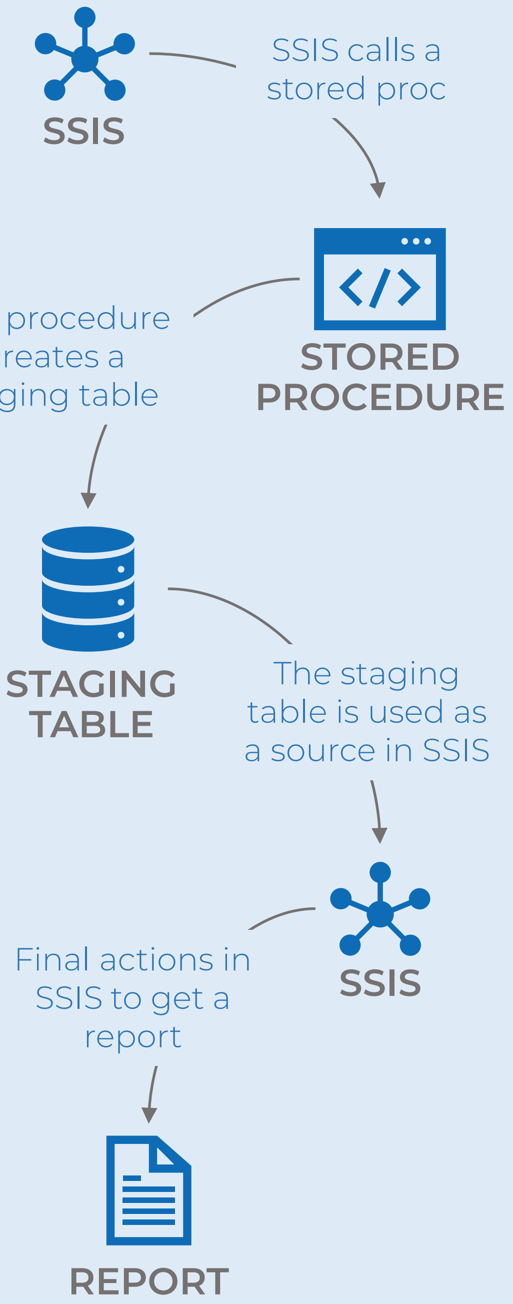
SSIS: CALL THE PROC

Use SSIS package to call the procedure. The proc saves results into a staging table. The table will be a source for further actions in SSIS

4

SSIS: FINAL ACTIONS

In the package, connect to the staging table. Use it as a source to perform actions on the SSIS side to get the desired output



SSIS

SSIS calls a stored proc

STORED PROCEDURE

The procedure creates a staging table

STAGING TABLE

The staging table is used as a source in SSIS

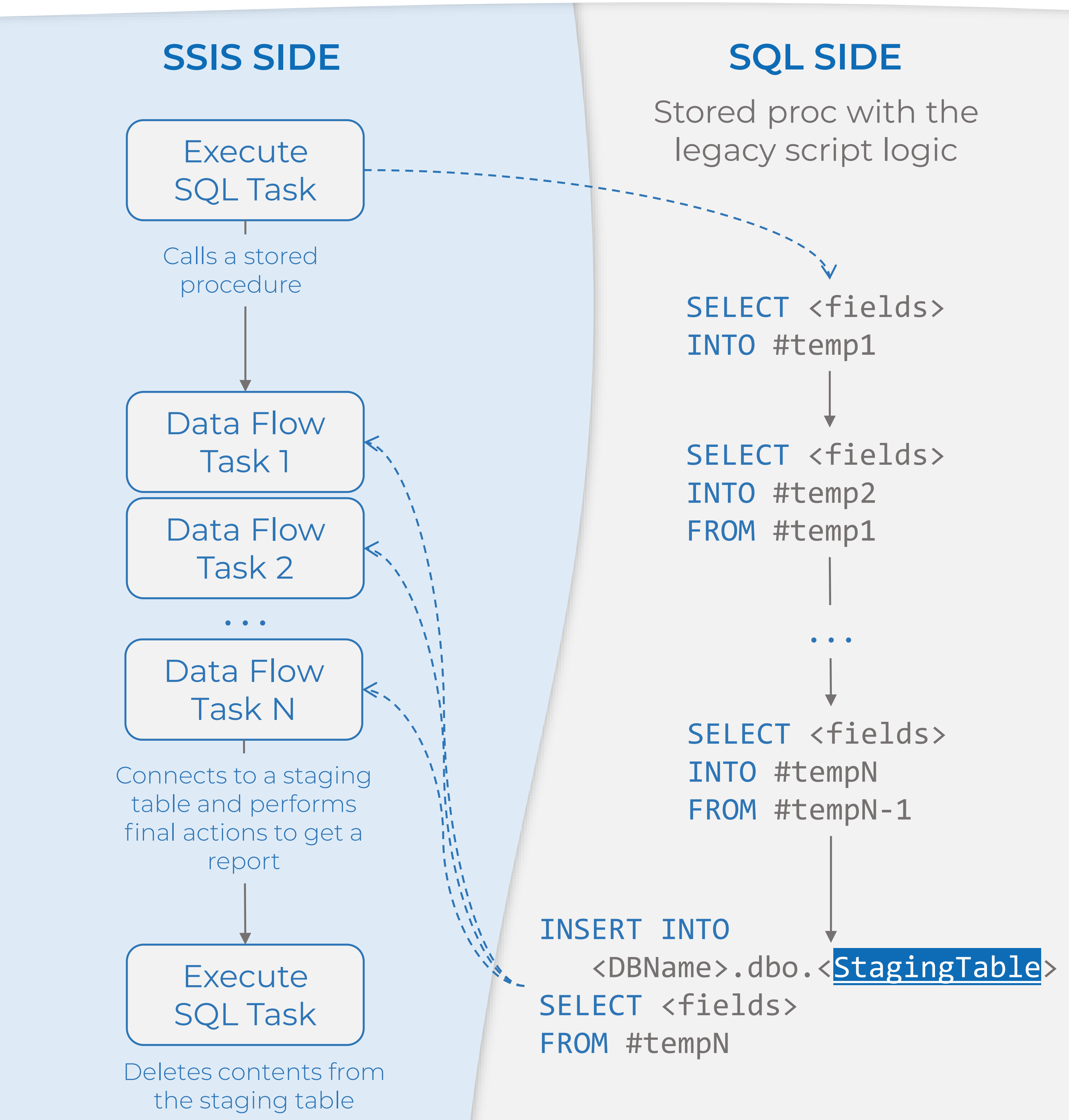
SSIS

Final actions in SSIS to get a report

REPORT

THE IDEA

an optimal combination of SSIS and SQL server capabilities may solve the issue of large data processing automation



SSIS SIDE

Execute SQL Task

Calls a stored procedure

Data Flow Task 1

Data Flow Task 2

. . .

Data Flow Task N

Connects to a staging table and performs final actions to get a report

Execute SQL Task

Deletes contents from the staging table

SQL SIDE

Stored proc with the legacy script logic

```
SELECT <fields>
INTO #temp1

SELECT <fields>
INTO #temp2
FROM #temp1

. . .

SELECT <fields>
INTO #tempN
FROM #tempN-1

INSERT INTO
   <DBName>.dbo.<StagingTable>
SELECT <fields>
FROM #tempN
```

ALTERNATIVE

alter the legacy script to be executed fully on SSIS side by replacing #temp tables with other constructions – **common table expressions** and/or **table variables**

COMMON TABLE EXPRESSION (CTE)

✓

- Works fine in SSIS
- Is sometimes faster than #temp table as it is stored in memory rather than Tempdb database

✗

- One CTE cannot be used multiple times
- Cannot create indexes on a CTE

In SSIS, CTE is good for straightforward data flow tasks where it needs to be used only once to pass its results either to another CTE or to the final output

TABLE VARIABLE

✓

- Works fine in SSIS
- Can be used multiple times, just like #temp table

✗

- Not recommended to use with large tables (greater than 100 rows)

In SSIS, table variable is good for processing simple data flow tasks with little data amount. Otherwise, you must be cautious