



# An Overview of Changes in the New Achilles (v1.6)

Ajit Londhe  
Taha Abdul-Basser  
Vojtech Huser

# Major Changes

- Dropped support for CDM v4
  - Use older versions of Achilles if CDM v4 support needed; the package will verify your data is using CDM v5, otherwise stop executing
- Added cost table support for CDM v5
  - The unified cost table can now be summarized, although this is still a very time-consuming set of analyses when using larger datasets
- Fixed several group by SQL queries
  - Impala and Netezza were unable to run these previously
- Compression of exported JSON files into 1 zip file for easier portability
- To help improve performance in massively parallel processing (MPP) database platforms like Amazon Redshift and MS Parallel Data Warehouse, we've added multi-threading support

# Multi-threading

- Previously, Achilles and Achilles Heel were singular SQL scripts that ran each of their analyses in serial by creating the final tables and inserting rows to it for each analysis or heel warning
- Inspection of these scripts led to the realization that there is no need to run each analysis one after another
- Instead, we split all queries that have no dependency up into separate files
- Using OhdsiRTools clusterApply function, we can now run each of these queries in parallel over  $n$  SQL sessions, where  $n$  can be set by the user
- Replaced all insert statements with CTAS queries; inserts are very slow on MPP systems

# Does it help?

On Amazon Redshift dense-compute clusters, using Truven CCAE (140 million patients):

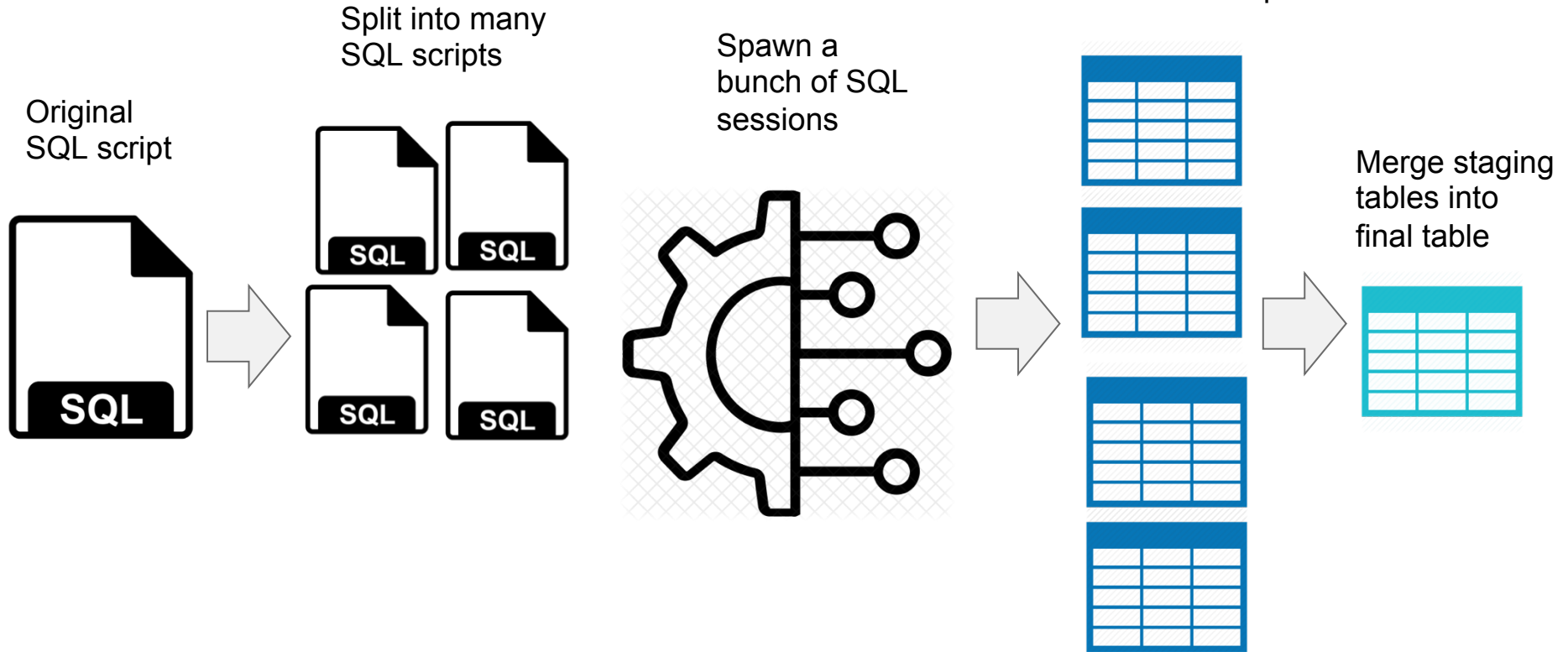
## **Achilles without Heel**

- 450 minutes to process Achilles v1.4
- 114 minutes to process Achilles v1.6 with 1 thread
- 71.2 minutes to process Achilles v1.6 with 10 threads

## **Achilles Heel Only**

- 14.8 minutes to process Achilles Heel v1.6 with 1 thread
- 10 minutes to process Achilles Heel v1.6 with 10 threads

# How does it work?



# Executing the new Achilles

```
Achilles::achilles(connectionDetails,  
  cdmDatabaseSchema,  
  oracleTempSchema = cdmDatabaseSchema,  
  resultsDatabaseSchema = cdmDatabaseSchema,  
  scratchDatabaseSchema = resultsDatabaseSchema,  
  vocabDatabaseSchema = cdmDatabaseSchema,  
  sourceName = "",  
  analysisIds,  
  createTable = TRUE,  
  smallCellCount = 5,  
  cdmVersion = "5",  
  runHeel = TRUE,  
  validateSchema = FALSE,  
  runCostAnalysis = FALSE,  
  conceptHierarchy = TRUE,  
  createIndices = TRUE,  
  numThreads = 1,  
  tempAchillesPrefix = "tmpach",  
  dropScratchTables = TRUE,  
  sqlOnly = FALSE,  
  outputFolder = "output",  
  logMultiThreadPerformance = FALSE)
```

The scratchDatabaseSchema is where staging tables for each individual analysis is stored

The source name, if left NULL, will be pulled from the cdm\_source table if possible

Variable name updated to OHDSI standards (camel case)

Package will verify CDM version >= 5

Verifies that the CDM schema is valid (contains all tables and fields). Previously used insert statements, now uses a big CTAS for faster execution in MPP systems

The number of concurrent SQL sessions to use to execute the whole analysis. Use > 1 for faster performance on MPP platforms, or just stick with 1. NULL also means 1.

Create a log file of execution times of all SQL scripts

The prefix of staging tables, by default, we use "tmpach"

Should we drop the staging tables at the end, or just keep them? Keeping them could be useful if results generation is time-sensitive

# Executing the new Achilles Heel

```
Achilles::achillesHeel(connectionDetails,  
  cdmDatabaseSchema,  
  resultsDatabaseSchema = cdmDatabaseSchema,  
  scratchDatabaseSchema = resultsDatabaseSchema,  
  cdmVersion = "5",  
  numThreads = 1,  
  tempHeelPrefix = "tmpheel",  
  dropScratchTables = FALSE,  
  ThresholdAgeWarning = 125,  
  ThresholdOutpatientVisitPerc = 0.43,  
  ThresholdMinimalPtMeasDxRx = 20.5,  
  outputFolder = "output",  
  sqlOnly = FALSE)
```

The prefix of staging tables, by default, we use "tmpheel"

Should we drop the staging tables at the end, or just keep them? Keeping them could be useful if results generation is time-sensitive

The scratchDatabaseSchema is where staging tables for each individual analysis is stored

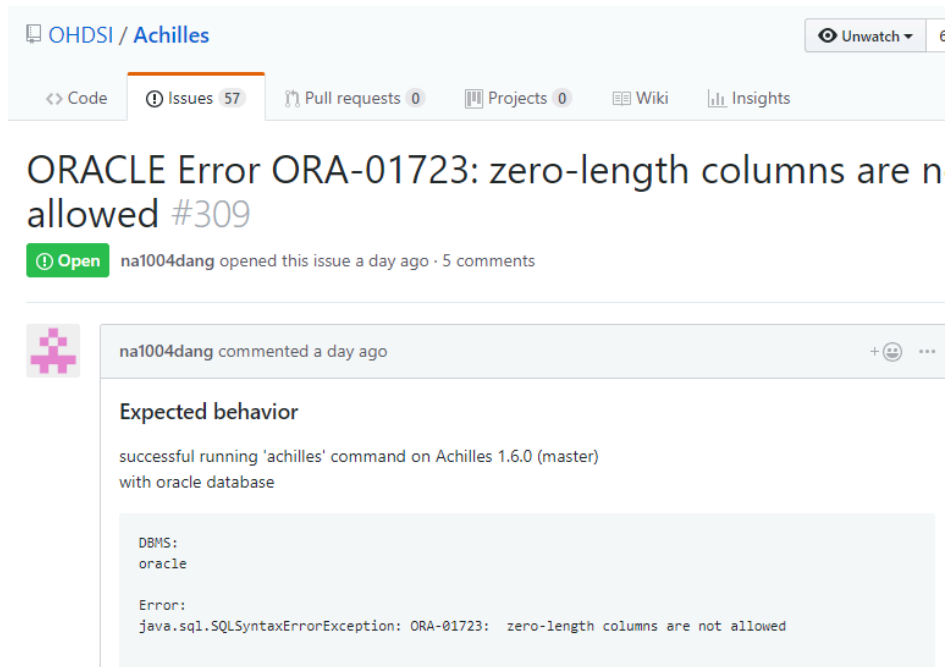
Package will verify CDM version >= 5

The number of concurrent SQL sessions to use to execute the whole analysis. Use > 1 for faster performance on MPP platforms, or just stick with 1. NULL also means 1.

Threshold settings, previously unavailable to users

# Oracle users -- sorry!

Fixing this ASAP, use v1.5 for now.




OHDSI / Achilles Unwatch

[Code](#) [Issues 57](#) [Pull requests 0](#) [Projects 0](#) [Wiki](#) [Insights](#)

## ORACLE Error ORA-01723: zero-length columns are not allowed #309

[Open](#) na1004dang opened this issue a day ago · 5 comments

 na1004dang commented a day ago + 🗨️ ⋮

### Expected behavior

successful running 'achilles' command on Achilles 1.6.0 (master) with oracle database

```
DBMS:
oracle

Error:
java.sql.SQLException: ORA-01723: zero-length columns are not allowed
```