SQL START! 2022

SQL Server 2022 Parameter Sensitive Plan Optimization

Sergio Govoni

Materials: https://bit.ly/3GTMOqk

Speaker bio

Sergio Govoni

Microsoft® MVP Most Valuable Professional

twitter.com/segovoni

github.com/segovoni

linkedin.com/in/sgovoni

Partners

UNIVERSITÀ
POLITECNICA
DELLE MARCHE

Agenda

- Introduction to Plan Caching mechanisms
- Introduction to Parameter Sniffing/Parameter Sensitivity
- SQL Server 2022 Parameter Sensitive Plan (PSP) Optimization

Introduction to plan caching mechanisms

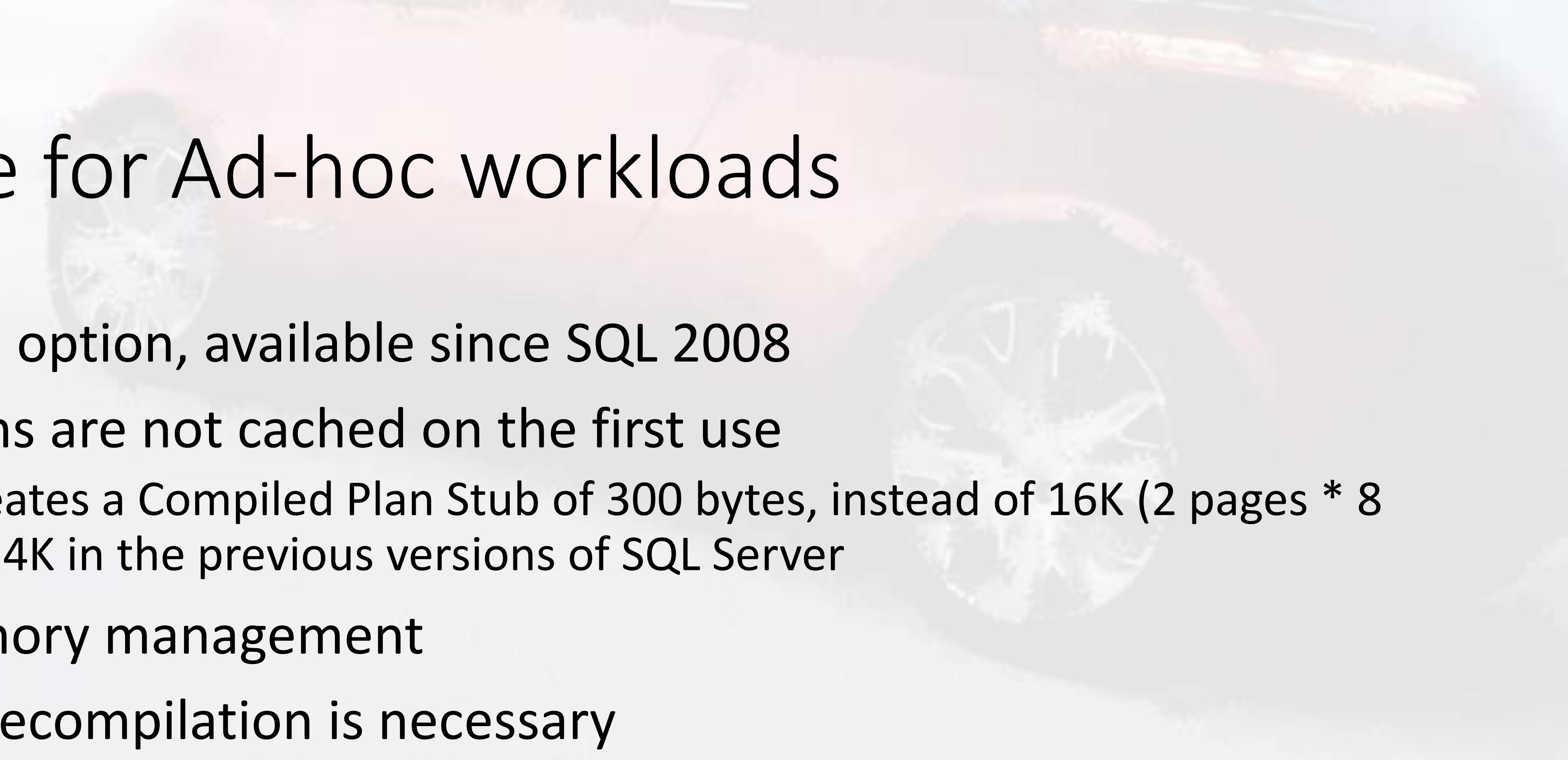
Plan Cache Internals

- sys.dm_exec_cached_plans
- There are six different types of plan that could be cached
    - Compiled Plan, Compiled Plan Stub, Parse Tree, Extended Proc, CLR Proc, CLR Function
- For Compiled Plan category we have three types
    - Adhoc, Prepared, Proc
- usecounts gives you a good indication about how many times a particular plan has been used

Ad-hoc Plans

- Ad-hoc plans are the default operation for SQL Server plan caching mechanism
- In order to reuse an ad-hoc plan
  - You have to submit the **same query text**
  - The **environment** must be **identical**, SET options must be identical!
- Every difference will force a recompilation of the query and another plan will be generated for the same logical query

Optimize for Ad-hoc workloads

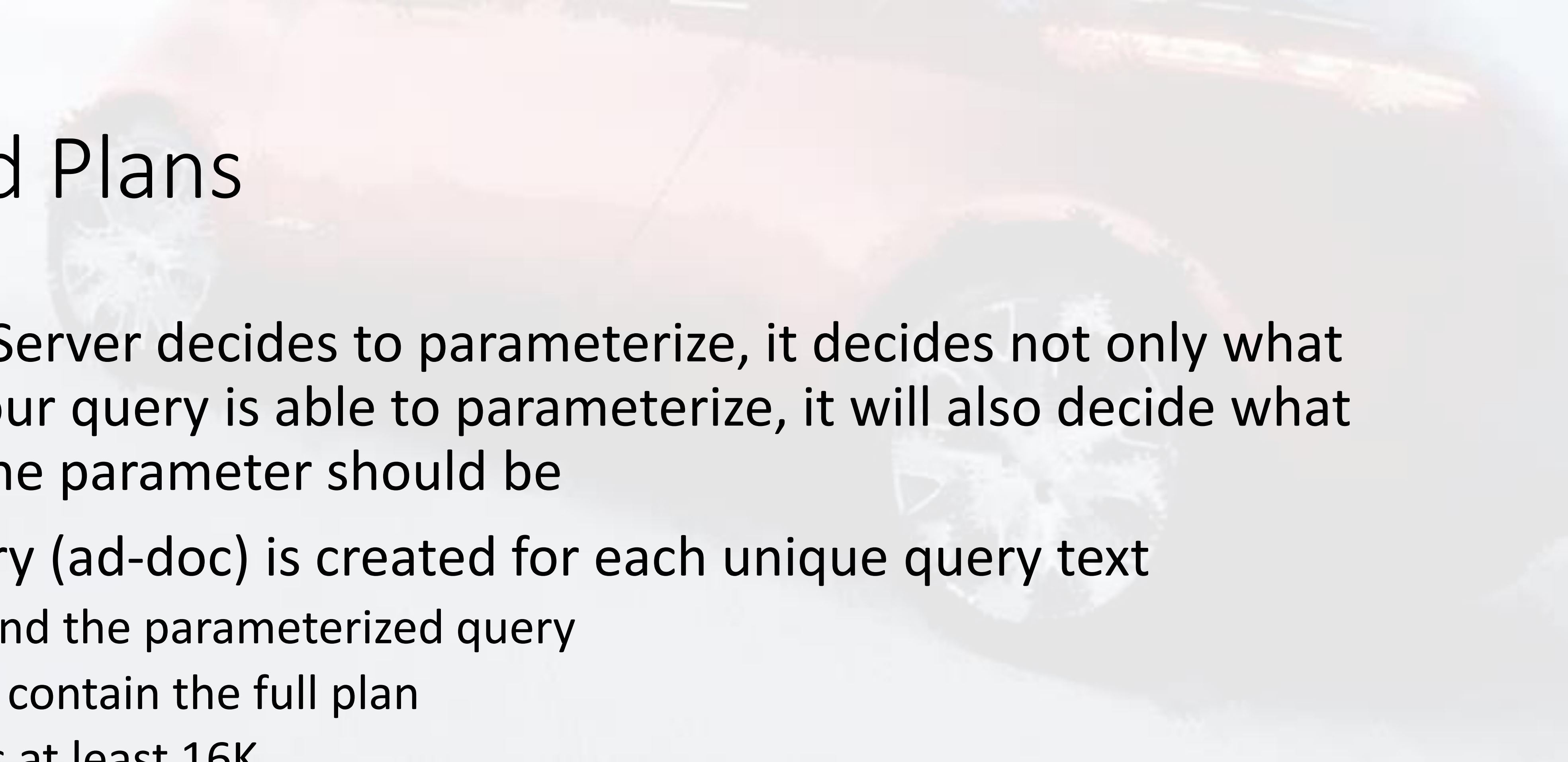


- Server level option, available since SQL 2008
- Ad-hoc plans are not cached on the first use
  - It only creates a Compiled Plan Stub of 300 bytes, instead of 16K (2 pages * 8 Kbytes), 24K in the previous versions of SQL Server
- Better memory management
- One more recompilation is necessary
- sys.dm_exec_cached_plans
  - objtype “Adhoc”, cacheobjtype “Compiled Plan Stub”

Prepared Plans

- When objtype shows “Prepared” means that SQL Server has used parameters in the execution plan
- Auto-parameterization
  - SIMPLE (default) applied to small class of queries
  - FORCED parametrization, database option (since SQL 2005), allows SQL Server to parametrize more queries
- Explicit parameterization
  - Could be done at the application level by “Prepare” method

Prepared Plans



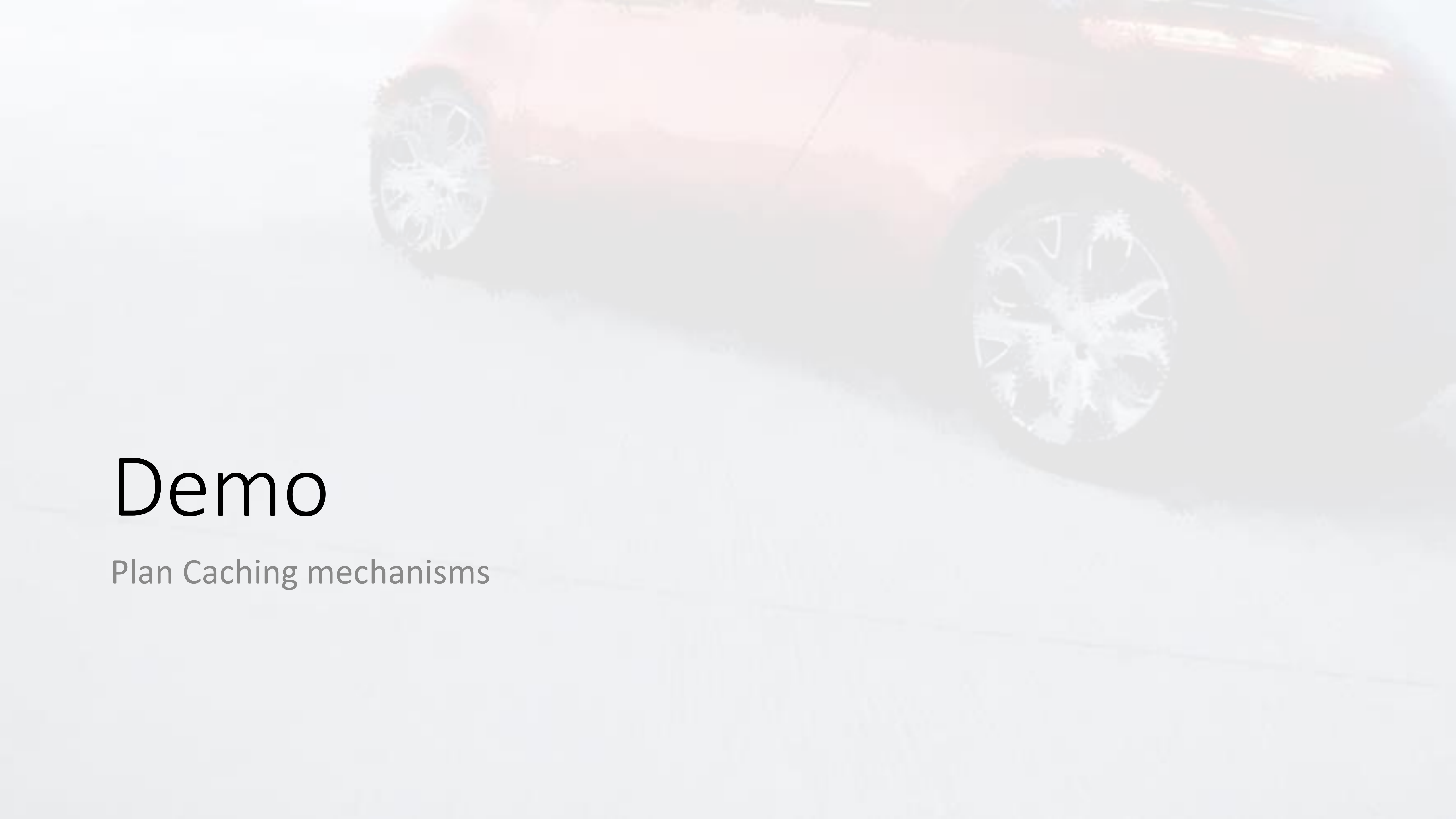
- When SQL Server decides to parameterize, it decides not only what values in your query is able to parameterize, it will also decide what data type the parameter should be
- A shell query (ad-doc) is created for each unique query text
  - Used to find the parameterized query
  - It doesn’t contain the full plan
  - The size is at least 16K

SIMPLE Parametrization

- SQL Server will performs simple parametrization if the plan will be recognized as “Safe”
- Simple parameterization requires simple query!
- A query isn’t simple if it contains
  - JOIN
  - Sub-queries
  - SET clause that contains variables (for UPDATEs)
  - UNION, INTO, DISTINCT, TOP
  - GROUP BY, HAVING or COMPUTE BY
  - ...

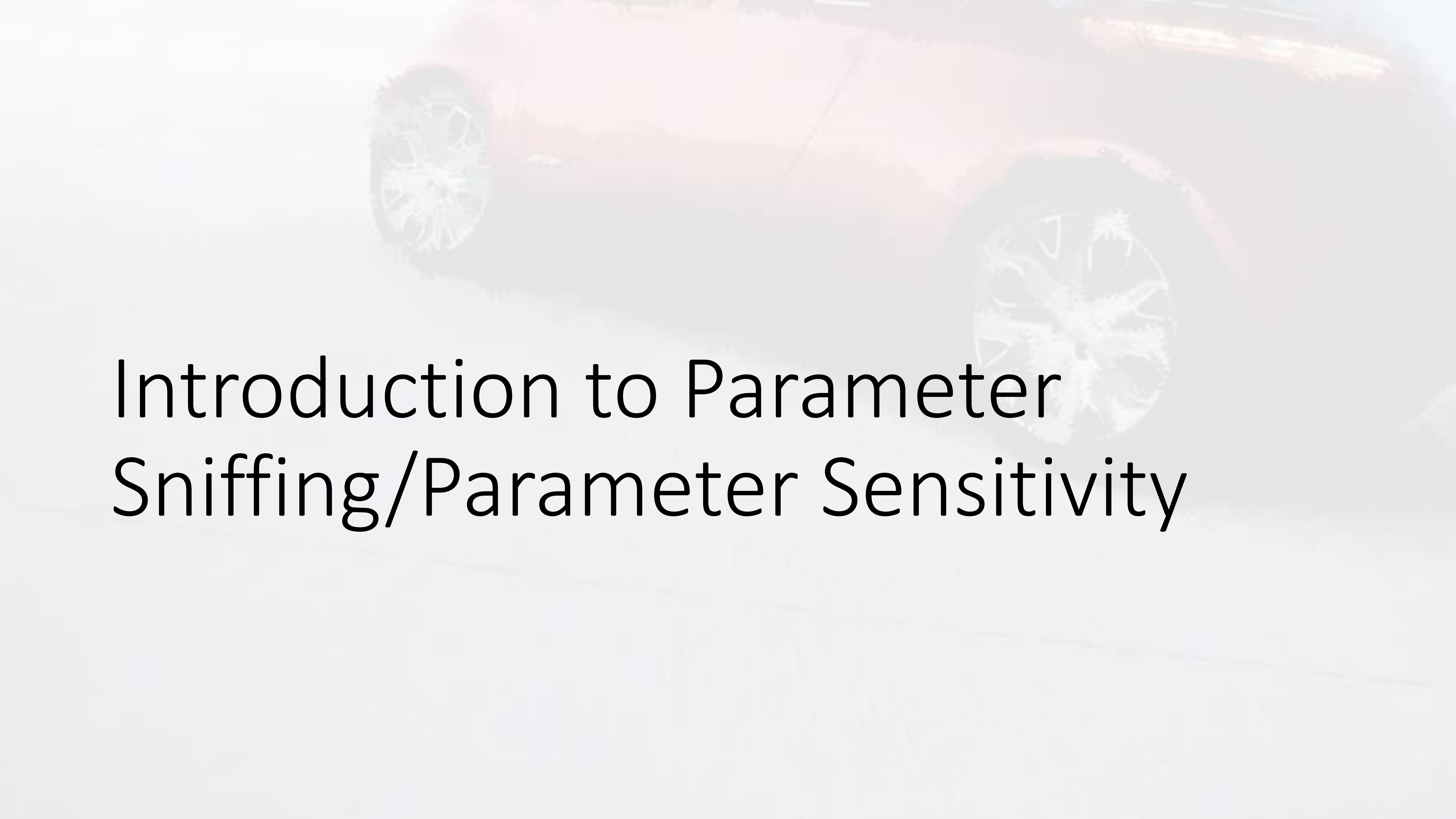
FORCED Parametrization

- When you set the PARAMETERIZATION database option to FORCED, SQL Server will starts to parameterizing almost everything
- sp_get_query_template returns the parameterized form of a query mimic FORCED parametrization
- Exclusions
  - INSERT.. EXEC
  - Statements into SPs, Triggers, UDFs and cursors
  - Statements running with ANSI_PADDING or ANSI_NULLS set to OFF
  - …



Demo

Plan Caching mechanisms



Introduction to Parameter Sniffing/Parameter Sensitivity

Parameter Sensitivity

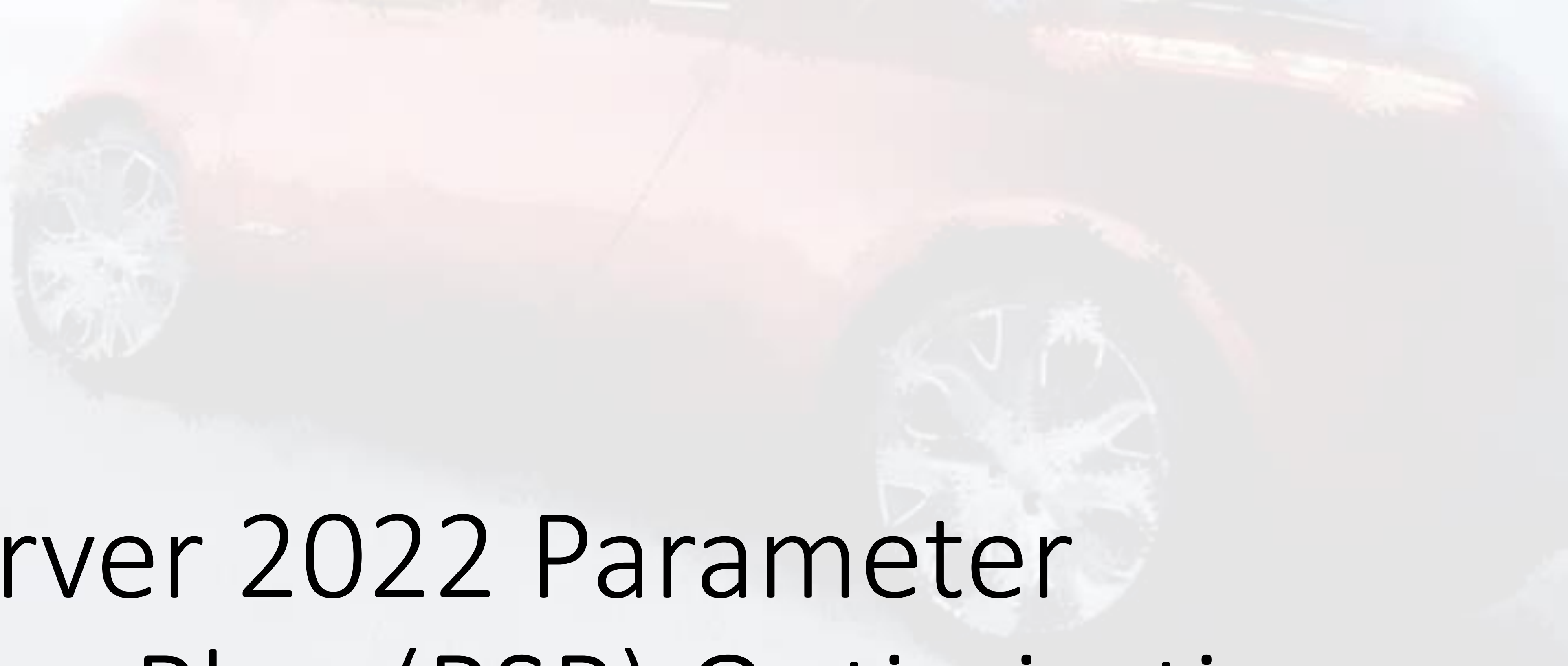
- Parameter sensitivity, also known as Parameter Sniffing, refers to a process whereby SQL Server “sniffs” the current parameter values during compilation or recompilation, and passes it along to the Query Optimizer
- Parameter values are sniffed during compilation or recompilation for the following types of batches
  - Stored procedures
  - Queries submitted via sp_executesql
  - Prepared queries

Parameter Sensitivity

- A parameter sensitive plan (PSP) problem happens when the query optimizer generates a query execution plan that’s optimal only for a specific parameter value (or set of values) and the cached plan is then not optimal for parameter values that are used in consecutive executions
- Plans that aren’t optimal can then cause query performance problems and degrade overall workload throughput

Parameter Sensitivity

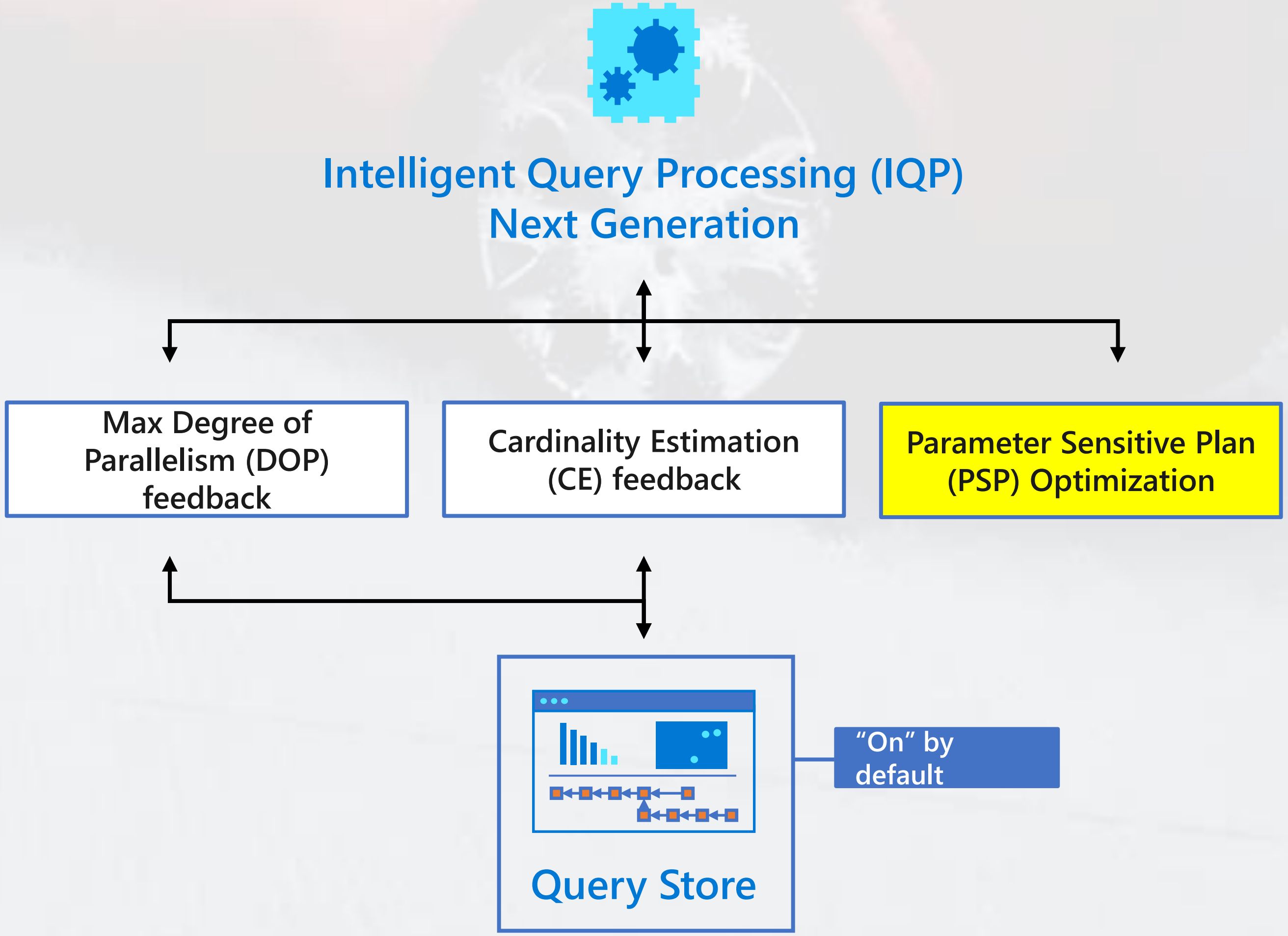
- It isn’t necessary a bad thing
- It’s a bad thing when it’s a bad thing
- It’s a good thing when it’s a good thing ☺, so it depends!
- The problems come when cached plan, with compiled parameters value, is not good enough for current parameters



SQL Server 2022 Parameter Sensitive Plan (PSP) Optimization

Query Store and Intelligent Query Processing

- Accelerate query performance and tuning with no code changes
  - Query Store now turned on by default
  - Query Store support for read replicas from availability groups
  - Query hints to shape plans with no code changes
  - New IQP scenarios enabled through better together capabilities



Intelligent Query Processing (IQP)
Next Generation

Max Degree of Parallelism (DOP) feedback

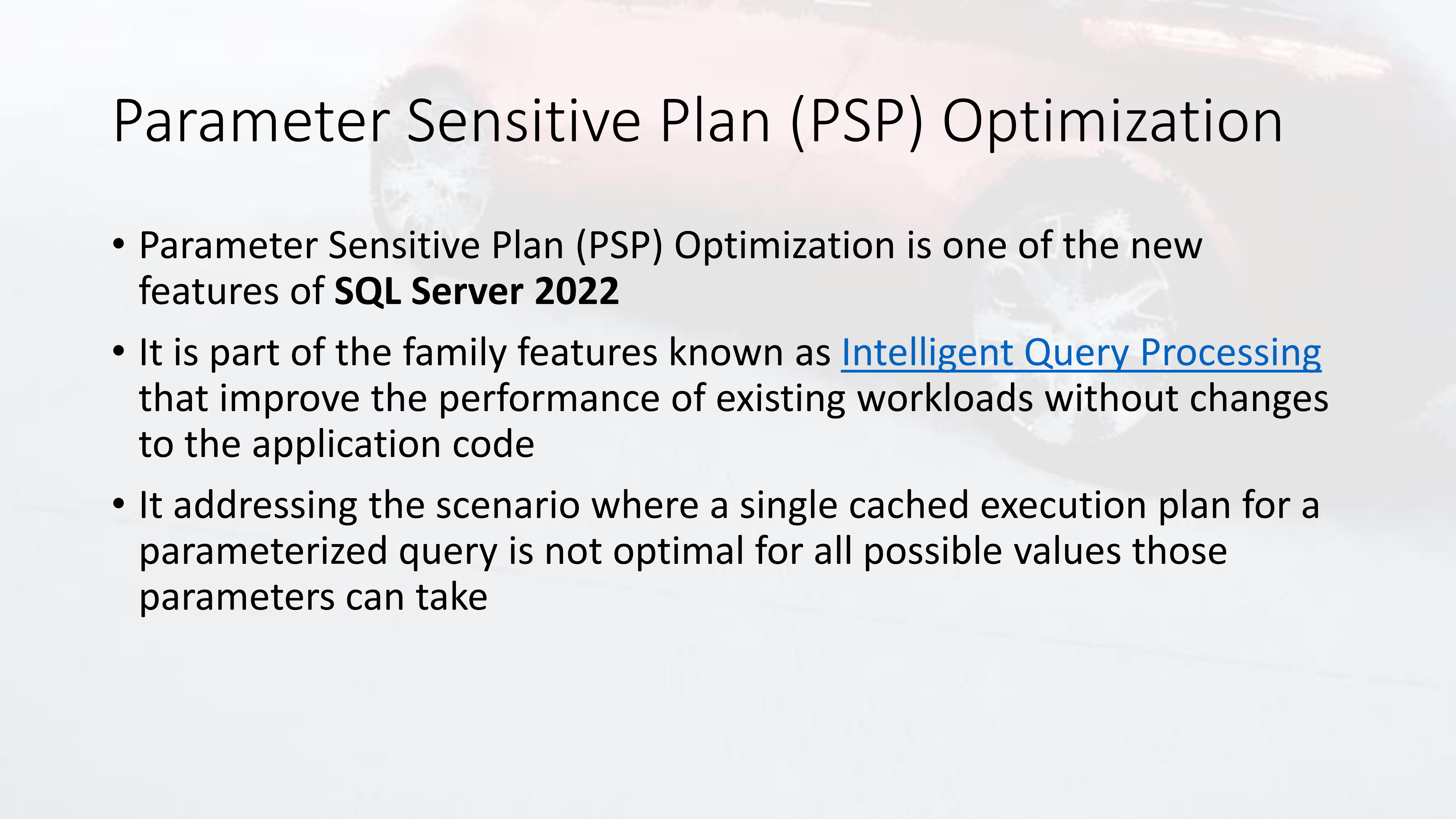
Cardinality Estimation (CE) feedback

Parameter Sensitive Plan (PSP) Optimization

Query Store

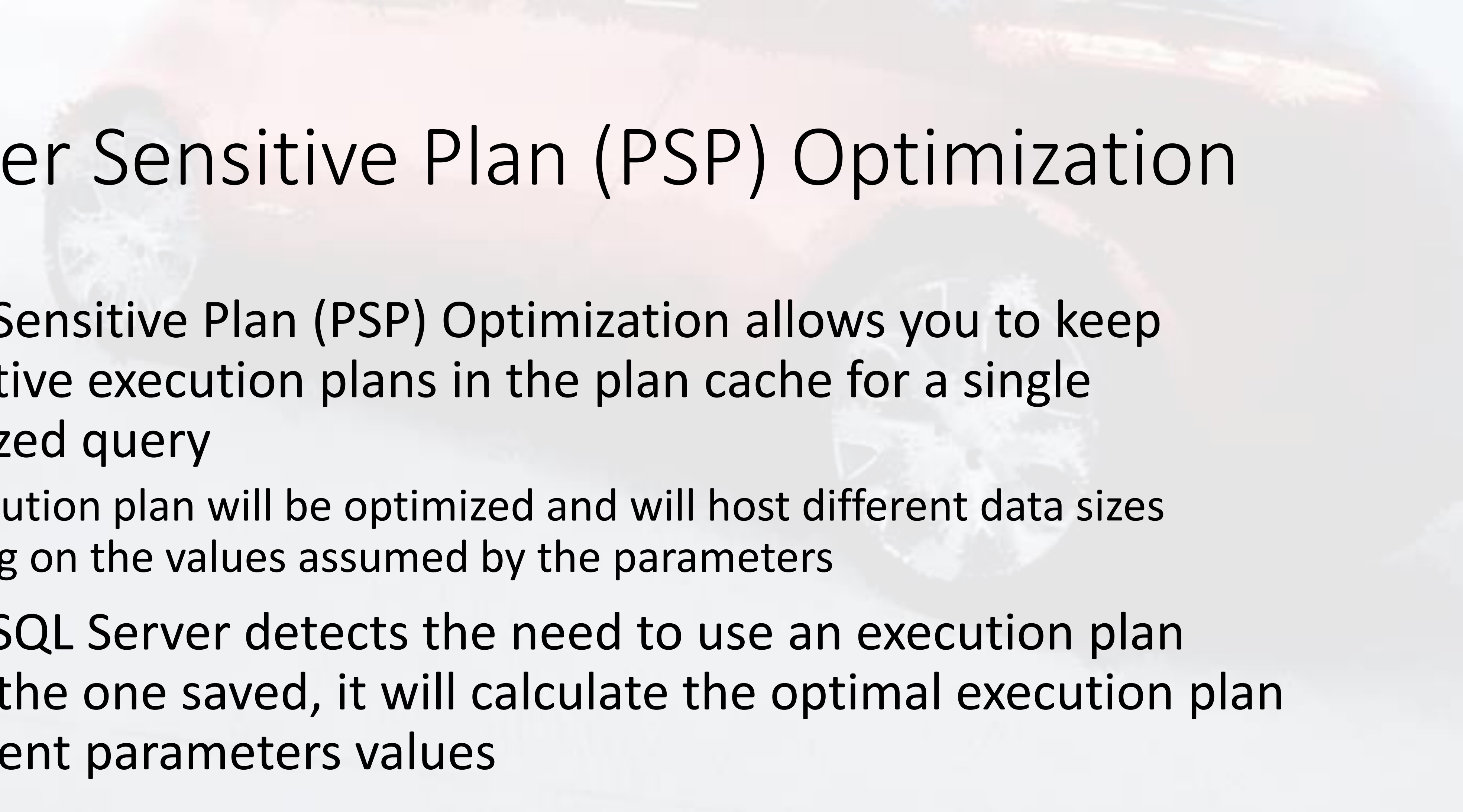
"On" by default

Parameter Sensitive Plan (PSP) Optimization



- Parameter Sensitive Plan (PSP) Optimization is one of the new features of **SQL Server 2022**
- It is part of the family features known as Intelligent Query Processing that improve the performance of existing workloads without changes to the application code
- It addressing the scenario where a single cached execution plan for a parameterized query is not optimal for all possible values those parameters can take

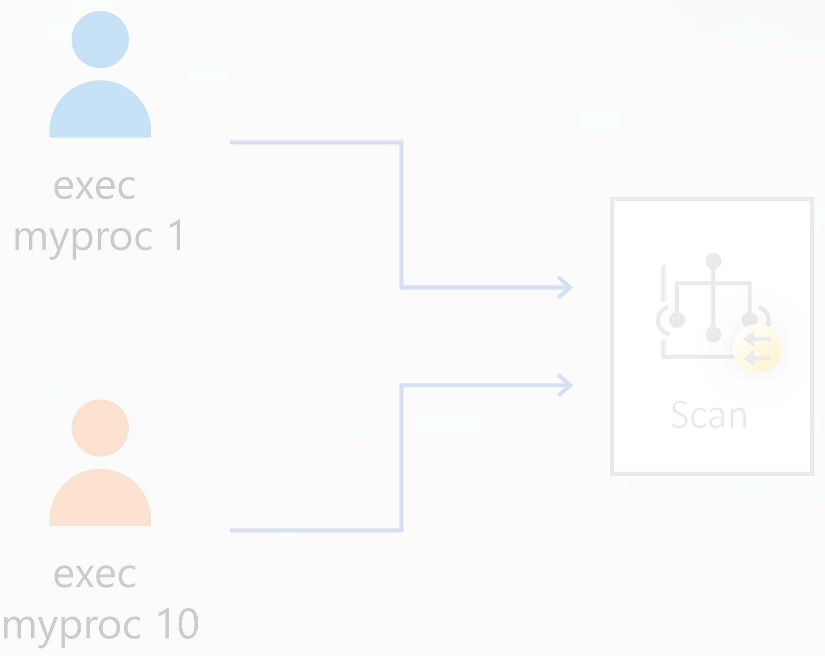
Parameter Sensitive Plan (PSP) Optimization



- Parameter Sensitive Plan (PSP) Optimization allows you to keep multiple active execution plans in the plan cache for a single parameterized query
    - Each execution plan will be optimized and will host different data sizes depending on the values assumed by the parameters
- Whenever SQL Server detects the need to use an execution plan other than the one saved, it will calculate the optimal execution plan for the current parameters values

Parameter Sensitive Plan (PSP) Optimization

Before

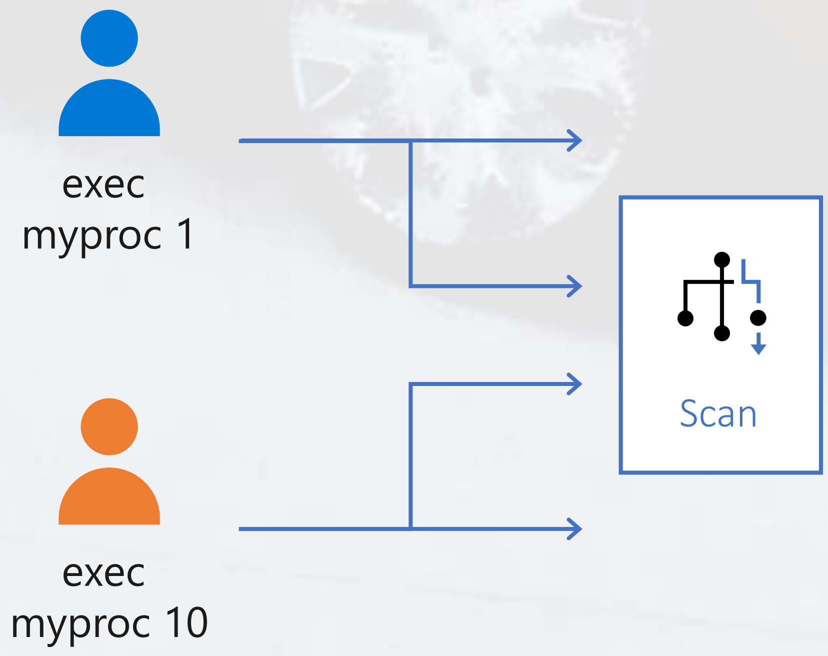


exec myproc 1

exec myproc 10

Scan

With PSP optimization



exec myproc 1

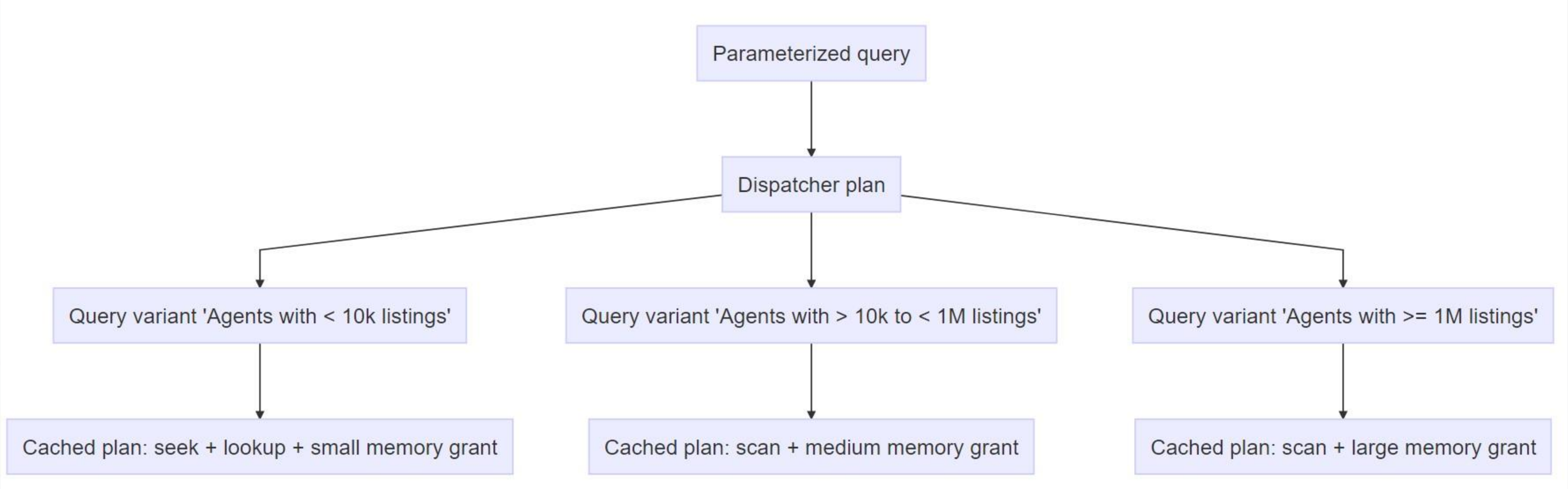
exec myproc 10

Scan

Dispatcher plan

- For execution plans suitable to be optimized with PSP, the initial compilation produces a dispatcher plan that contains the optimization logic
- The dispatcher plan maps to query variants based on the predicates cardinality range boundary values
  - Each variant is linked to an execution plan in which you will find the most suitable operators to handle the dataset that is estimated to be returned by that specific query variant
- Dispatcher plans are automatically updated
- Execution plans linked to query variants are recompiled independently as needed

Dispatcher plan



Parameterized query

Dispatcher plan

Query variant 'Agents with < 10k listings'

Query variant 'Agents with > 10k to < 1M listings'

Query variant 'Agents with >= 1M listings'

Cached plan: seek + lookup + small memory grant

Cached plan: scan + medium memory grant

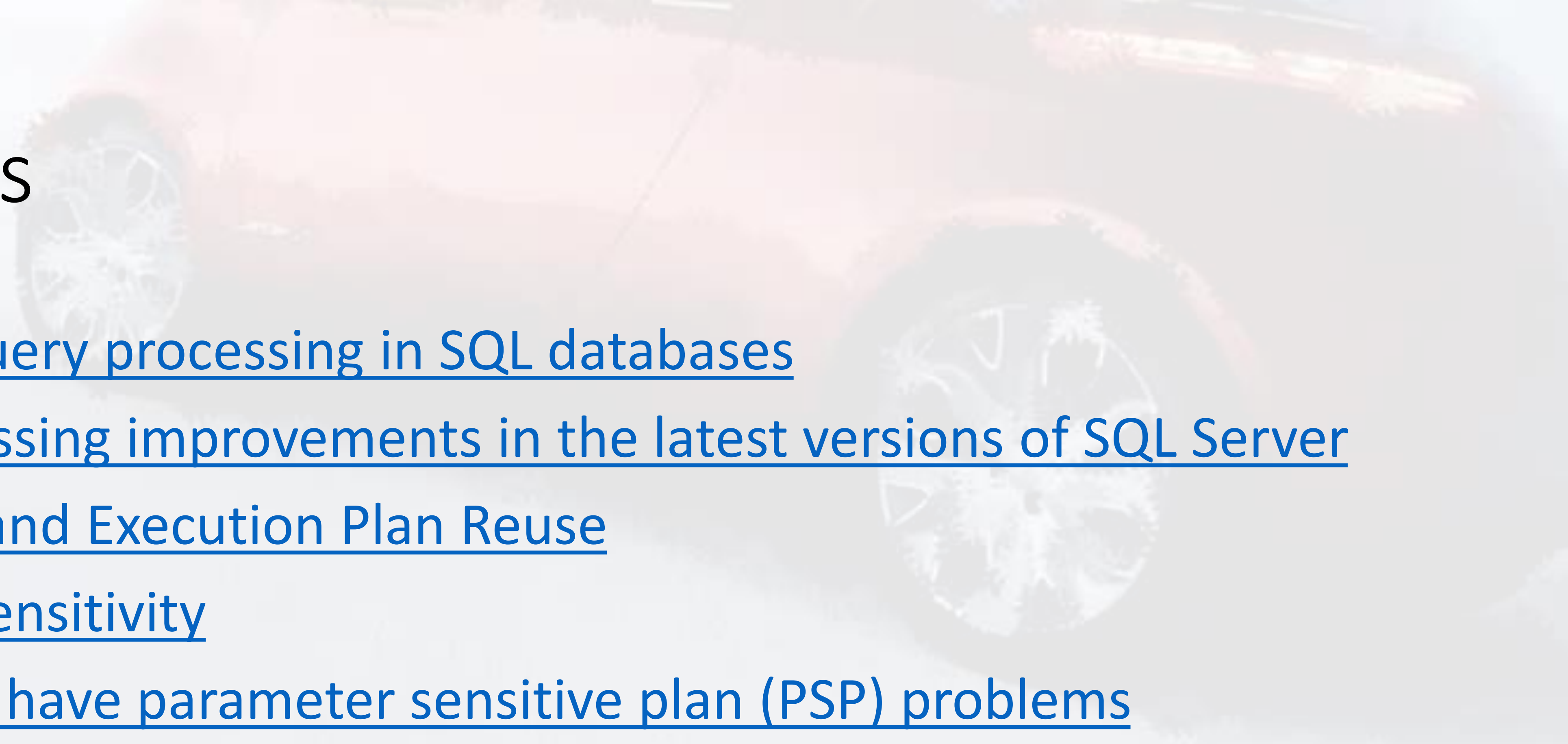
Cached plan: scan + large memory grant

https://docs.microsoft.com/en-us/sql/relational-databases/performance/parameter-sensitivity-plan-optimization

Demo

SQL Server 2022 Parameter Sensitive Plan (PSP) Optimization

Resources



- Intelligent query processing in SQL databases
- Query Processing improvements in the latest versions of SQL Server
- Parameters and Execution Plan Reuse
- Parameter Sensitivity
- Queries that have parameter sensitive plan (PSP) problems
- SQL Server 2022 Parameter Sensitive Plan Optimization
- Session materials

Summary

- Query Store enabled by default in SQL Server 2022 combined with the new generation of Intelligent Query Processing allow to improve performance in some common scenarios, without changes to the T-SQL code
- Parameter Sensitive Plan (PSP) Optimization allows to keep multiple active execution plans in the plan cache for a single parameterized query solving the famous problem known with the name of Parameter Sensitivity or Parameter Sniffing

Thanks

Questions?

github.com/segovoni

twitter.com/segovoni

linkedin.com/in/sgovoni