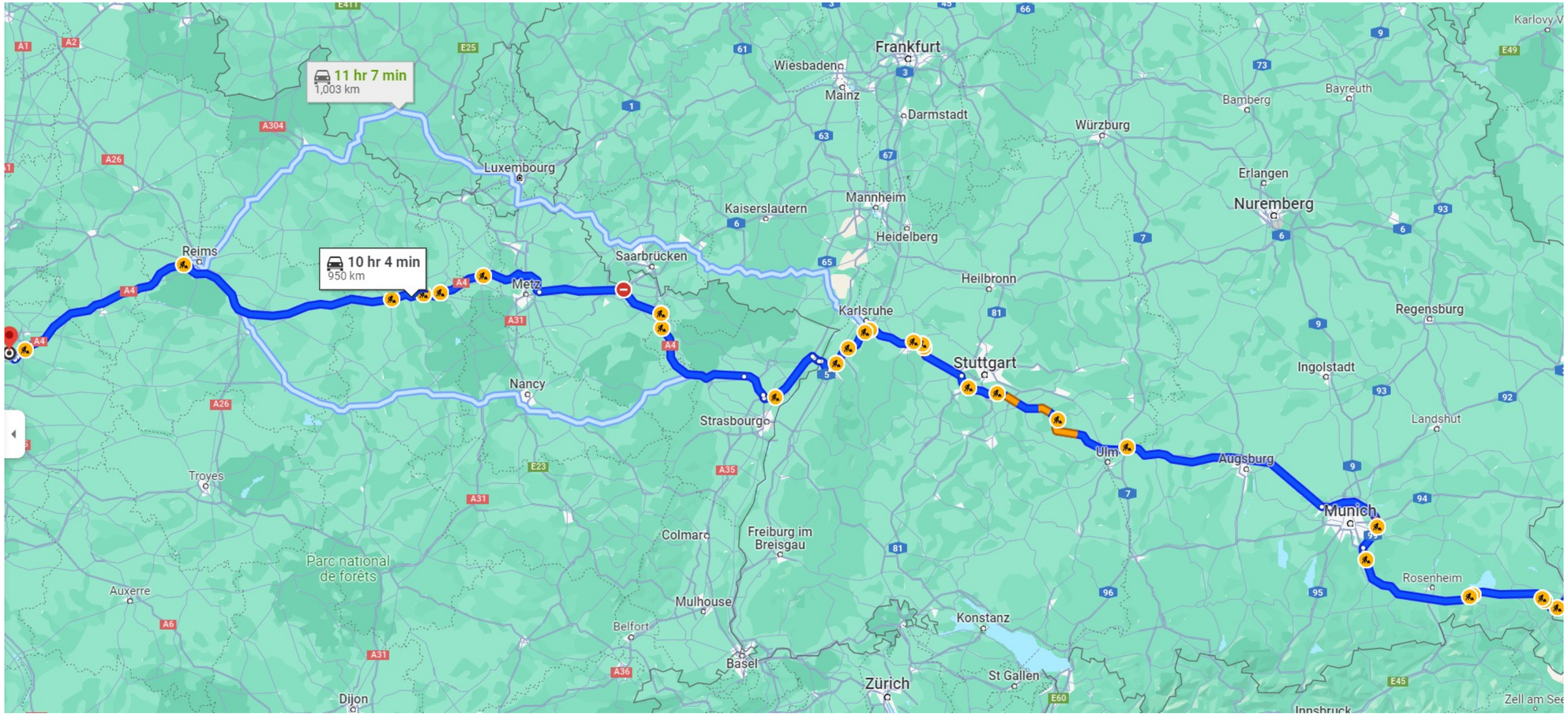
Understanding Plan Reuse and Plan Cache

Nikola Ilic
Data Mozart

@DataMozart | www.data-mozart.com



Salzburg -> Paris

Plan cache is the area where the SQL Server stores execution plans

Overview

Ad-hoc vs prepared workloads

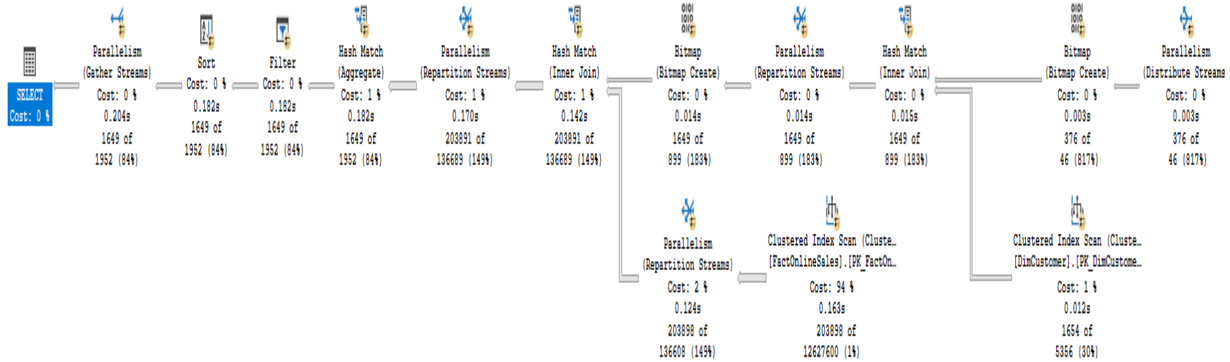
Query plan cache to obtain information about execution plans

How Query Caching Works?

```sql
SELECT cu.FirstName
    , cu.LastName
    , geo.RegionCountryName AS country
    , geo.CityName AS city
    , SUM(fso.SalesAmount) AS salesAmount
FROM dbo.FactOnlineSales AS fso
    INNER JOIN dbo.DimCustomer AS cu ON cu.CustomerKey = fso.CustomerKey
    --INNER JOIN dbo.DimProduct AS pr ON pr.ProductKey = fso.ProductKey
    INNER JOIN dbo.DimGeography AS geo ON geo.GeographyKey = cu.GeographyKey
WHERE cu.Education = 'Bachelors'
    AND geo.RegionCountryName = 'Australia'
GROUP BY cu.CustomerKey
    , cu.FirstName
    , cu.LastName
    , geo.RegionCountryName
    , geo.CityName
HAVING SUM(fso.SalesAmount) > 100
ORDER BY salesAmount DESC
```

Query optimization

Plan Cache



Retrieve only the data you need and only when you need it!

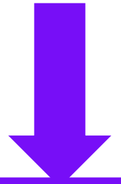
Ad-Hoc Workloads in SQL Server

Executing the query without parameters

```sql
SELECT cu.FirstName
    , cu.LastName
    , geo.RegionCountryName AS country
    , geo.CityName AS city
    , SUM(fso.SalesAmount) AS salesAmount
FROM dbo.FactOnlineSales AS fso
    INNER JOIN dbo.DimCustomer AS cu ON cu.CustomerKey = fso.CustomerKey
    --INNER JOIN dbo.DimProduct AS pr ON pr.ProductKey = fso.ProductKey
    INNER JOIN dbo.DimGeography AS geo ON geo.GeographyKey = cu.GeographyKey
WHERE cu.Education = 'Bachelors'
    AND geo.RegionCountryName = 'Australia'
GROUP BY cu.CustomerKey
    , cu.FirstName
    , cu.LastName
    , geo.RegionCountryName
    , geo.CityName
HAVING SUM(fso.SalesAmount) > 100
ORDER BY salesAmount DESC
```

Execution plan A

```sql
SELECT cu.FirstName
    , cu.LastName
    , geo.RegionCountryName AS country
    , geo.CityName AS city
    , SUM(fso.SalesAmount) AS salesAmount
FROM dbo.FactOnlineSales AS fso
    INNER JOIN dbo.DimCustomer AS cu ON cu.CustomerKey = fso.CustomerKey
    --INNER JOIN dbo.DimProduct AS pr ON pr.ProductKey = fso.ProductKey
    INNER JOIN dbo.DimGeography AS geo ON geo.GeographyKey = cu.GeographyKey
WHERE cu.Education = 'High School'
    AND geo.RegionCountryName = 'Germany'
GROUP BY cu.CustomerKey
    , cu.FirstName
    , cu.LastName
    , geo.RegionCountryName
    , geo.CityName
HAVING SUM(fso.SalesAmount) > 1000
ORDER BY salesAmount DESC
```



Execution plan B

Ad-Hoc Workloads in SQL Server

Executing the query without parameters

```
SELECT cu.FirstName
    , cu.LastName
    , geo.RegionCountryName AS country
    , geo.CityName AS city
    , SUM(fso.SalesAmount) AS salesAmount
FROM dbo.FactOnlineSales AS fso
    INNER JOIN dbo.DimCustomer AS cu ON cu.CustomerKey = fso.CustomerKey
    --INNER JOIN dbo.DimProduct AS pr ON pr.ProductKey = fso.ProductKey
    INNER JOIN dbo.DimGeography AS geo ON geo.GeographyKey = cu.GeographyKey
WHERE cu.Education = 'Bachelors'
    AND geo.RegionCountryName = 'Australia'
GROUP BY cu.CustomerKey
    , cu.FirstName
    , cu.LastName
    , geo.RegionCountryName
    , geo.CityName
HAVING SUM(fso.SalesAmount) > 100
ORDER BY salesAmount DESC
```

```
SELECT cu.FirstName
    , cu.LastName
    , geo.RegionCountryName AS country
    , geo.CityName AS city
    , SUM(fso.SalesAmount) AS salesAmount
FROM dbo.FactOnlineSales AS fso
    INNER JOIN dbo.DimCustomer AS cu ON cu.CustomerKey = fso.CustomerKey
    --INNER JOIN dbo.DimProduct AS pr ON pr.ProductKey = fso.ProductKey
    INNER JOIN dbo.DimGeography AS geo ON geo.GeographyKey = cu.GeographyKey
WHERE cu.Education = 'High School'
    AND geo.RegionCountryName = 'Germany'
GROUP BY cu.CustomerKey
    , cu.FirstName
    , cu.LastName
    , geo.RegionCountryName
    , geo.CityName
HAVING SUM(fso.SalesAmount) > 1000
ORDER BY salesAmount DESC
```

Optimize For Ad Hoc workload

Prepared Workloads in SQL Server

Parameters as a fundamental part

Change the query results without changing the query code

Stored procedures

Encapsulate one or more statements to accept user-defined parameters

sp_executesql

Executes a T-SQL statement which is not defined as a stored procedure

Clearing the Query Cache

DBCC FREEPROCCACHE

All plans on all the databases on the server

BE CAREFUL!!!

ALTER DATABASE SCOPED CONFIGURATION CLEAR PROCEDURE_CACHE

All plans in the scope of an individual database

Execution Plan Caching Best Practices

Use explicit parameters in queries

Leverage stored procedures…

…or sp_executesql as an alternative

Avoid ad-hoc queries

– Almost impossible plan reuse

Enable Optimize For Ad Hoc

Demo

Query plan cache with SSMS

- Understand different information from the cache

Plan cache behavior for ad-hoc vs prepared workloads

Summary

Query optimization is an expensive process

Filter your data!

Ad-hoc queries don’t include parameters

Prepared workloads rely on parameters

- Stored procedures
- sp_executesql

Be careful when clearing the query cache!

Up Next:

Understanding Indexing and Statistics