SQLintersection

Session: Wednesday, 9:45am – 11:00am

Built for Speed: SQL Server Database Application Design for Performance

Pam Lahoud

Pedro Lopes

SQL intersection

Speakers: Pam Lahoud & Pedro Lopes

Microsoft

@SQLGoddess

@SQLPedro

SQL intersection

Reminder: Intersect with Speakers and Attendees

- **Tweet *tips and tricks* that you learn and follow tweets posted by your peers!**
  - □ Follow: #SQLintersection and/or #DEVintersection
- **Join us – Wednesday Evening – for SQLafterDark**
  - □ Doors open at 7:00 pm
  - □ Trivia game starts at 7:30 pm

    Winning team receives something fun!
  - □ Raffle at the end of the night

    Lots of great items to win including a seat in a five-day SQLskills Immersion Event!
  - □ The first round of drinks is sponsored by SentryOne and SQLskills

Performance tuning – it's not just for DBAs!

What you can do as a developer to help generate efficient SQL Server code

- **Application Design Patterns**
  - To ORM or not to ORM
  - Are you cloud-ready?
    - Technical debt
    - DB Compatibility Certification
- **Writing Efficient T-SQL**
  - Cardinality
  - SARGability
  - Common T-SQL Anti-Patterns

Application Design Patterns

SQL intersection

To ORM or not to ORM

Developer

Manageability	Agility	Expertise
Application logic belongs in the client layer	Want to leverage “code first” design	Don’t have a dedicated database developer or database expertise

Database Administrator

Performance	Performance	Performance
Some application logic in the database improves performance	ORMs generate spaghetti T-SQL code	No ability to make performance-improving T-SQL code changes

https://aka.ms/EFPerf

To ORM or not to ORM

Developer

Manageability

Application logic belongs in the client layer

Wa… leverage “code first” design

database developer or database expertise

Database Administrator

Performance

logic in the database improves performance

generate spaghetti T-SQL code

No ability to make performance-improving T-SQL code changes

Code first – tune later!

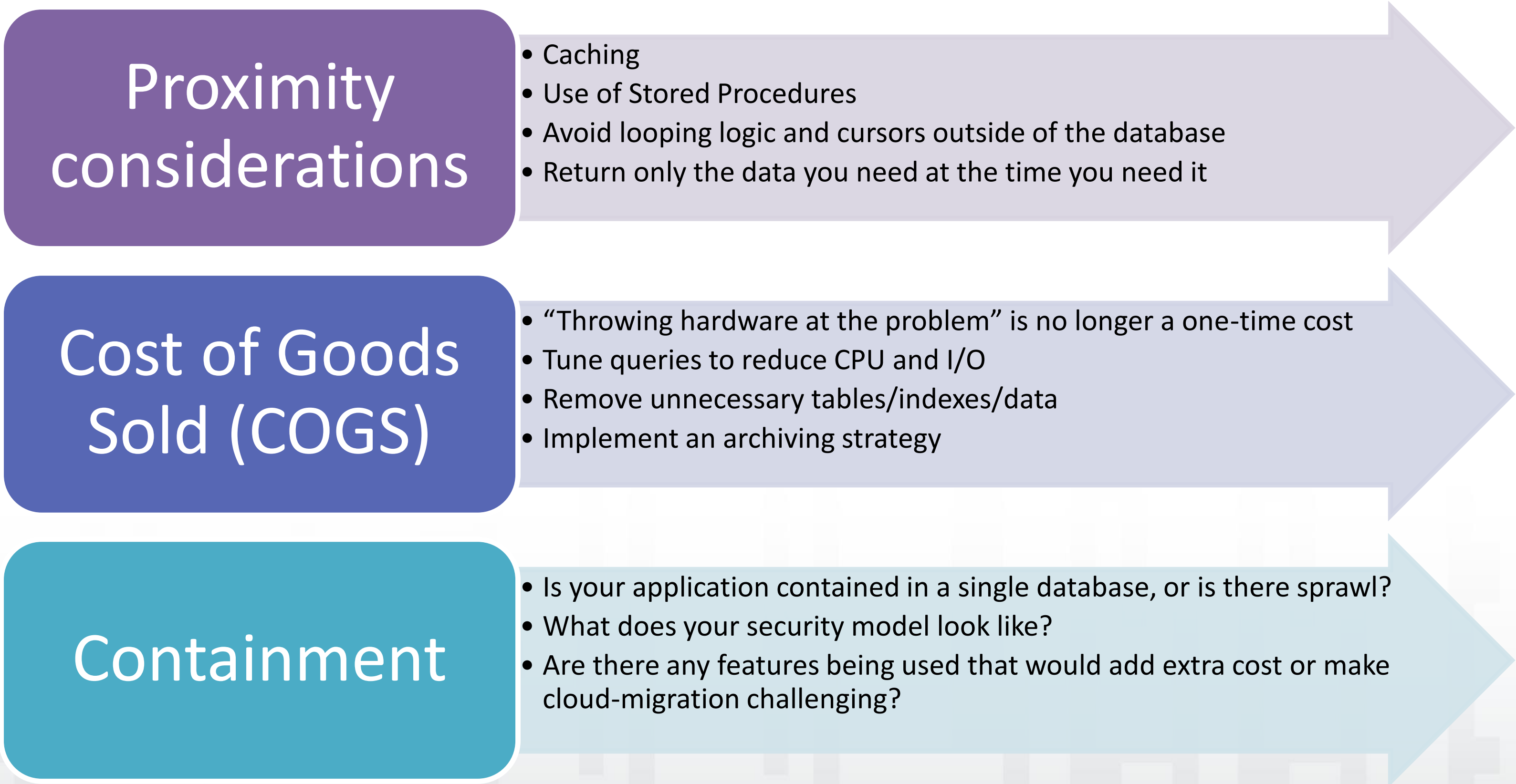
https://aka.ms/EFPerf

Are you cloud-ready?

What can you do today to make moving to the cloud tomorrow easier?

Pay down technical debt

Or don’t accrue it in the first place



Proximity considerations

- Caching
- Use of Stored Procedures
- Avoid looping logic and cursors outside of the database
- Return only the data you need at the time you need it

Cost of Goods Sold (COGS)

- “Throwing hardware at the problem” is no longer a one-time cost
- Tune queries to reduce CPU and I/O
- Remove unnecessary tables/indexes/data
- Implement an archiving strategy

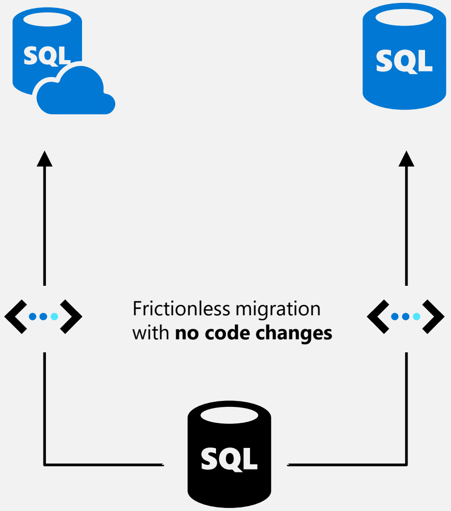
Containment

- Is your application contained in a single database, or is there sprawl?
- What does your security model look like?
- Are there any features being used that would add extra cost or make cloud-migration challenging?

Certify once, run on-premises and in the cloud with Compatibility Certification

Upgrade & modernize your SQL Server database on-premises, in the cloud and on the edge with Compatibility Certification that eliminates risks of application compatibility

Upgrade to the latest SQL Server Database Engine without changing your critical applications



Compatibility Certification benefits

Unified application certification

Applications tested and certified on a given SQL Server version are also implicitly tested and certified on that SQL Server version native database compatibility level

Reduce upgrade risks

Separate application and platform layer upgrade cycles for less disruption

Microsoft fully supports Compatibility Certification

Upgrade to latest SQL Database Engine version

Upgrade your SQL Server Database Engine or move instances to the cloud with no code changes

Database Compatibility Level protection with Microsoft

Microsoft provides an ecosystem of tools and services to test whether Compatibility Certification is right for you and protect you as you upgrade

Maintain backwards compatibility

Applications running on a newer SQL Server Database Engine while using an older Database Compatibility Level can still leverage server-level enhancements without application changes

Database Compatibility Level settings affect behaviors for a specified database, not the entire server

Predictable performance

Microsoft gates query optimization changes and improvements behind Database Compatibility Level to upgrade without issues once validation testing is successfully completed

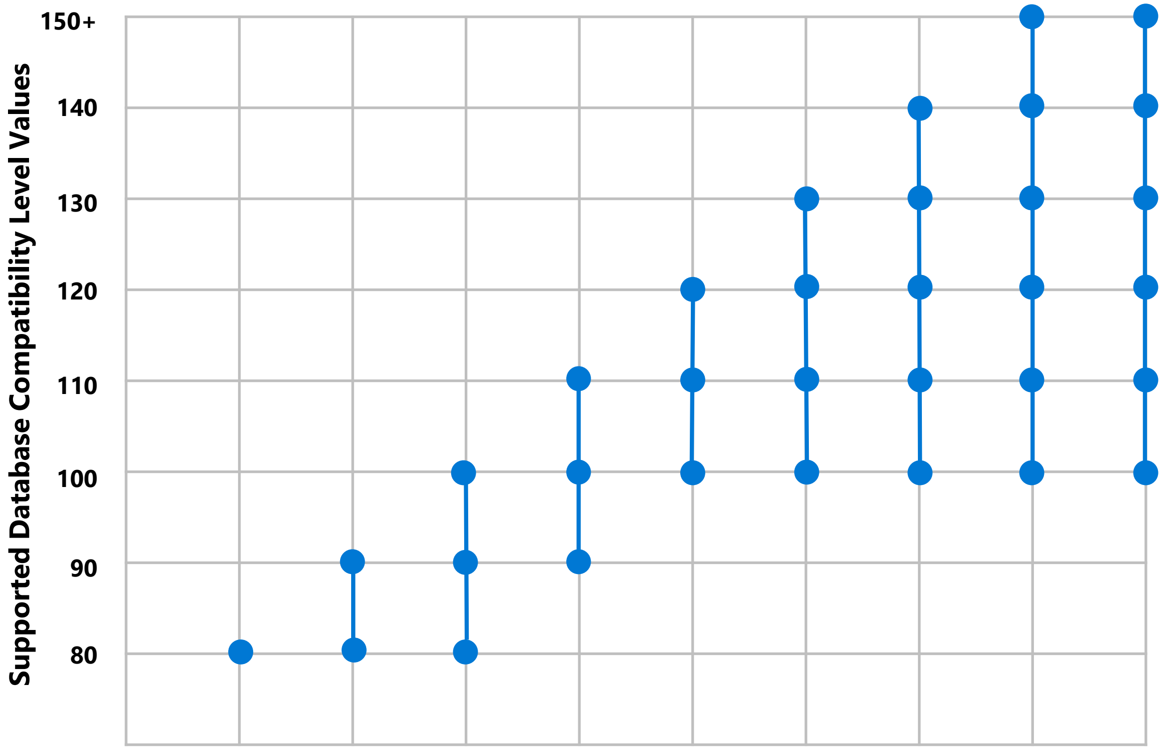
Validation testing tools

Use Data Migration Assistant (DMA) to validate your readiness to upgrade

The DMA tool validation results help protect applications from any functional regressions on target versions

Learn more here: http://aka.ms/dbcompat

Explore your Database Compatibility Level supported values



Product	SQL Server 2000	SQL Server 2005	SQL Server 2008 R2	SQL Server 2012	SQL Server 2014	SQL Server 2016	SQL Server 2017	SQL Server 2019	Azure SQL Database
Default Compatibility Level Designation per product	**80**	**90**	**100**	**110**	**120**	**130**	**140**	**150**	**140**

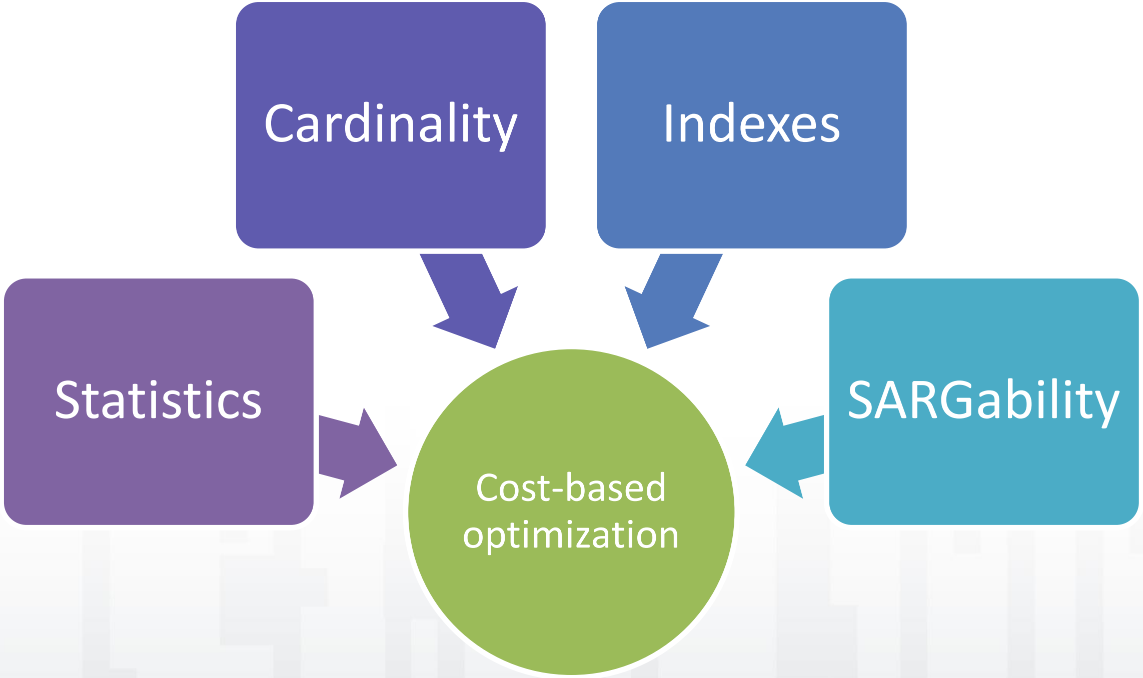
Upgrade from any earlier version of SQL Server and the database retains its existing compatibility level if it is at least minimum allowed for that instance of SQL Server

For example, SQL Server 2008 databases have supported compatibility up to SQL Server 2019 and Azure SQL Database

Writing Efficient T-SQL

SQL intersection

Why did SQL Server pick this plan?



Cardinality

Indexes

Statistics

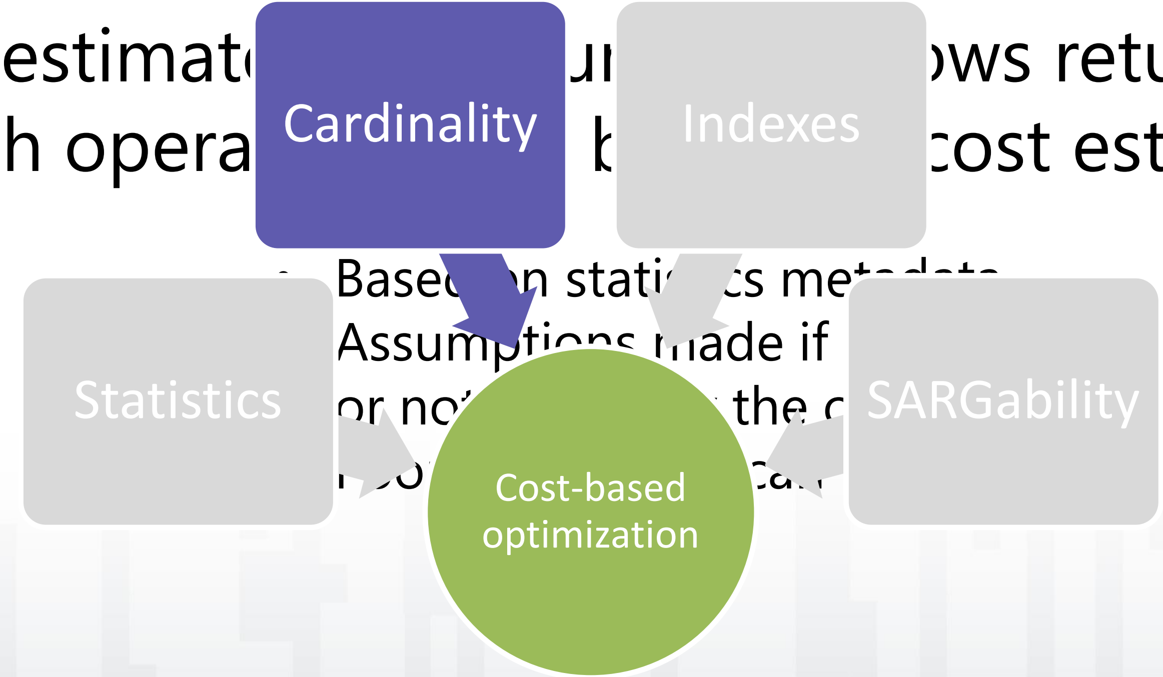
SARGability

Cost-based optimization

What is Cardinality?

An estimat ur ows returned by each opera b cost estimate.

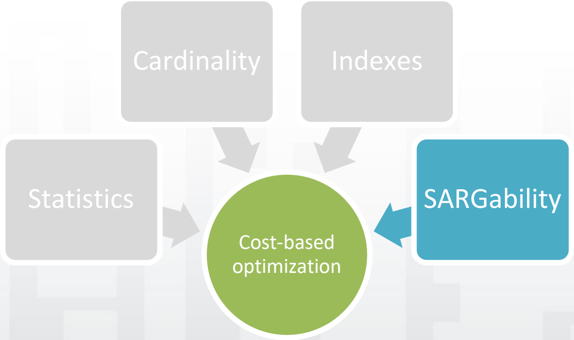
- Base n stati cs met
- Assumptions made if not available or no the c
- Poo ca cient plans



What is SARGability?

The extent to which a predicate can be used as a Search ARGument for an index seek

- Non-SARGable expressions cannot seek, they must scan a table or an index
- Non-SARGable expressions can significantly slow down queries



Common T-SQL Anti-patterns

Certain types of expressions can limit SQL Server’s ability to correctly estimate cardinality and/or use an index to evaluate a predicate

- Implicit Conversions
- Functions in the Predicate
- LIKE with a Leading Wildcard
- OR in the WHERE Clause
- Composable Logic
- Table-valued Functions

Implicit Conversions

ProductID is defined as VARCHAR(8)

```
SELECT *
FROM Product
WHERE ProductID = 7;
```

Rules of data type precedence

Functions in the Predicate

```
SELECT *
FROM Person.Person
WHERE SUBSTRING(FirstName, 1, 1) = 'B';

SELECT *
FROM Sales.SalesOrderHeader
WHERE YEAR(OrderDate) = 2008;
```

Functions in the Predicate

```
SELECT *
FROM Person.Person
WHERE FirstName LIKE 'B%';

SELECT *
FROM Sales.SalesOrderHeader
WHERE OrderDate BETWEEN '1/1/2008'
AND '12/31/2008';
```

Rewritten as SARGable expressions

LIKE with a leading wildcard

Non-SARGable

```
SELECT *
FROM Person.Person
WHERE FirstName LIKE '%B%';
```

SARGable

```
SELECT *
FROM Person.Person
WHERE FirstName LIKE 'B%';
```

No leading wildcard = range scan

OR in the WHERE clause

```
SELECT CustomerID, OrderDate,
       ShipDate, [Status]
FROM Sales.SalesOrderHeader
WHERE SalesPersonID = 277
      OR CustomerID = 29523;
```

OR in the WHERE clause

```
SELECT CustomerID, OrderDate,
       ShipDate, [Status]
FROM Sales.SalesOrderHeader
WHERE SalesPersonID = 277
UNION
SELECT CustomerID, OrderDate,
       ShipDate, [Status]
FROM Sales.SalesOrderHeader
WHERE CustomerID = 29523;
```

Try a UNION instead

Composable Logic – The All-Purpose Query

```
CREATE PROCEDURE usp_GetSalesPersonOrders @SalesPerson
INT NULL AS
SELECT SalesOrderID,
       p.FirstName AS SalesFirstName,
       p.LastName AS SalesLastName
FROM Sales.SalesOrderHeader AS soh
LEFT JOIN Person.Person AS p
     ON soh.SalesPersonID = p.BusinessEntityID
WHERE @SalesPerson IS NULL
OR SalesPersonID = @SalesPerson;
```

Composable Logic – The All-Purpose Query

```
CREATE PROCEDURE usp_GetSalesPersonOrders @SalesPerson INT
NULL AS
IF @SalesPerson IS NULL
   SELECT SalesOrderID,
          p.FirstName AS SalesFirstName,
          p.LastName AS SalesLastName
   FROM Sales.SalesOrderHeader AS soh
   LEFT JOIN Person.Person AS p
        ON soh.SalesPersonID = p.BusinessEntityID;
ELSE …
```

Move conditional logic outside the query

Composable Logic – The All-Purpose Query

```
DECLARE @sql nvarchar(max);
SET @sql = 'SELECT SalesOrderID, p.FirstName AS
SalesFirstName, p.LastName AS SalesLastName
FROM Sales.SalesOrderHeader AS soh
LEFT JOIN Person.Person AS p
     ON soh.SalesPersonID = p.BusinessEntityID';
IF @SalesPerson IS NOT NULL
     SET @sql = @sql + 'WHERE SalesPersonID = @p1';
EXEC sp_executesql @stmt = @sql, @params = N'@p1 INT', @p1 =
@SalesPerson;
```

Or use dynamic SQL

Table-valued Functions

Works like a parameterized view

```
SELECT EmployeeID,
       FirstName,
       LastName,
       JobTitle,
       RecursionLevel
FROM dbo.ufn_FindReports(25);
```

Table-valued Functions

Can be a multi-statement TVF (MSTVF)

```
CREATE FUNCTION dbo.ufn_FindReports (@InEmpID INT)
RETURNS @retFindReports TABLE   (
    EmployeeID int primary key NOT NULL,
    FirstName nvarchar(255) NOT NULL,
    LastName nvarchar(255) NOT NULL,
    JobTitle nvarchar(50) NOT NULL,
    RecursionLevel int NOT NULL  ) AS
BEGIN [multiple statements]
```

Table-valued Functions

Can be an inline TVF

```
CREATE FUNCTION
dbo.ufn_FindReports (@InEmpID int)
RETURNS TABLE AS
RETURN
[single query]
```

Always use Inline TVFs if possible

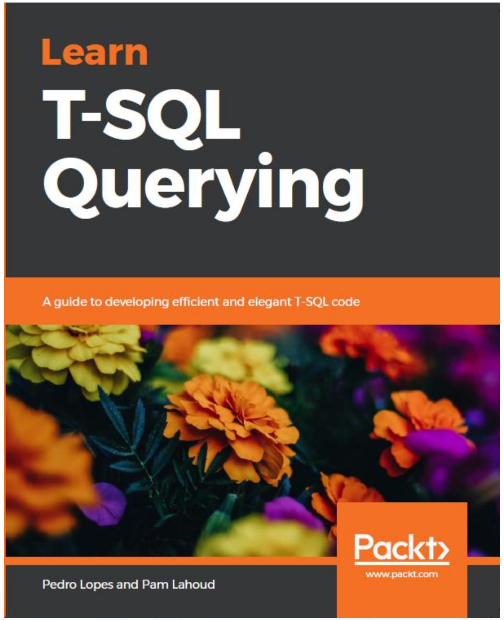
But SQL Server 2019 and Azure SQL will materialize MSTVF automatically

Demo

Detecting Anti-Patterns in a Query Plan

Learn more

Download and try SQL Server 2019	https://aka.ms/ss19 https://aka.ms/SQL2019WhatsNew
Check out these great data-related demos	https://aka.ms/DataDemos https://aka.ms/IQPDemos https://aka.ms/SQL2019Notebooks
Continue learning with our new book	https://aka.ms/LearnTSQLQuerying https://aka.ms/LearnTSQLQuerying_errata
One shortcut to rule them all!	https://aka.ms/SQLShortcuts
Use our free training	https://aka.ms/sqlworkshops


Learn
T-SQL
Querying
A guide to developing efficient and elegant T-SQL code
Packt>
www.packt.com
Pedro Lopes and Pam Lahoud

Thank you!

@SQLGoddess

@SQLPedro

Questions?

Don’t forget to complete an online evaluation!

Built for Speed: SQL Server Database Application Design for Performance

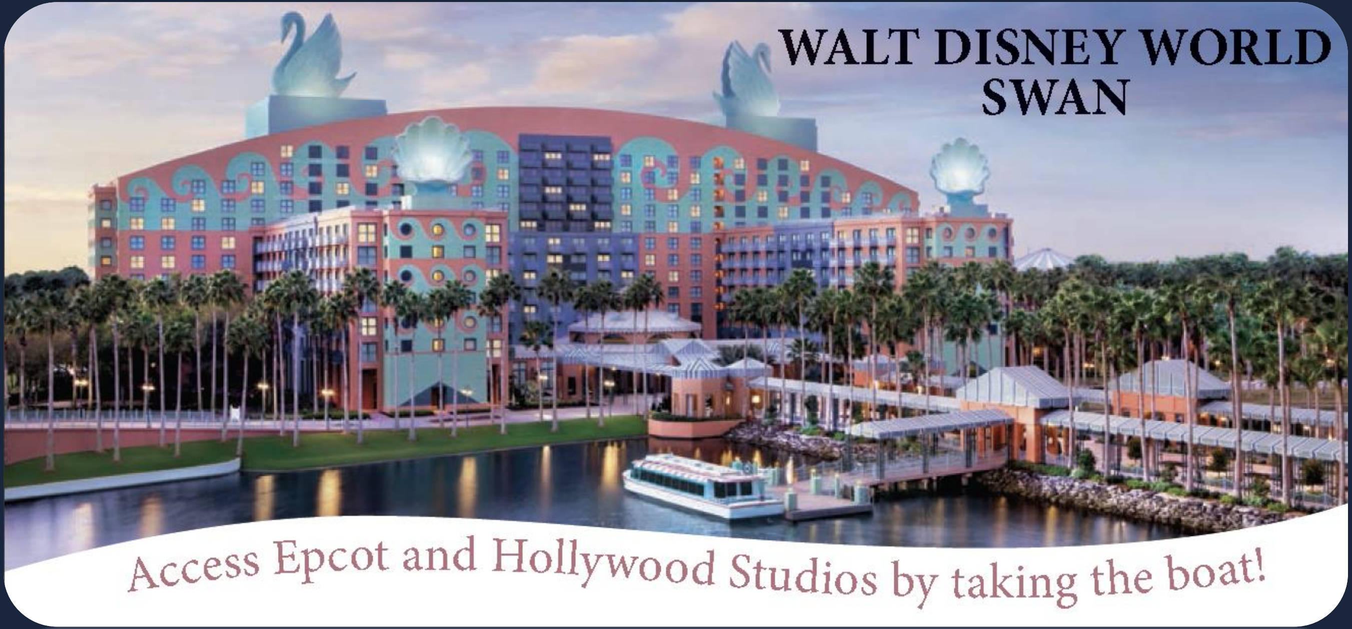
Your evaluation helps organizers build better conferences and helps speakers improve their sessions.

SQL *intersection*

Thank you!

Save the Date!

www.SQLintersection.com



2020

Week of April 6

We're back in Orlando!

SQL intersection

Leave the every day behind and enter a world of wonder and enchantment at the Walt Disney World® Resort. Located in the heart of the most magical place on earth, the Walt Disney World Swan and Dolphin Resort provides a truly extraordinary backdrop for our event! Beautiful tropical landscaping, tranquil waterways, and classic art and architecture work together to create a stunning landmark!