MILOŠSQL

5

HÄUFIGE FEHLER BEIM DATENBANKDESIGN

SQL KONFERENZ, HANAU, 13.09.2023



MILOŠ RADIVOJEVIĆ

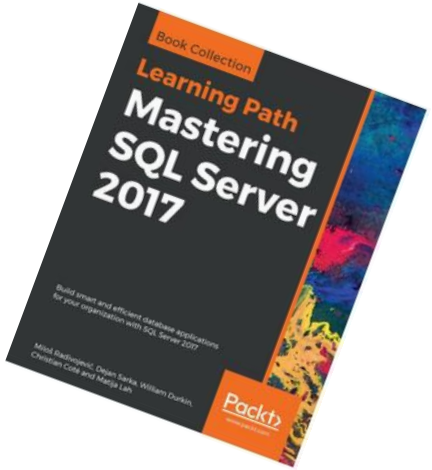


Entain

- Head of MSSQL Database Engineering at Entain, Austria
- Conference speaker
- Book author
- Contact: **milos.radivojevic@chello.at**
- LinkedIn: **milossql**







1

ZEIT



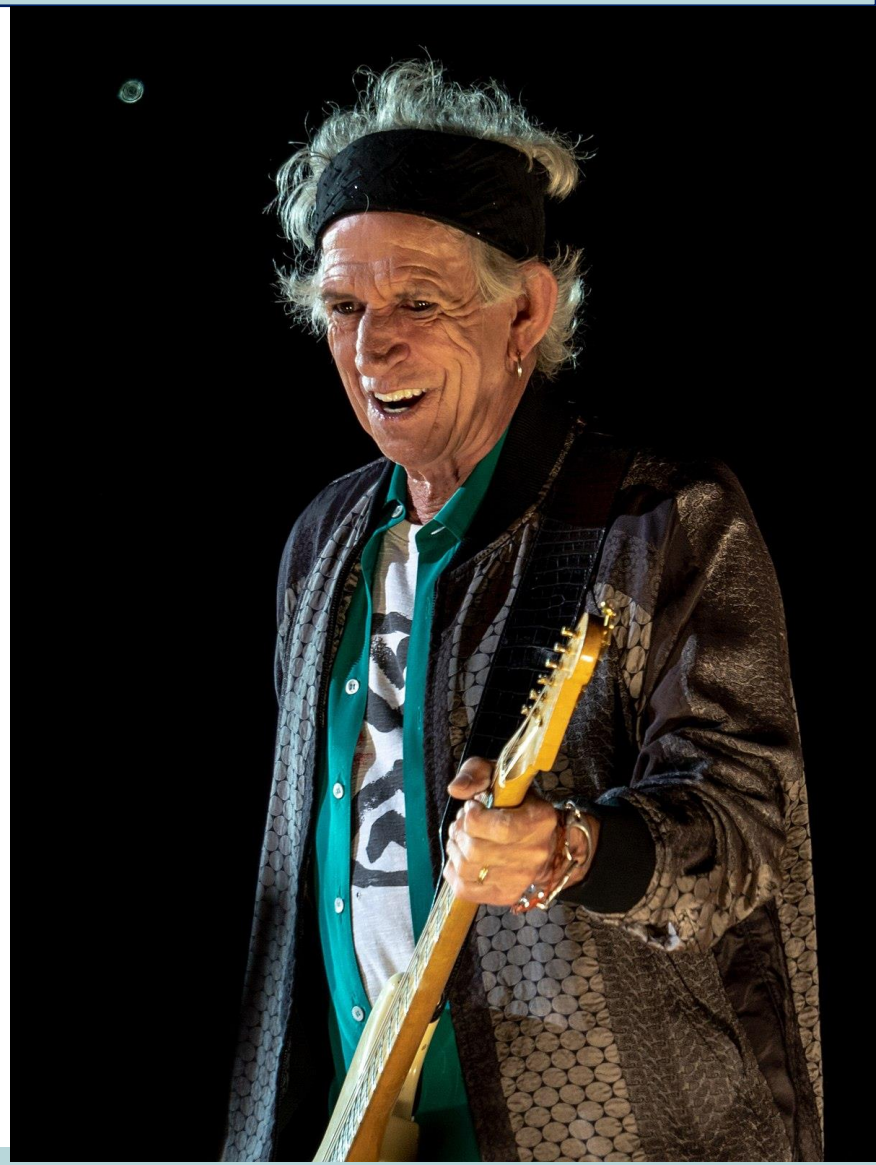
DATENBANKDESIGN WIRD UNTERSCHÄTZT

- Datenbankdesign wird unterschätzt
  - Von Entwicklern
  - Von DBAs
  - Von SQL-Community
  - Von Datenbankberatern
  - Von Microsoft und anderen Vendors

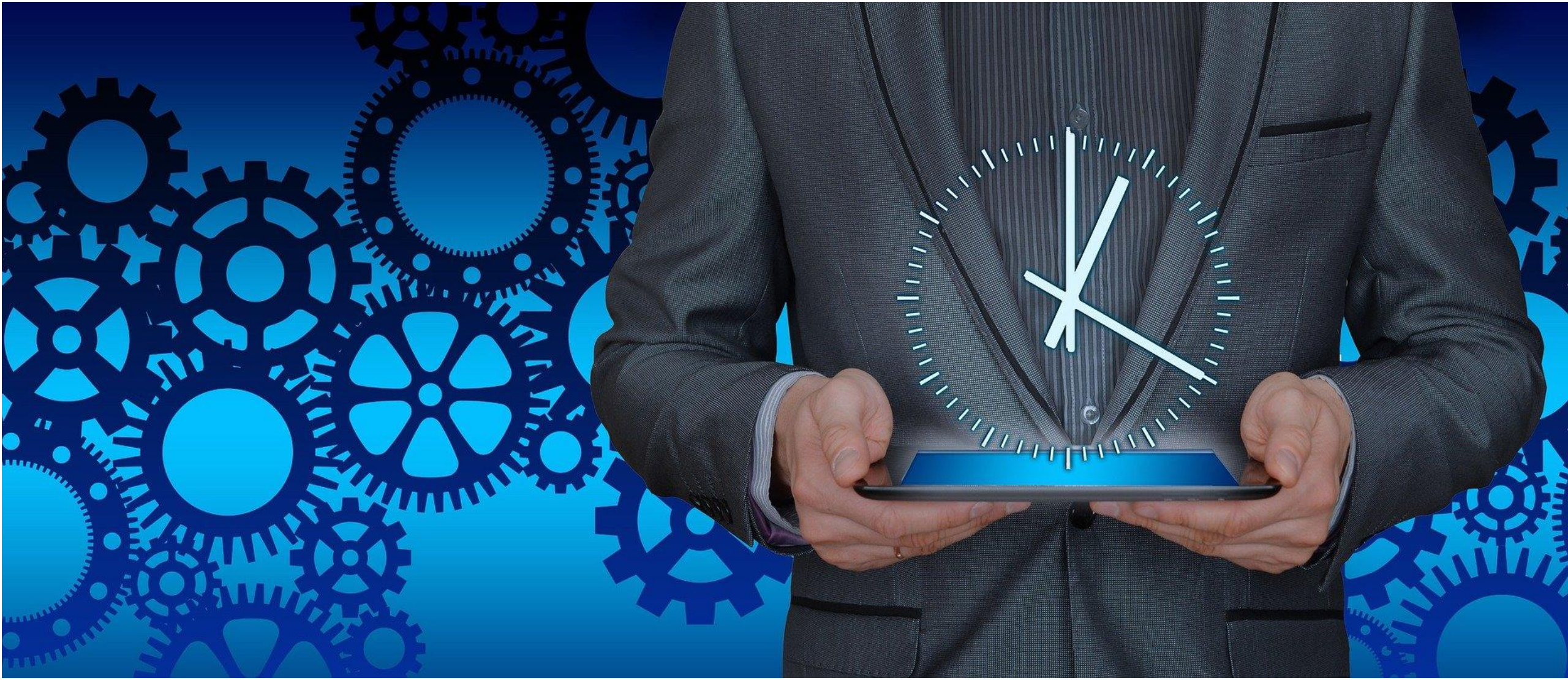
- Deshalb nimmt man nicht genug Zeit für Tabellen- und Datenbankdesign

DATENBANKDESIGN

- Datenbankdesign ist die Grundlage eines Projekts
- Datenbanken überleben in der Regel Applikationen und Services
- Gutes Design spart Geld, Zeit und Reputation



NEHMEN SIE SICH ZEIT!



NEHMEN SIE SICH ZEIT!

- Unterschätzen Sie Datenbankdesign nie!
- Die Implementierung der Geschäftslogik beginnt mit CREATE TABLE
- Do not create a table in two minutes!
- Nehmen Sie sich genug Zeit für die Analyse der Anforderungen
- Änderungen sind teuer und können sehr schmerzhaft sein

2

MACH MA SPÄTER!



MACH MA SPÄTER!

- Agile
  - Oft eine Ausrede, um Sachen nicht richtig umzusetzen
  - Wir haben keine Zeit, wir müssen liefern
  - Wir konzentrieren uns auf jetzt, auf eine Story
  - Über Scalability, Performance reden wir später!

- Archivierung
  - Es gibt keine Out-of-box Lösung für die Archivierung => es gibt keine Archivierung

MACH MA SPÄTER - EPILOG

- Viele Tabellen, die 100M+ Zeilen haben, obwohl wir nur < 100K tatsächlich brauchen
- Wir machen Backups, Restores
- Wir indizieren brav all diese Tabellen
- Archivierung wird immer verschoben

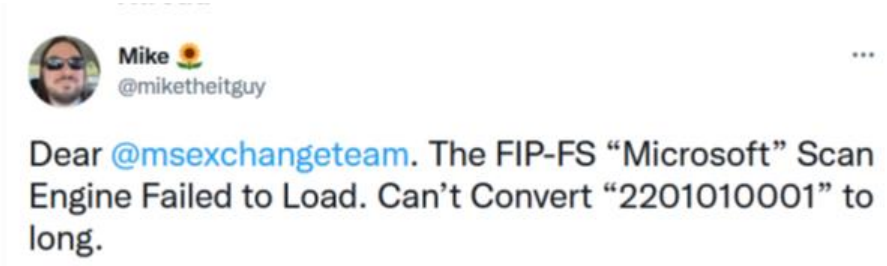
3

UNGEEIGNETER DATENTYP



JAHR 2022 BUG

"Jahr 2022 Bug": Microsoft-Fehler bei Exchange sorgt für weltweite Mail-Ausfälle



Remember the Y2K bug? Microsoft confirms new Y2K22 issue

which resulted in the antivirus engine crashing, which meant emails got stuck in transport queues and were not being delivered

Y2K22-Bug stoppt Exchange-Mailzustellung: Antimalware-Engine stolpert über 2022

JAHR 2022 BUG

- MS Exchange Problem 1.1.2022
  - Falsch gewählter Datentyp
  - 31.12.2021. 14:59 => 2112311459
  - 01.01.2022. 00:01 => 2201010001
  - Arithmetic overflow
  - Workaround: 33rd December 2021 LOL



DATUM

- Basteln Sie nie ein Datum aus Zahlen oder mit Text!

- Verwenden Sie den richtigen Datentyp, es gibt genug davon
  DATE, DATETIME, DATETIME2, DATETIMEOFFSET

INT DATENTYP FÜR „KLEINE“ SPALTEN

- Welchen Datentyp verwenden Entwickler am häufigsten für Spalten **Status**, **Typ** oder **Art**?
- **INT**
- INT ist ein sehr flexibler Datentyp => 4 294 967 295 mögliche Stati

$$-2^{31} \leq INT \leq 2^{31} - 1$$

$$-2{,}147{,}483{,}648 \leq INT \leq 2{,}147{,}483{,}647$$

- Für den Status nehmen Sie nicht INT, sondern TINYINT
- INT benötigt 4 Bytes und die Größe einer Spalte kann eine große Rolle beim Performance spielen

DATA TYPE SIZE DOES MATTER

```
CREATE TABLE dbo.StudentExams1(
    exam_number int NOT NULL,
    student_id int NOT NULL,
    exam_id int NOT NULL,
    exam_note int NULL,
    exam_date datetime  NULL,
 CONSTRAINT PK_StudentExams1 PRIMARY KEY CLUSTERED (exam_number ASC)
)
GO

CREATE TABLE dbo.StudentExams2(
    exam_number int NOT NULL,
    student_id int NOT NULL,
    exam_id int NOT NULL,
    exam_note tinyint NULL,
    exam_date date NULL,
 CONSTRAINT PK_StudentExams2 PRIMARY KEY CLUSTERED (exam_number ASC)
)
GO
```

DATA TYPE SIZE DOES MATTER

```
DECLARE @date_from DATETIME = '20000101';
DECLARE @date_to DATETIME = '20171231';
DECLARE @number_of_rows INT = 10000000;
INSERT INTO dbo.StudentExams1
SELECT n AS exam_number,
    1 + ABS(CHECKSUM(NEWID())) % 50000 AS student_id,
    1 + ABS(CHECKSUM(NEWID())) % 40 AS exam_id,
    5 + ABS(CHECKSUM(NEWID())) % 6 AS exam_note,
(SELECT(@date_from +(ABS(CAST(CAST( NewID() AS BINARY(8) )AS INT))
% CAST((@date_to - @date_from)AS INT)))) exam_date
FROM dbo.GetNums(@number_of_rows)
GO
```

DATA TYPE SIZE DOES MATTER

```
--Check table size
SELECT  OBJECT_NAME(s.object_id) AS table_name, CAST((s.used_page_count/128.0) AS int) ta
FROM sys.dm_db_partition_stats AS s
INNER JOIN sys.indexes AS i
    ON s.[object_id] = i.[object_id] AND s.index_id = i.index_id
INNER JOIN sys.tables AS t
    ON s.[object_id] = t.[object_id]
WHERE s.object_id IN ( OBJECT_ID('dbo.StudentExams1'), OBJECT_ID('dbo.StudentExams2')) AN
/*
table_name                                table_size_MB
----------------------------------------- -------------
StudentExams1                             320
StudentExams2                             240
*/
```

+25% STORAGE

DATA TYPE SIZE DOES MATTER

```
--Check the data page number
SELECT OBJECT_NAME(p.object_id) AS table_name, data_pages
FROM sys.allocation_units AS a
INNER JOIN  sys.partitions  AS p
  ON a.container_id = p.partition_id
WHERE p.object_id IN ( OBJECT_ID('dbo.StudentExams1'), OBJECT_ID('dbo.StudentExams2'))
GO
/*
table_name                     data_pages
------------------------------ --------------------
StudentExams1                  40842
StudentExams2                  30960
*/
```

+30% DATA PAGES

UNTERSCHÄTZUNG

- Man kann auch die Datentypgröße unterschätzen
- Es ist sehr mühsam, teuer und sogar gefährlich nach Jahren den Datentyp für eine Spalte von INT auf BIGINT zu ändern!

- INT vs. BIGINT
  - Wenn Sie glauben, dass es 100M+ Zeilen sein werden, nehmen Sie BIGINT!

- SMALLINT vs. INT
  - Wenn Sie glauben, dass es 5K+ Zeilen sein werden, nehmen Sie INT!

WÄHLEN SIE DEN ENTSPRECHENDEN DATENTYP

- Wählen Sie den kleinsten Datentyp, der ausreicht, um den erforderlichen Bereich abzudecken
- Wenn Sie einen robusten Datentyp wählen
  - Er braucht mehr Speicherplatz, Sicherung und Wiederherstellung sind langsamer
  - SQL Server muss mehr Seiten lesen, um das gleiche Ergebnis zu generieren
- Wählen Sie den Datentyp DATE, wenn Sie keine Genauigkeit in Minuten oder Sekunden benötigen
- Datentypänderungen sind sehr teuer, schwer zu verkaufen und niemand hat Interesse, dafür zu bezahlen

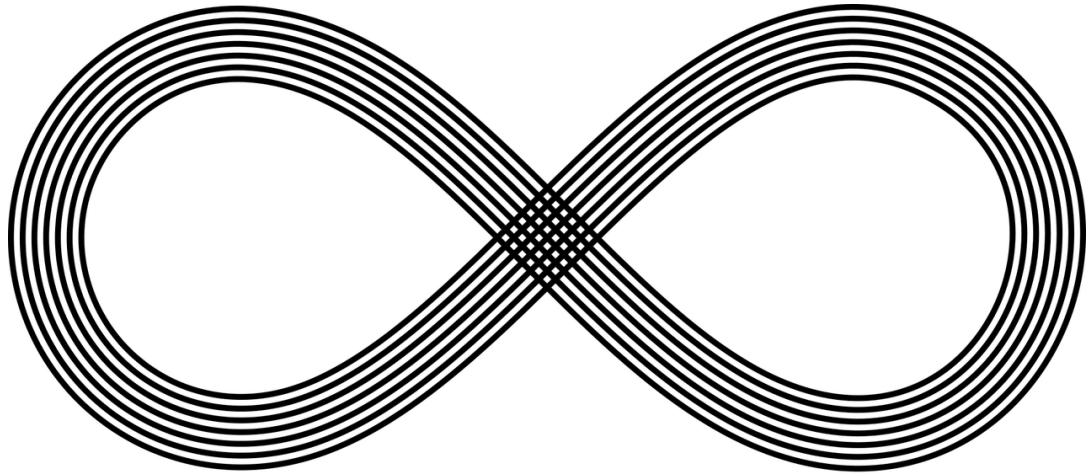
CHOOSE THE APPROPRIATE DATA TYPE

- Changes that are necessary when changing the data type:
  - ALTER the data type of the column within the table
  - Update the parameters of stored procedures that use this column
  - Update the class in the application code
  - Up
  - Th

```
IF OBJECT_ID('dbo.insertStudentExams','U') IS NULL
    DROP PROCEDURE dbo.insertStudentExams
GO
CREATE PROCEDURE dbo.insertStudentExams
```

```
//se.ExamNote = dr.GetInt32(2);
se.ExamNote = (int) dr.GetByte(2);
```

4



KEINE CONSTRAINTS!

CHECK CONSTRAINTS

- Ein typisches Ergebnis dieser Abfrage

```
select * from sys.check_constraints
```

176 %

Results | Messages

name	object_id	principal_id	schema_id	parent_object_id	type	type_desc	create_date	modify_date	is_ms_shipped	is_published	is_schema_published	is_disabled	is_not_for_replication	is_not_trusted	p

- Keine Constraints!

EIN BEISPIEL

- die Spalte heißt **Note** und modelliert die Note im Schulsystem
  - Datentyp: meistens INT
  - Constraints: keine

- Es soll nicht nur TINYINT sein, sondern auch CHECK CONSTRAINT angelegt sein, um Werte auf 1, 2,3 4, oder 5 einzuschränken

- Typische Ausrede: Business Logic belongs to Business Layer

CONSTRAINTS

- CHECK Constraints
  - In einer Spalte zulässige Grenzwerte
  - Jeder wird die Werte in der Spalte gleich interpretieren
  - Erhöhen Sie die Datenqualität

- UNIQUE Constraints
  - Wenn etwas unique ist, verwenden Sie die UNIQUE Constraints und teilen Sie diese Information mit dem RDBMS-System!

- Das ist gut nicht für Integrität sondern auch für Performance!

DATABASE CONSTRAINTS

CHECK CONSTRAINTS

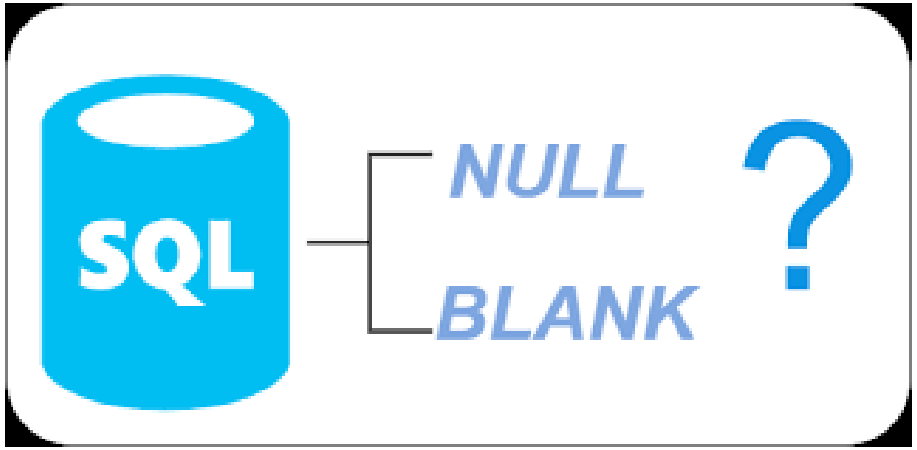




Business Logic belongs to Business Layer

5

ZU VIELE NULL-WERTE



*

* Credits for the image: https://sqlskull.com

ISSUES WITH NULLS

- Three Valued Logic
- Unterschiedliche Interpretationen
- Performance

p	NOT *p*
T	F
U	U
F	T

p	*q*	*p* AND *q*
T	T	T
T	U	U
T	F	F
U	T	U
U	U	U
U	F	F
F	T	F
F	U	F
F	F	F

p	*q*	*p* OR *q*
T	T	T
T	U	T
T	F	T
U	T	T
U	U	U
U	F	U
F	T	T
F	U	U
F	F	F

THREE VALUED LOGIC

```
CREATE TABLE dbo.Color (id INT NOT NULL, name VARCHAR(30) NOT NULL)
GO
INSERT INTO dbo.Color(id, name) VALUES(1, 'Black'),(2, 'White'), (3, 'Purple');
GO
SELECT * FROM dbo.Color;
/*Results
id          name
----------- ------------------------------
1           Black
2           White
3           Purple
*/
```

```
SELECT DISTINCT Color
FROM Production.Product;
```

109 %

Results | Messages

	Color
1	NULL
2	Black
3	Blue
4	Grey
5	Multi
6	Red
7	Silver
8	Silver/Black
9	White
10	Yellow

THREE VALUED LOGIC

```
SELECT * FROM dbo.Color
WHERE name IN (SELECT Color FROM AdventureWorks2019.Production.Product);
```

9 %

Results

id	name
1	Black
2	White

```
SELECT * FROM dbo.Color
WHERE name NOT IN (SELECT Color FROM AdventureWorks2019.Production.Product);
```

109 %

Results

id	name

(0 rows affected)

THREE VALUED LOGIC

```sql
SELECT * FROM dbo.Color
WHERE name NOT IN (SELECT Color FROM AdventureWorks2019.Production.Product)
UNION ALL
SELECT * FROM dbo.Color
WHERE name IN (SELECT Color FROM AdventureWorks2019.Production.Product);
```

09 %

Results

```
id          name
----------- ------------------------------
1           Black
2           White

(2 rows affected)
```

PURPLE FEHLT!

UNTERSCHIEDLICHE INTERPRETATIONEN

- auf der Grundlage von NULL-Werten, soll man keine Entscheidungen treffen

Id	FirstName	LastName	BirthDate	City	ZipCode	AddressLine
2	Vlado	Kreslin	1953-11-29	NULL	NULL	NULL

- Keine Adresse, keine bekannte Adresse, obdachlos, GDPR...?

- Aber die Leute treffen Entscheidungen...

id	isConvicted
1	1
2	0
3	NULL
4	NULL
5	0

NULL-WERTE UND DATENQUALITÄT

- NOT NULL

```
INSERT INTO dbo.Customers2 (Id, FirstName, LastName)
VALUES(5, N'Vlado', N'Kreslin');
```

```
109 %
Messages
Msg 515, Level 16, State 2, Line 37
Cannot insert the value NULL into column 'BirthDate', table 'DBDesign.dbo.Customers2'; column does not allow nulls. INSERT fails.
The statement has been terminated.
```

- NULL
- INSERT/UPDATE funktioniert!
- Aber...

Id	FirstName	LastName	BirthDate	City	ZipCode	AddressLine
1	Zoran	Predin	1958-06-16	Maribor	2000	Lačni Franc 1
2	Vlado	Kreslin	1953-11-29	NULL	NULL	NULL
3	asssd	effrefe	NULL	NULL	NULL	NULL
4	aaa	bbb	NULL	NULL	NULL	NULL

NULL-WERTE UND DATENQUALITÄT

- Reduzieren Sie die Anzahl der Spalten, die NULL zulassen
  - NULL bringt den Overhead mit sich
  - Der Anwendungscode muss Datensätze verarbeiten, die NULL enthalten
- Verschiedene Akteure haben oft unterschiedliche Interpretationen der Zeilen, in denen die Spalten NULL akzeptieren
- Zeigen Sie, dass Ihnen die Qualität und Konsistenz der Daten am Herzen liegt
- Manchmal ist es unmöglich, NULL zu vermeiden, aber Sie sollten es zumindest versuchen
- Wenn Sie nicht schnell beantworten können, warum eine Spalte NULL unterstützen sollte, machen Sie sie zu NOT NULL

FAZIT

- Die Implementierung der Geschäftslogik beginnt mit CREATE TABLE
- „Business Logic belongs to Business Layer“ und „ Database is a storage“ – zwei gefährliche Mythen in der Programmierung
- Die Datenbank überlebt normalerweise die Anwendungen und Services
- Nehmen Sie sich ausreichend Zeit, um Tabellendesign richtig zu machen

FAZIT

- Für jede Spalte in einer Tabelle:
  - Wählen Sie den kleinsten Datentyp, der die Attributdomäne abdeckt
  - Wenn die Spalte nur bestimmte Werte akzeptieren kann, verwenden Sie die CHECK-Einschränkung
  - Wenn etwas unique ist, teilen Sie diese Info mit der Datenbank
  - Reduzieren Sie die Anzahl der nullablen Spalten

SESSION FEEDBACK

