Nejčastější chyby při návrhu a implementaci databáze

RNDr. David Gešvindr, Ph.D.
MVP: Data Platform | MCSE: Data Platform | MCT
david@wug.cz
@gesvindr

Mgr. Vladimír Mužný
MVP: Data Platform | MCSE: Data Platform | MCT
vladimir.muzny@dropman.cz
@VladimirMuzny

Osnova

1. Chyby v databázovém designu
2. Nevhodné dotazování
3. Nevhodné objekty

Osnova

1. **Chyby v databázovém designu**
2. Nevhodné dotazování
3. Nevhodné objekty

Chyby v databázovém designu: Normalizace

- Vynechání normálních forem
  - Ukládání opakujících se skupin hodnot (vícejazyčný web)
  - Ukládání hodnot, které se dají rozdělit („Ing. Jan Novák“)
  - Redundance hodnot ve sloupci
  - Chybějící „číselníky“

- Nevhodné návrhové vzory
  - „Řídké“ schéma tabulky vs. Databázová „dědičnost“
  - Generické úložiště
  - XML + JSON

Chyby v databázovém designu: Constraints

- Chybějící klíče způsobují problémy v reálném životě…
  …natož pak v databázi
- Validace dat na více úrovních: v databázi / aplikaci / UI
- **Primární klíč**
  - Chybějící primární klíč znemožňuje se jednoznačně odkázat na daný záznam
  - Nevhodně zvolený primární klíč způsobuje:
    - Nutnost kaskádových změn klíčové hodnoty (přirozený klíč)
    - Riziko duplicity (přirozený klíč)
    - Riziko chyby ve spojení s cizí tabulkou (vícepoložkový klíč)
    - Fragmentaci dat (Clustered Primary Key nad nerostoucí/neklesající hodnotou)

Chyby v databázovém designu: Constraints

- **Cizí klíč**
  - Osamocené záznamy
- **Check Constraint**
  - Validace platnosti řádku pomocí predikátu
- **Unique Constraint**
- **Default Constraint**

Chyby v databázovém designu: NULL

- NULL je projevem tříhodnotové logiky (true/false/not known)

- Špatný návrh NULL může způsobit
  - Nutnost zadat neznámá data náhradní hodnotou (např.: datum ukončení pracovního poměru k 1. 1. 1900, kde hrozí velké riziko chyby)
  - Možnost nezadat nic
  - Nespolehlivé dotazování

Chyby v databázovém designu: datové typy

- Všechno NENÍ VARCHAR!
- Co je VARCHAR, může být NVARCHAR
- Všechno není VARCHAR(MAX)
- Uniqueidentifier NENÍ VARCHAR
- Float se opravdu do účetnictví nehodí
- Časová složka v datu narození VADÍ
- Datový typ je důležitý podklad pro optimalizaci dotazu

Chyby v databázovém schématu: schémata

- Schéma je nedílnou součástí názvu objektu
- Schéma pomáhá rozdělit objekty v databázi podle podproblémů
- Schéma pomáhá zabezpečit objekty podle významu!
- Problém: historicky mnoho databází používá jen schéma DBO
- Jak na schémata?
  - Téměř každý projekt dnes obsahuje více use-cases
  - Výčet aktorů v use-cases pomáhá pochopit role uživatelů ➔ uživatelé mají různá oprávnění

Osnova

1. Chyby v databázovém designu
2. **Nevhodné dotazování**
3. Nevhodné objekty

Nevhodné dotazování: Návrh dotazu

- Efektivní dotaz načítá z databáze jen **nutné minimum informací**, které dostačuje aplikaci pro daný účel
  - Např.: Pro vypsání seznamu objednávek nemusím načítat 50 dalších sloupců v tabulce objednávek, když nejsou zobrazeny v UI

- Je důležité:
  - Vracet jen sloupce, co skutečně využijeme (pozor na SELECT * FROM tabulka)
  - Filtrovat a stránkovat záznamy na serveru
  - Zbytečně neřadit záznamy, pokud to opravdu nepotřebujeme

Nevhodné dotazování: Nevhodný JOIN

- Při více kritériích spojení (často složený PK) se zapomene na některé z nich
  - Výsledkem je pomalý dotaz
  - Výsledkem je částečný Kartézský součin
    - Občas nevhodně „léčen“ pomocí slova DISTINCT
- Zbytečné užití OUTER JOINu
  - Zbytečné riziko chyby ve výsledku
  - Diskvalifikace vhodné optimalizace
  - Nečitelné dotazy

Nevhodné dotazování: NoSARGABLE podmínky

- Pokud je v podmínce nevhodně zapsaný výraz či skalární funkce, dojde k nutnosti vyhodnotit hodnotu výrazu pro každý řádek tabulky

- Výraz v podmínce WHERE by měl být, je-li to možné, tzv. „SARGABLE“ [1]
  - Podmínku lze rozhodnout na úrovni operátoru přistupujícího k datům podle hodnot přečtených v tabulce či indexu
  - To umožňuje efektivně číst pouze podmnožinu řádků použitím operace Index Seek

- Nefiltrovat dle hodnoty výrazu, kde dochází k transformaci hodnot sloupců
  - Je nutné používat skalární funkce ve výrazu podmínky tak, aby neměly jako vstupní parametry hodnoty sloupců

Zdroj: [1] https://cs.stanford.edu/people/chrismre/cs345/rl/selinger.pdf

Nevhodné dotazování: Hodnota NULL

- Dotazování na hodnotu NULL:
  - Polozka = @p OR (Polozka IS NULL AND @p IS NULL)
  - Nebo COALESCE(Polozka, „hodnota“) = „hodnota“
    - Druhý příklad eliminuje rizikovou kombinaci logických spojek AND a OR
      - a*(b + c) != a*b + a*c

- Pozor na použití operátoru NOT IN
  - Pokud množina obsahuje hodnotu NULL, nikdy nebude výsledek PRAVDA

Nevhodné dotazování: Implicitní konverze

- Při porovnání různých datových typů v podmínce dotazu musí dojít k implicitní konverzi na stejný datový typ

- Pokud směr implicitní konverze vede ke konverzi sloupce v tabulce místo parametru, musí se konverze vyhodnotit pro všechny hodnoty v tabulce
  - Směry implicitní konverze jsou popsány v dokumentaci [1]

- Implicitní konverze navíc vede k nemožnosti odhadnout kardinalitu

Zdroj: [1] https://docs.microsoft.com/en-us/sql/t-sql/data-types/data-type-precedence-transact-sql?view=sql-server-ver15

Osnova

1. Chyby v databázovém designu
2. Nevhodné dotazování
3. **Nevhodné objekty**

Nevhodné objekty: Kurzory

- Kurzor je smyčka
- Smyčka je VŽDY potenciální výkonnostní problém
  - A hlavně v transakci!
  - I jednoduchá akce se může spustit mnohokrát
- Statický kurzor zatěžuje tempdb
- Pokud možno hledat dávkovou náhradu zpracování

Nevhodné objekty: Funkce

- **Skalární funkce** má nezávislé vyhodnocení pro každý svůj běh
  - S výjimkou v SQL Serveru 2019 při použití Intelligent Query Processingu
- **Multi-statement table valued funkce** se pokaždé vyhodnocuje nezávisle
  - Pozor na použití v operátoru APPLY
  - Chybný odhad kardinality 100
- Pouze **In-line table valued funkce** se stejně jako pohled stane součástí exekučního plánu dotazu, který se optimalizuje jako celek

Nevhodné objekty: Triggery

- Trigger:
  - Není vidět
  - Nehodí se pro auditovací akce
    - není SELECT TRIGGER: čtení krade data
    - Dá se vypnout „ALTER TABLE x DISABLE TRIGGER tr“
  - Je výkonnostně nebezpečný
    - Kurzor v triggeru
  - Je „nekonečný“
    - Vnořování triggerů
  - Nespolehlivé pořadí
    - sp_settrigerorder jen pro first a last
  - Rekurze
    - Max 32 úrovní (a i to je moc)

Nevhodné objekty: @tabulky vs. #tabulky

Table Variables

- Platnost v rámci dávky
- Ukládá se do tempdb
- Méně rekompilací uložených procedur
- Neprovede se na nich ROLLBACK
- Má pevný odhad počtu řádků 1
  - S výjimkou SQL Serveru 2019

- Jak je dostat jen do paměti?

Session Temporary Table

- Platnost v rámci session
- Ukládá se do tempdb
- Více rekompilací uložených procedur
- Má správný odhad počtu řádků

Další zdroje informací

- Problémy s transakcemi a zámky:
  - https://www.wug.cz/zaznamy/331-SQL-Server-Bootcamp-2016-Transakce-zamky-a-izolacni-urovne-v-SQL-Serveru
  - https://www.wug.cz/zaznamy/705-Optimistic-concurrency-control-in-Microsoft-SQL-Server
- Problémy s OR mappery:
  - https://www.wug.cz/zaznamy/606-SQL-Server-Bootcamp-2019-Entity-Framework-vs-SQL-Server-Co-jste-mohli-udelat-a-neudelali-a-co-jste-delat-nemeli-a-presto-udelali
- Problémy s tempdb:
  - https://www.wug.cz/zaznamy/535-SQL-Server-Bootcamp-2019-TempDB-Internals

Další zdroje informací

- Kompletní vývojářský kurz na ladění výkonu databázových dotazů:
  - GOC 631 Optimalizace, ladění a monitorování T-SQL dotazů

- Kompletní administrátorský kurz na ladění výkonu SQL Serveru:
  - GOC 630 Monitorování, optimalizace a ladění výkonu Microsoft SQL Serveru

Dotazy

RNDr. David Gešvindr, Ph.D.
MVP: Data Platform | MCSE: Data Platform | MCT
david@wug.cz
@gesvindr

Mgr. Vladimír Mužný
MVP: Data Platform | MCSE: Data Platform | MCT
vladimir.muzny@dropman.cz
@VladimirMuzny