Enterprise funkce SQL Serveru 2016, které jsou od SP1 zdarma

RNDr. David Gešvindr

MVP: Data Platform | MCSE: Data Platform | MCSD: Windows Store | MCT

david@wug.cz

@gesvindr

SQL Server 2016 Service Pack 1

- Vydán v listopadu 2016
- Zpřístupňuje všechny enterprise funkce SQL Serveru nehledě na edici, které **mají dopad na vývoj databázového řešení**:
  - InMemory OLTP
  - Columnstore Index
  - Dynamic Data Masking, Row-level Security, Always Encrypted
  - Partitioning, Compression, Database Snapshot, Change Data Capture
  - Fine grained auditing, Multiple filestream containers
  - PolyBase

Osnova

1. Funkce zaměřené na výkon
2. Funkce zaměřené na zabezpečení

Osnova

1. **Funkce zaměřené na výkon**
   - In-memory OLTP
   - Column Store Indexy a Real-Time Operational Analytics
   - Partitioning
   - Komprese dat
   - Databázové snapshoty
2. Funkce zaměřené na zabezpečení

In-memory OLTP

- SQL Server 2014 přidává podporu pro **Memory Optimized Tabulky** a **nativně kompilované uložené procedury**
- Jedná se o nový engine pro uložení a zpracování dat přímo v operační paměti
  - Nevyužívá zámky, ale multi-version optimistic concurrency
  - Je zaručena persistence dat při výpadku díky využití transakčního logu
- Dochází až k **30 násobnému zvýšení počtu zpracovaných transakcí** za vteřinu

Výkon In-memory OLTP

- In-memory technologie poskytuje vysoký výkon
- Microsoft v testech demonstroval [1]:
  - 1 milion batch requests/s při čtení a zápisu 4KB dat
  - Ukládání řádků rychlostí 10 milionů hodnot za vteřinu (100B na řádek)
  - Zpracování transakcí s objednávkami rychlostí 260 000 transakcí za vteřinu

- In-memory řešení a zkušenosti s ním popisuje i společnost bwin [2]:
  - Použití pro cachování session state webové aplikace
  - Ve verzi 2016 dosáhli přes 1 200 000 batch req/sec

[1] Kalen Delaney: SQL Server In-Memory OLTP Internals for SQL Server 2016, Technical White Paper
[2] https://blogs.msdn.microsoft.com/sqlcat/2016/10/26/how-bwin-is-using-sql-server-2016-in-memory-oltp-to-achieve-unprecedented-performance-and-scale/

Vhodné použití In-Memory OLTP

- Problémy se zámky (locks i latches)
- Vysoká I/O zátěž
- Veliké množství INSERTů a SELECTů
- Požadavky na velmi rychlé business transakce
- Ukládání session z webserverů

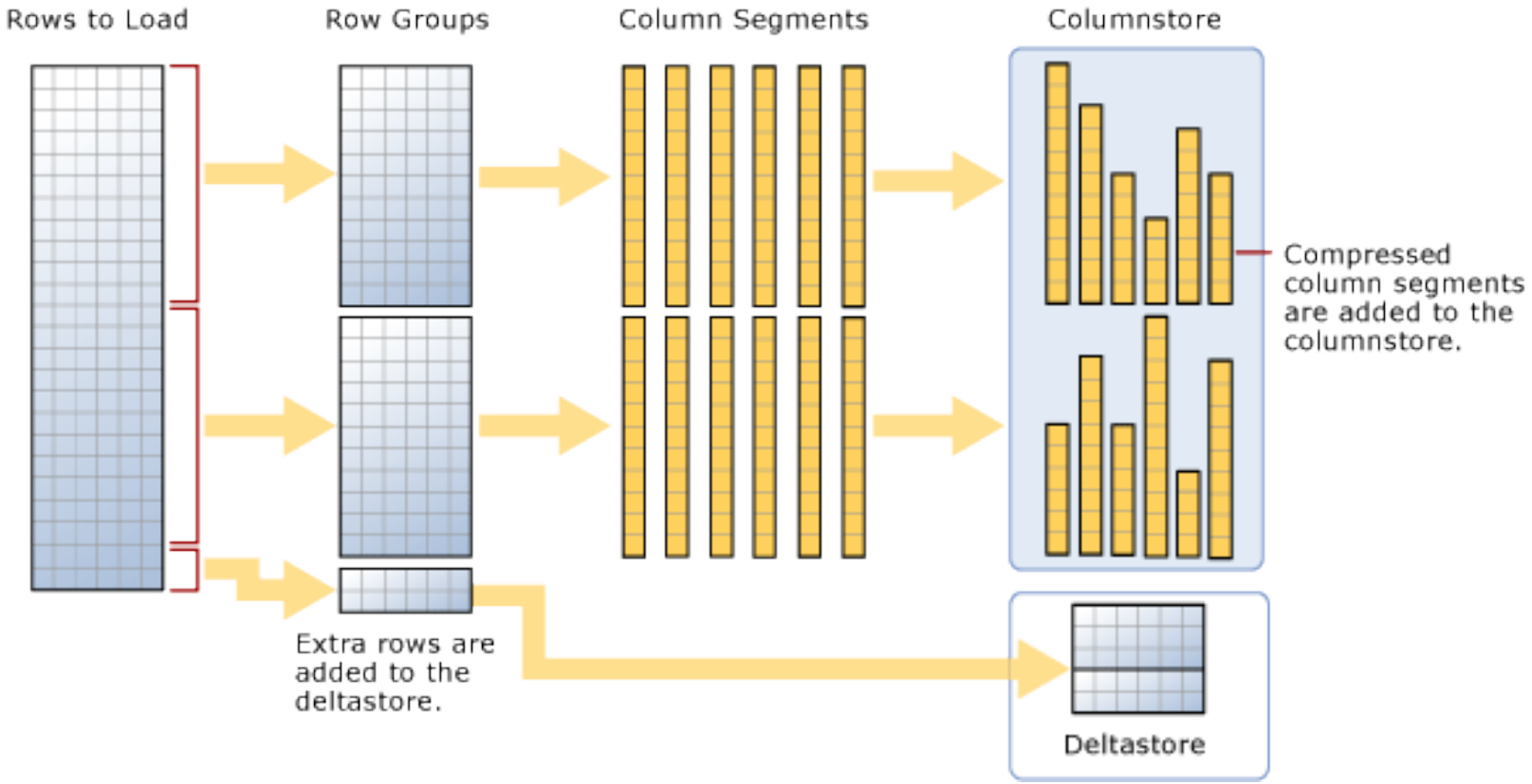
- Nevhodné použití:
  - Jiná zátěž než OLTP (datové sklady)
  - Závislost aplikace na mechanizmu zámků

Columnstore Index

- V SQL Serveru běžně platí, že data jsou uložena **po řádcích** v clustered indexu nebo heap
- **Columnstore Index ukládá data komprimovaná po sloupcích**
- S pomocí Columnstore Indexů lze dosáhnout až 100x zvýšení výkonu analytických dotazů
  - Rozsáhlé agregace nad datovým skladem, které čtou veliké rozsahy řádků
- Velmi silná komprese umožňuje dosáhnout až 10 násobného zmenšení uložených dat – rozsáhlé tabulky faktů

Terminologie Columnstore Indexů

- Data jsou rozdělena do RowGroups (až 1 048 576 řádků)



Zdroj: https://docs.microsoft.com/en-us/sql/relational-databases/indexes/columnstore-indexes-data-loading-guidance

Různé verze Columnstore Indexů

- SQL Server 2012
  - Přinesl podporu pro **Read-only nonclustered columnstore index**
  - Využití pro Data Warehouse, složité vkládání nových dat
- SQL Server 2014
  - Uvedl **Updateable clustered columnstore index**, ale nebylo možné vytvořit jiné nonclustered indexy
  - Využití pro Data Warehouse, nevhodné pro OLTP zátěž
- SQL Server 2016 uvádí 2 novinky:
  - **Updateable nonclustered columnstore index**
  - Btree index on a clustered columnstore index

Vhodné použití Column Store indexů

- Dotazy, které **čtou celou tabulku**
  - Agregace dat
  - Rozsáhlé tabulky faktů v datových skladech

- Cloumn Store Index **nepodporuje operaci seek**
  - **Nevhodný pro vyhledávání individuálních řádků**

- Rozšíření OLTP tabulek s B+ stromovým clustered indexem o **aktualizovatelný non-clustered column store index**

Jak Column Store indexy zvyšují výkon

- **Data Compression**
  - Až 10x menší objem data uložený na disku a načítaný z disku (menší I/O)
  - Data se uchovávají komprimovaná i v operační paměti

- **Column Elimination**
  - Díky uložení po sloupcích se nemusí vůbec načítat data sloupců, se kterými nepracujeme

- **Rowgroup Elimination**
  - U každé rowgroup si SQL ukládá metadata o minimální a maximální hodnotě v každém sloupci
  - Filtrování na úrovni celých rowgroups

Zdroj: https://docs.microsoft.com/en-us/sql/relational-databases/indexes/columnstore-indexes-query-performance

Jak Column Store indexy zvyšují výkon

- **Batch Mode Execution**
  - Vybrané operátory se spouští v dávkovém režimu (batch mode), kdy pracují až s 900 řádky najednou, místo spouštění řádek po řádku (row mode)
  - Umí pracovat nad komprimovanými daty a snižují zbytečnou zátěž

- **Aggregate Pushdown**
  - Vybrané operace na výpočet agregačních funkcí jsou přeneseny až na operaci SCAN, která načítá data
  - Pouze funkce MIN, MAX, SUM, COUNT, AVG nad datovými typy <= 64 bit

Zdroj: https://docs.microsoft.com/en-us/sql/relational-databases/indexes/columnstore-indexes-query-performance

Real-Time Operational Analytics

- V současnosti se pro analýzu dat z OLTP systému data kopírují do datového skladu, kde se pak analyzují
  - S tím jsou spojené další náklady na nasazení a provoz
  - Komplexní řešení složené z několika propojených služeb
  - Toto řešení pracuje se zpožděnými daty

- **Real-Time Operational Analytics**
  - Pokud v OLTP databázi informačního systému vytvoříme **Updateable nonclustered columnstore index**
  - Minimální dopad na výkon OLTP databáze
  - Možnost spouštět analytické dotazy s vysokým výkonem

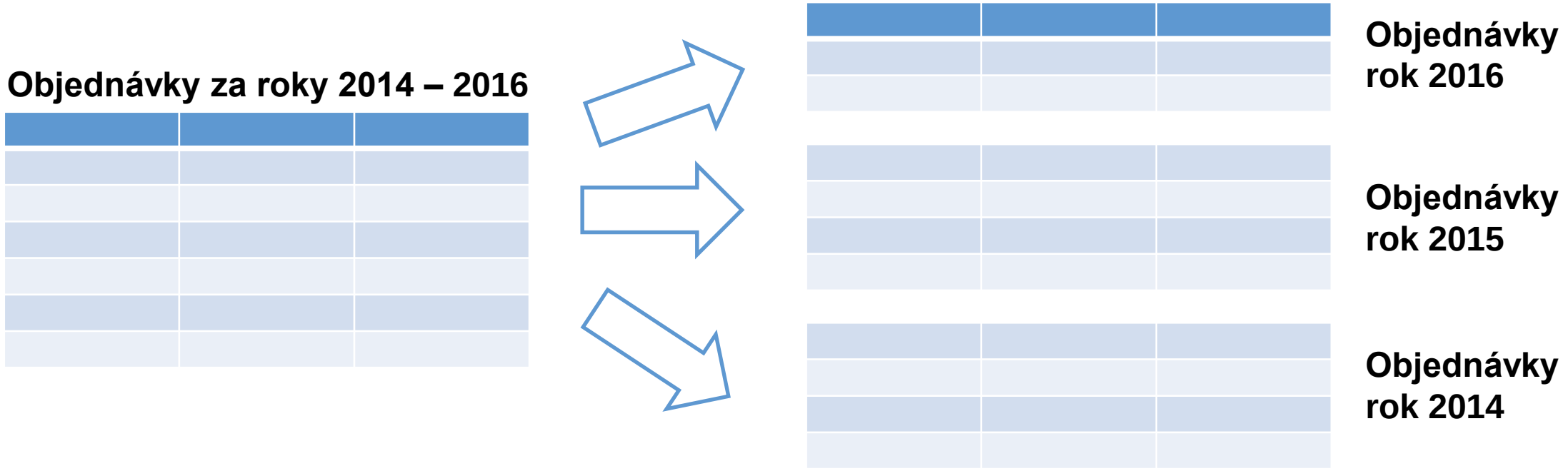
Real-Time Operational Analytics

- Columnstore index je možné vytvořit nad klasickou tabulkou na disku, přičemž i zde se dostaví znatelné zlepšení výkonu při dotazování

- Je také možné vytvořit **In-Memory Columnstore Index**
  - Vytváří se vždy nad všemi sloupci memory optimized tabulky
  - Další zvýšení výkonu, protože se index nenačítá z disku
  - Snížení režie při zápisu, protože i deltastore je jako memory optimized tabulka

Partitioning

- Partitioning je horizontální rozdělení tabulky nebo indexu
- Řádky tabulky jsou rozděleny mezi jednotlivé partitions
  - Místo 1 rozsáhlého clustered indexu máme nyní více menších indexů
  - Každá partition obsahuje pouze řádky dle definovaného kritéria



Kdy se hodí použít partitioning

- Izolace archivních dat
  - Možnost omezit údržbu indexů a statistik nad historickými daty
  - Umístění „studených dat“ na pomalé disky

- Efektivní implementace sliding-window
  - Pravidelně přidáváme nová data a promazáváme historická data
  - Odmazání historických dat přesunem do jiné tabulky
  - Při použití partitioningu je možné „zaměňovat“ partitions mezi identickými tabulkami ve stejné FILEGROUP

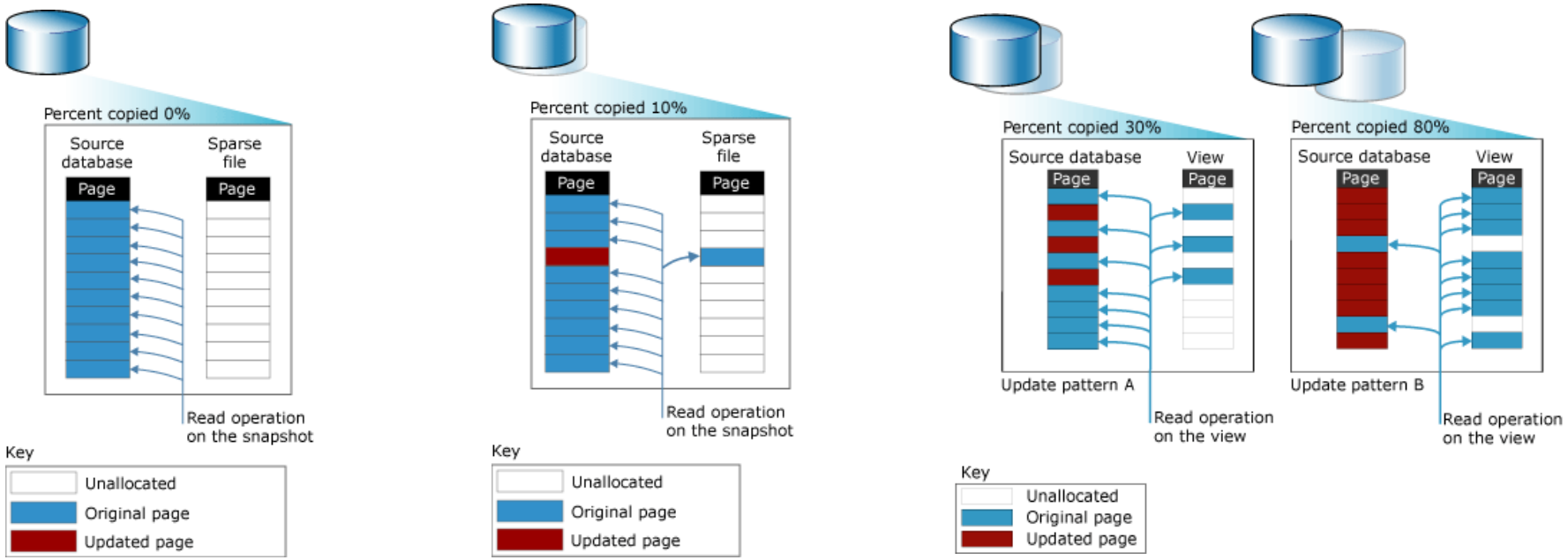
Komprese dat

- SQL Server umožňuje komprimovat uložené datové stránky
- Jsou podporovány 2 úrovně komprese:
  - **ROW** – dochází ke konverzi fixed-length datových typů na variabilní délku
  - **PAGE** – ROW + odstranění duplicitních hodnot a prefixů v datové stránce

- Použití komprese vede ke snížení zátěže na disky, ale zvyšuje zátěž na CPU
  - Největší benefit nad archivními daty, které se již moc nemění
  - Pro často aktualizovaná data se nehodí PAGE compression

Databázové snapshoty

- Databázový snapshot je nová kopie databáze, která zobrazuje původní databázi ve stavu k okamžiku vytvoření snapshotu
- Vytvoření snapshotu je okamžité a nenáročné na zdroje
- Využívá se mechanizmus copy-on-write



Zdroj: https://docs.microsoft.com/en-us/sql/relational-databases/databases/database-snapshots-sql-server

Využití databázových snapshotů

- Pro reportovací účely
  - Na přelomu účetního období je třeba zmrazit databázi
  - Reporty generované z historických dat, která již byla přepsána
- Ochrana před chybou uživatele
  - Průběžné vytváření snapshotů pro případ nutnosti obnovit poškozená data uživatelem
  - Možnost obnovit celou databázi do podoby snapshotu
- Provoz testovací databáze

Osnova

1. Funkce zaměřené na výkon
2. Funkce zaměřené na zabezpečení
   - Dynamic Data Masking
   - Row-level Security
   - Always Encrypted

Dynamic Data Masking

- SQL Server podporuje nastavení práv pro přístup k jednotlivým sloupcům v tabulce
  - Aplikace s tím musí počítat

- **Dynamic Data Masking** umožňuje neprivilegovanému uživateli zobrazit „znehodnocená data“
  - U každého sloupce, kde se funkce povoluje se určuje způsob maskování
  - Aplikace nepozná rozdíl, není ji třeba upravovat

- Privilegovaný uživatel vidí data v původní podobě

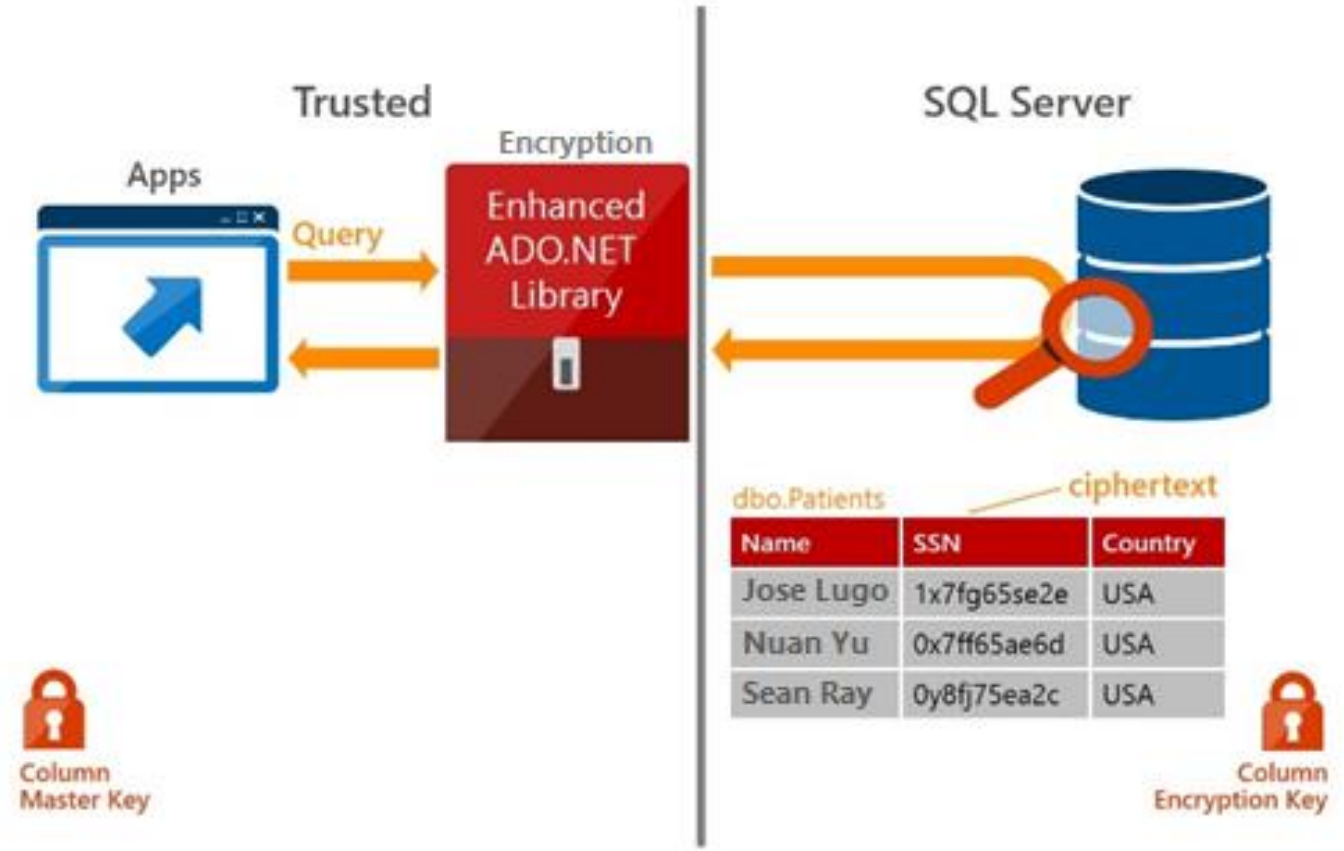
Row-level Security

- Flexibilní způsob, jak omezit přístup k datům na základě vlastní klasifikace
  - Identita uživatele, členství v rolích, denní doba…
- Funguje transparentně vzhledem k aplikaci
  - Aplikace pošle dotaz do tabulky, ale SQL Server k němu přidá navíc predikát podle nastavení Row-level Security
  - Je možné blokovat INSERT/UPDATE/DELETE operace

Always Encrypted

ENTERPRISE

- Jedná se o techniku, kdy jsou vybrané sloupce v tabulce zašifrovány, ale šifrovací klíče **nejsou uloženy na SQL Serveru, ale na klientovi**
  - K datům se nedostane ani správce SQL Serveru
- Šifrování dat provádí klient, **transparentně k aplikaci**



Zdroj: https://blogs.microsoft.com/next/2015/05/27/always-encrypted-sql-server-2016-includes-new-advances-that-keeps-data-safer/#sm.000sdug1618u5driwjv29g5ykt4sn

Dotazy

RNDr. David Gešvindr

MVP: Data Platform | MCSE: Data Platform | MCSD: Windows Store | MCT

david@wug.cz

@gesvindr