Vysoké učení technické v Brně
Fakulta informačních technologií

Databázové systémy 2018/2019

Projekt č.: 6
Název projektu: **Pekárna**

Lukáš Drahník (xdrahn00) 24. dubna 2018

1. Návrh databáze

Návrh databáze vychází z přiloženého ER diagramu. Při testování byl zjištěn jeden nedostatek a to navržené spojení objednávky s pečivem. Byla dovytvořena tabulka spojující objednávku a pečivo. Tato tabulka byla obohacena o novou položku množství. Ve výsledku je tedy v databázi pečivo označené například jako rohlík uvedeno jenom jednou a každá objednávka na něj může odkazovat s již specifickým počtem uvedeným v položce množství. Jde o stejný návrh jakým bylo vytvořeno spojení suroviny a pečiva pomocí tabulky mnozstvi.

2. EXPLAIN PLAN a vytvoření indexů

Index se přidává pomocí příkazu CREATE INDEX <name> ON <table_name>(<columns>). Co to znamená, když se přidá index? Znamená to, že když budeme například pomocí uvedeného dotazu vyhledávat pouze konkrétní sloupec, sloupce v tabulce, bude se bez indexu přistupovat pomocí TABLE ACCESS FULL, ted pomalá varianta, s použitím indexu a tedy vytvoření další tabulky se bude hledat pomocí INDEX RANGE SCAN, jelikož je to rychlejší varianta a auto-optimalizátor ji při vykonávání dotazu zvolí. Při vytváření indexu pro konkrétní dotaz nás tedy zajímá, aby byly zahrnuty všechny sloupce, které potřebujeme a pokud možno nic navíc. Pokud by nebyly zahrnuty všechny sloupce, tak je nutné poté přistoupit skrze odkaz opět do úplné tabulky.

3. Materializovaný pohled

Materializovaný pohled se vytváří a ruší stejně jako ostatní struktury pomocí CREATE MATERIALIZED VIEW AS kde poté následuje konkrétní dotaz pro konkrétní data a DROP MATERIALIZED VIEW <name>. Aby byl materializovaný pohled aktuální je nutné jeho data obnovovat pomocí REFRESH.

Účel vytváření materializovaného pohledu je v urychlení vyhledávání nad daty, která se například často nemění a samotná distribuce dat, aniž by byla dotazována originální tabulka.

Často jsou považovány Snapshot a Materialized view za rozdílné přístupy, ale není tomu tak, v případě Snapshotu jde pouze o zastaralý výraz.

Stejně jako si můžeme uložit materializovaný pohled, tak můžeme uložit pouze pohled, VIEW, tedy bez dat.

4. Transakce

Na ukázku je ve skriptu uvedena transakce s názvem ukazkova_transakce, která na začátek ukončuje všechny předcházející transakce pomocí COMMIT, nastuje název pomocí SET TRANSACTION a dále upravuje cenu pečiva, v průběhu transakce dojde ke změně ceny pečiva vícekrát a pokaždé je zde vytvořen “záchytný bod” pomocí SAVEPOINT. Pokud by transakce řádky neměnila, ale pouze četla, je vhodné uvést, že je transakce pouze READ ONLY. V případě posledního COMMIT by v případě READ ONLY nedošlo k žádné permanentní změně.

Při jakékoliv práci s řádky tabulky/tabulkou jsou použity zámky znemožňující změnu jinému uživateli při přístupu ke stejné databázi, ke stejnému řádku. Zámky se používají průběžně, ale samotná transakce se provede až po ukončení pomocí COMMIT. Průběh transakce můžeme stornovat až na začátek pomocí ROLLBACK, případně pomocí ROLLBACK TO SAVEPOINT se vrátit na konkrétní záchytný bod viz. vysvětleno výše. Pokud by jiná paralelní transakce chtěla přistoupit k jinému řádku než upravujeme, je to samozřejmě možné, na jiný řádek zde není žádný lock.

5. Závěr a postřehy z vypracování projektu

Při vytváření indexu za pomoci EXPLAIN PLAN lze za důkaz úspěšnosti mimo upravený návrh provedení dotazu považovat změření urychleného dotazu. Stopování dotazů se zapne příkazem SET TIMING ON/OFF;.

Je vhodné obzvlášť složitější procedury aj. debugovat pomocí DBMS_OUTPUT.put_line(…).

Výstup EXPLAIN PLAN je možné v sqldeveloper vidět pomocí Explain plan(F10) nebo příkazu SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY);.

Je možné si zobrazit všechny uložené materializované pohledy pomocí příkazu SELECT * FROM all_snapshots;