VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií

DATABÁZOVÉ SYSTÉMY (IDS)
2018/2019

Projekt 4. a 5. část – SQL skript pro vytvoření pokročilých objektů schématu databáze

Zadání č. 66 – Spolujízda

Filip Jeřábek (xjerab24)
Daniel Konečný (xkonec75)

Brno, 29. dubna 2019

Zadání

SQL skript, který nejprve vytvoří základní objekty schéma databáze a naplní tabulky ukázkovými daty, a poté zadefinuje či vytvoří pokročilá omezení či objekty databáze. Dále skript bude obsahovat ukázkové příkazy manipulace dat a dotazy demonstrující použití výše zmiňovaných omezení a objektů tohoto skriptu.

Implementace

Databázový server: Oracle 12c
Implementace: PL/SQL
Vývojové prostředí: JetBrains DataGrip

Databázové triggery

Pro automatické určení primárního klíče ze sekvence v tabulce `uzivatel` se využívá triggeru `uzivatel_insert`, který před každým vložením nastaví primární klíč `id_uzivatel` na následující hodnotu za posledním záznamem. Tu získáváme pomocí funkce `nextval` z tabulky `dual`.

Před vložením záznamu do tabulky `ucastni_se_jizdy` kontrolujeme pomocí triggeru `check_souradnice` správný rozsah souřadnic místa nástupu a místa výstupu cestujícího. V případě čísla mimo povolený rozsah příkaz zahlásí chybu s příslušným popisem problému.

Procedury

Je možné si nechat na výstup vytisknout seznam všech uživatelů spolu se statistikou o tom, jak dobří řidiči jsou. K tomu se využívá procedura `statistika_ridicu`, která pouze tiskne data získaná pomocí dalších volaných procedur. Při výpisu jména uživatele a následném volání procedur s identifikátorem uživatele je využíván datový typ odkazující na sloupec tabulky. Demonstračně je zavolána právě tato procedura, která se postará i o demonstraci výsledků níže uvedených procedur.

Prvně je volána procedura `zkusenost_ridice`, která podle počtu osob, které na svých jízdách řidič převezl, určí množství jeho zkušeností. Procedura prochází pomocí kurzoru všechny jízdy, které nabízel daný řidič a podle počtu uživatelů, kteří se těchto jízd účastnili, zvyšuje dočasnou proměnnou `skore`. Podle výše `skore` se poté určuje zkušenost řidiče, kterou procedura vrátí jako výstupní řetězec `VARCHAR2`.

Dále je volána procedura `hodnoceni_ridice`, která získá průměr hodnocení udělených ostatními uživateli požadovanému řidiči. Pro zpracování všech hodnocení se využívá kurzoru, který rovnou vybírá hodnocení danému řidiči (počet hvězdiček). Procedura vrátí aritmetický průměr jednotlivých hodnocení zaokrouhlený na jedno desetinné místo. Zde je pro případ, že řidiči zatím nebylo uděleno žádné hodnocení, využito zachycení výjimky `ZERO_DIVIDE`.

Explain plan a index

Explain plan je realizovaný příkazem `EXPLAIN PLAN FOR`, který je před daným příkazem `SELECT`, u kterého si přejeme zjistit jeho realizace a náročnost. Pomocí příkazu `SELECT` z `DBMS_XPLAN.DISPLAY()` vypíšeme `PLAN_TABLE_OUTPUT`, kde jsou rozepsány jednotlivé operace a jejich cena.

Příkaz `SELECT` je optimalizovaný pomocí indexů. Po první optimalizaci přidáním indexu pro tabulku `uzivatel` (jméno a identifikátor) se cena výpočtu snížila z 36 na 32. Po přidání druhého indexu pro tabulku `hodnoceni` (hodnocený uživatel) se cena výpočtu snížila na hodnotu 20.

Přehled daných operací a jejich cen:

```
Původní SELECT          Optimalizace 1                    Optimalizace 2

SELECT STATEMENT  8     SELECT STATEMENT            6     SELECT STATEMENT            4
SORT ORDER BY     8     SORT ORDER BY               6     SORT ORDER BY               4
HASH GROUP BY     8     HASH GROUP BY               6     SORT GROUP BY NOSORT        4
HASH JOIN         6     HASH JOIN                   4     NESTED LOOPS                3
TABLE ACCESS FULL 3     NESTED LOOPS                3     NESTED LOOPS                3
TABLE ACCESS FULL 3     INDEX FULL SCAN             1     INDEX FULL SCAN             1
                        INDEX RANGE SCAN            0     INDEX FULL SCAN             0
                        TABLE ACCESS BY INDEX ROWID 1     TABLE ACCESS BY INDEX ROWID 1
```

Přístupová práva

Přístupová práva se udělují příkazem `GRANT`. Pomocí `ALL` jsou udělena veškerá práva na úpravy jednotlivých tabulek. Dále jsou udělena práva na spuštění procedur (příkaz `EXECUTE`). U materializovaného pohledu jsou také udělena práva na veškeré operace.

Materializovaný pohled

Materializovaný pohled je vytvořený na tabulce `jizda` a ukazuje, kolik jízd nabídli jednotliví uživatelé. Pro fungování v reálném čase je využíván log materializovaného pohledu. Změna materializovaného pohledu probíhá `ON COMMIT`. Demonstrace je provedena pomocí výpisu dat z materializovaného pohledu, po kterém jsou vložena do tabulky, ze které materializovaný pohled čerpá, nová data a je proveden příkaz `COMMIT`. Poté je znovu vypsán obsah materializovaného pohledu, kde lze vidět změnu odpovídající nově vloženým datům.