Vysoké učení technické v Brně
Fakulta informačních technologií

Databázové systémy
2018/2019

Téma: Fotografický ateliér

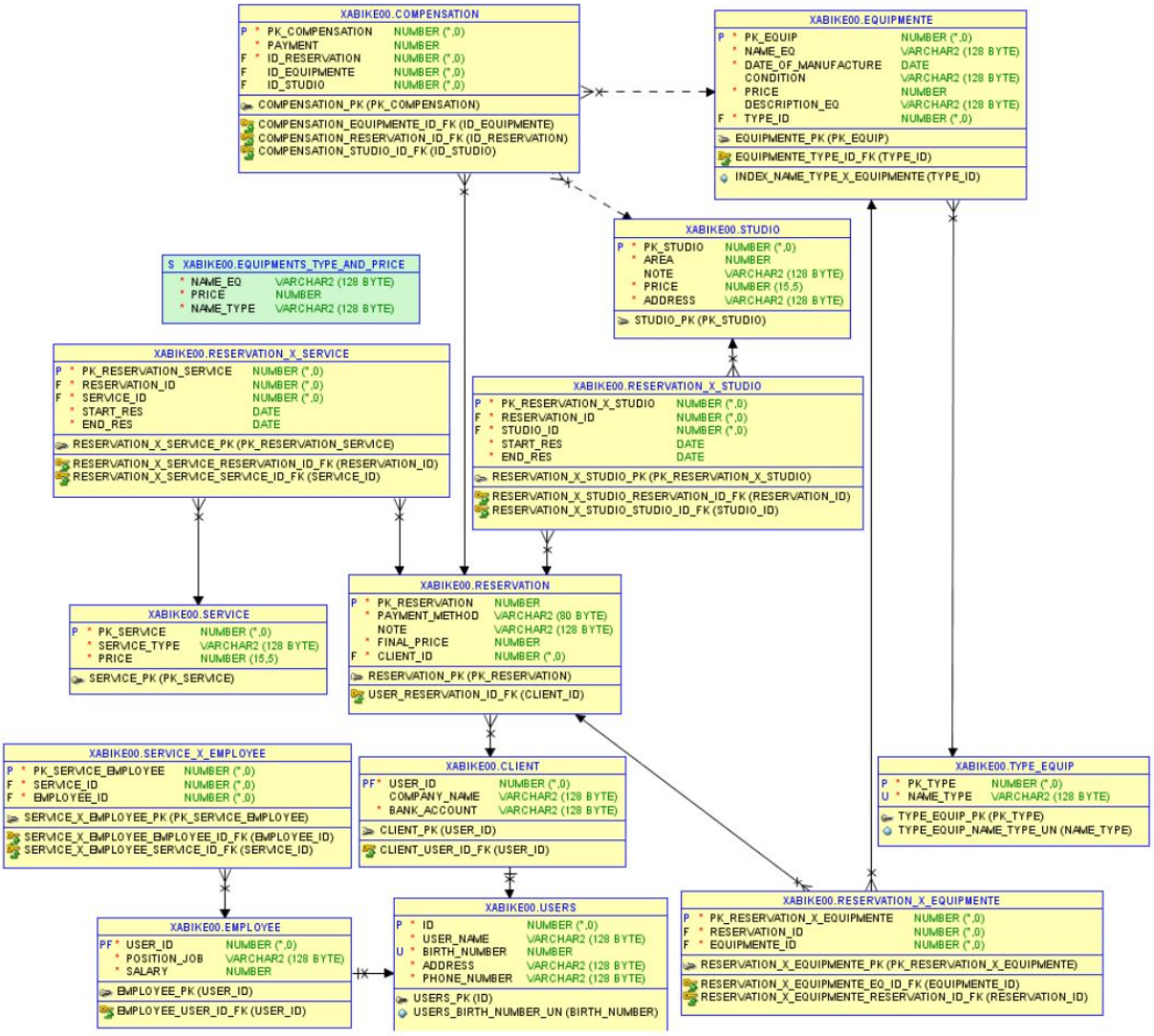
Tým: Abikenova Zhamilya (xabike00)
Kristián Glós (xglosk00)

Brno 28.4.2019

ZADANIE: 63. FOTOGRAFICKÝ ATELIÉR

Najal vás majitel řetězce foto-studií, aby jste mu vytvořili IS pro evidenci pronájmu prostor, vybavení a vyškoleného personálu. Sekretářka bude mít možnost zadávat a rušit rezervace jednotlivých klientů. IS ji následně vyčíslí cenu daného pronájmu. O klientech si bude uchovávat jméno nebo název firmy, kontaktní adresu, telefon a číslo účtu. Pronajímaný prostor (určen adresou) je členěn na ateliéry, u kterých nás zajímá rozloha a cena za hodinu pronájmu. Rezervace se vždy vztahuje na určitý ateliér a na určitou dobu. Klient si má možnost navíc domluvit dodatečné služby - přítomnost vizážistky, fotografa, osvětlovače. Služby nemusí být dohodnuté na celou dobu pronájmu ateliéru (např. vizážistka bude přítomná jen první hodinu pronájmu). U všech zaměstnanců bude uvedeno jejich jméno, funkce, adresa, kontakt a cena/hod. IS jim při každé nové rezervaci pošle upozornění ohledně najímaní si jejich služeb. Taktéž budou mít možnost si vypsat rezervace, které se jich týkají. V rámci rezervace si může klient pronajat i některé vybavení typu: počítač, fotoaparát, reflexní deštníky, reflektory, stativy... U každé položky je uveden název, popis, typ, věk, opotřebení (nový, poškozený, v opravě, vyřazený) a cena/hod. pronájmu a mohou se nacházet v různých prostorech. V případě zjištěné škody na majetku, se tato skutečnost zanese do systému k příslušné rezervaci a oznámí klientovi. Škoda se může týkat jak ateliéru, tak vybavení a je určená finanční kompenzace, která musí být uhrazena do třiceti denní lhůty. Den zaplacení škody se pak zaeviduje v systému.

DÁTOVÝ MODEL



FINÁLNY ZOZNAM TABULIEK

1. users
2. employee
3. client
4. reservation
5. service
6. service_x_employee
7. reservation_x_service
8. equipmente
9. reservation_x_equipmente
10. studio
11. reservation_x_studio
12. compensation
13. type_equip

PROCEDÚRY

Female_reservations_between(studio, from, to) – využíva parametre na výber štúdia, a časového rozhrania pre ktoré chceme zistiť výsledky. Procedúra využíva Cursor na uloženie viac ako jedného riadku, ošetrenie neočakávanej výnimky, a premenné odkazujúce sa na existujúce typy v databáze. Pri spustení so správnymi argumentmi sa vypíše cez cursor, ktorý si ukladá meno užívateľa, jeho rodné číslo, a kedy vykonal rezerváciu po riadku každá žena, ktorá si zarezervovala štúdio v danom časovom rozhraní, jej rodné číslo a taktiež počet rezervácii. Nakoniec sa vypíše celkový počet výsledkov ktorá procedúra našla. Na zistenie pohlavia využívame práve rodné číslo v riadku cursoru.

Control_phone_number – kontroluje telefónne číslo všetkých užívateľov v databáze, a pokiaľ je telefónne číslo nesprávne, vypíše ho. Taktiež tu využívame cursor a regulárny výraz pre formát českého telefonného čísla.

Výpis výsledkov procedúr je tvorený pomocou funkcie `dbms_output.put_line();`

TRIGGERS

Reservation_pk_id – Trigger slúži na vytvorenie a autoinkrementáciu primárneho kľúča rezervácií pomocou sekvencie. Spúšťa sa vždy pri vytvorení rezervácie a ID sa získa pomocou NEXTVAL.

Birth_number_of_client – Trigger na určenie validity rodného čísla pred jeho uložením INSERT do tabuľky users.

EXPLAIN PLAN

Zobrazuje postup operácií a informácie o ich čase a cene vykonania optimalizátoru Oracle pre daný SQL výraz. Bol vytvorený nad príkazom `SELECT`. Najprv je spustený bez použitia indexov, potom sme vytvorili `INDEX` na tabuľke `equipmente`, a spustili sme `EXPLAIN PLAN` znovu.

Pouzity `SELECT`:

```
SELECT type_equip.name_type,
COUNT (*) AS "Number of equipment"
FROM type_equip JOIN equipmente ON equipmente.type_id = type_equip.pk_type
GROUP BY type_equip.name_type
ORDER BY "Number of equipment" DESC;
```

Bez použitia indexov:

```
PLAN_TABLE_OUTPUT
Plan hash value: 611084591

-------------------------------------------------------------------------------------------
| Id  | Operation                     | Name       | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |            |     6 |   570 |     7  (15)| 00:00:01 |
|   1 |  SORT ORDER BY                |            |     6 |   570 |     7  (15)| 00:00:01 |
|   2 |   HASH GROUP BY               |            |     6 |   570 |     7  (15)| 00:00:01 |
|*  3 |    HASH JOIN                  |            |     6 |   570 |     6   (0)| 00:00:01 |
|   4 |     VIEW                      | VW_GBF_7   |     5 |   410 |     3   (0)| 00:00:01 |
|   5 |      TABLE ACCESS STORAGE FULL| TYPE_EQUIP |     5 |   395 |     3   (0)| 00:00:01 |
|   6 |     TABLE ACCESS STORAGE FULL | EQUIPMENTE |     6 |    78 |     3   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------
```

Potom sme vytvorili `INDEX`:

```
CREATE INDEX index_name_type_x_equipmente ON equipmente(type_ID);
```

Indexovanie je užitočné v prípade častého vyhľadávania v tabuľke, ale pri jeho použití musíme brať na úvahu fakt, že upravovanie tabuľky

bude viac operačne náročné, nakoľko sa vždy musia aktualizovať aj indexy.

Za použitia indexov:

```
PLAN_TABLE_OUTPUT
Plan hash value: 458703094

-------------------------------------------------------------------------------------------------------------------
| Id  | Operation                     | Name                          | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |                               |     6 |   570 |     4  (25)| 00:00:01 |
|   1 |  SORT ORDER BY                |                               |     6 |   570 |     4  (25)| 00:00:01 |
|   2 |   HASH GROUP BY               |                               |     6 |   570 |     4  (25)| 00:00:01 |
|   3 |    NESTED LOOPS               |                               |     6 |   570 |     3   (0)| 00:00:01 |
|   4 |     VIEW                      | VW_GBF_7                      |     5 |   410 |     3   (0)| 00:00:01 |
|   5 |      TABLE ACCESS STORAGE FULL| TYPE_EQUIP                    |     5 |   395 |     3   (0)| 00:00:01 |
|*  6 |     INDEX RANGE SCAN          | INDEX_NAME_TYPE_X_EQUIPMENTE  |     1 |    13 |     0   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------------------------------
```

Vidíme, že bez využitia `INDEX`ov, bolo použitých viac zásahov do pamäte, pri nižšej cene (využitia) procesoru, no pri vyšších počtoch riadkov by to znamenalo značné spomalenie oproti využitiu `INDEX`ov, kde sledujeme nárast využitia procesoru a zníženia počtu zásahov do pamäte a prehľadávaných riadkov.

UDELENIE PRÍSTUPOVÝCH PRÁV

Nastavené akožto administrátorovi, všetky práva pre tabuľky, materializovaného pohľadu a procedúr.

MATERIALIZOVANÝ POHĽAD

Bol vytvorený na jednoduchom príkaze `SELECT`, ktorý zobrazuje názvy a cenu príslušenstva.

```
CREATE MATERIALIZED VIEW equipments_type_and_price
CACHE -- optimalize reading from view
BUILD IMMEDIATE -- view immediately filled upon building
REFRESH ON COMMIT AS -- refreshes on commit
  -- show equipment's type and price
  SELECT XABIKE00.equipmente.name_eq, XABIKE00.equipmente.price, XABIKE00.type_equip.name_type
  FROM XABIKE00.equipmente JOIN XABIKE00.type_equip ON XABIKE00.equipmente.type_ID = XABIKE00.type_equip.pk_type
  ORDER BY XABIKE00.equipmente.name_eq;
```

Využíva `CACHE` pre optimalizáciu čítania, `BUILD IMMEDIATE` pre automatické vyplnenie dátami pri vytvorení. Dáta sa aktualizujú po volaní príkazu `COMMIT`. V skripte sa nachádza príklad využitia a aktualizácie aj zakomentovaný príklad využitia pre iných užívateľov s právami na jeho spustenie.

ZÁVER

Skript bol vypracovávaný a testovaný v SQLdelevoper na školných serveroch, alebo pomocou Oracle liveSQL v prípade núdze. Zdroje boli väčšinou čerpané z prednášok a spoločných konzultácií, ale aj online zdrojov, hlavne pri vytváraní príkazov `SELECT`. Vypracovávanie projektu nám pomohlo zoznámiť sa s vytváraním a pracovaním s databázami, a objasniť nám rôzne praktiky.