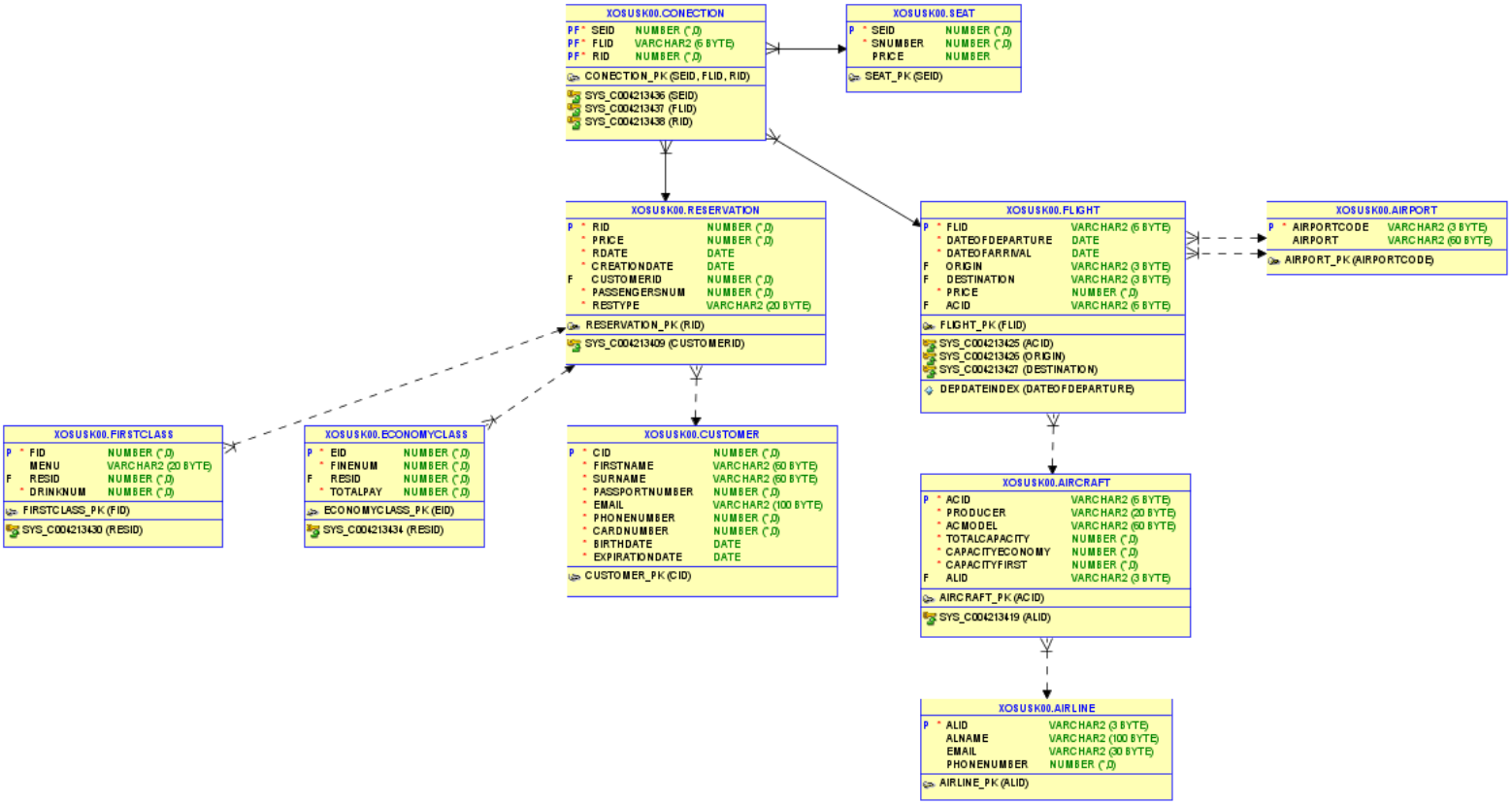
Projektová dokumentace
Projekt IDS – 4 část

Fakulta informačních technologií
Vysoké učení technické v Brně

Tým: xosusk00
Jan Osuský – xosusk00

OBSAH:

Návrhový diagram:



Úvod:

V této projektové dokumentaci popíšeme návrh a implementaci databáze pro leteckou společnost, poslední část projektu do IDS. Databáze byla navržena s ohledem na potřeby letecké společnosti, podle zadání z předmětu IUS a zahrnuje tabulky pro zákazníky, lety, letadla, sedadla, letiště a další relevantní informace. V dokumentaci popíšeme jednotlivé tabulky, včetně klíčových vlastností a omezení, a dále také implementaci tabulek s daty. V rámci projektu jsme také vytvořili několik dotazů, procedur a triggerů, které slouží k efektivnímu a bezpečnému spravování dat v databázi.

Popis řešení:

V tomto projektu jsem v individuálním plánu vytvořil návrh databáze pro leteckou společnost. Na základě požadavků zadání jsem navrhl entitně relační model, který jsem dále rozpracoval a optimalizoval. Vytvořil jsem tabulky pro ukládání dat o letadlech, letových plánech, cestujících, letenkách a dalších potřebných informacích. Každá tabulka obsahuje klíče, které umožňují propojení s ostatními tabulkami, a zároveň sloupce pro ukládání konkrétních informací.

Pro vytvoření databáze jsem použil jazyk SQL a relační databázový systém PL/SQL. Pro zajištění bezpečnosti dat jsem navrhl přístupová práva pro uživatele a zabezpečení vstupů dat pomocí validace vstupů a omezení datových typů.

Popis triggerů:

První trigger s názvem ***incrementReservationId*** slouží k automatickému inkrementování hodnoty ***rId*** před vložením nové rezervace do tabulky ***reservation***. Tento trigger se spustí před každým vložením nového řádku a používá funkci MAX k nalezení nejvyšší hodnoty ***rId*** v tabulce ***reservation*** a poté přidá 1 pro novou rezervaci.

Druhý trigger s názvem ***checkPassportNumber*** slouží k ověření, že číslo pasu zákazníka neexistuje již v databázi pro jiného zákazníka. Tento trigger se spustí před každým vložením nového řádku do tabulky ***customer***. Používá se k tomu SELECT dotaz, který kontroluje, zda již existuje zákazník s daným pasem. Pokud ano, vyvolá se chyba "Passport number already exists in the database.".

Popis procedur:

První procedura se jmenuje ***avgPassengersPerReservation*** a slouží k výpočtu průměrného počtu cestujících na rezervaci. V první části se vytváří kurzor ***cReservations***, který vrací počet cestujících na každé rezervaci v tabulce ***reservation***. Poté se pomocí smyčky ***for loop*** prochází každá rezervace v kurzoru a součet počtu cestujících se ukládá do proměnné ***vTotalPassengers***. Zároveň se počítá celkový počet rezervací v proměnné ***vReservationsCount***. Pokud nejsou nalezeny žádné rezervace, program vypíše zprávu "No reservations found." Pokud jsou nalezeny rezervace, vypočte se průměr počtu cestujících na rezervaci a vloží se do tabulky ***reservationStats*** spolu s aktuálním datem.

Druhá procedura se jmenuje ***updateFlightPrice*** a slouží k aktualizaci ceny letu. Tato procedura bere dva parametry - ID letu ***pFlightId*** a novou cenu ***pNewPrice***.

Nejprve se pomocí kurzoru ***cFlight*** vybere řádek letu, který má být aktualizován. Pokud let s daným ID není nalezen, procedura vyvolá výjimku **NO_DATA_FOUND**. Pokud je let nalezen, cena letu se aktualizuje na novou cenu. Nakonec se vypíše zpráva o aktualizaci ceny. Pokud se vyskytne jakákoliv jiná výjimka, procedura vypíše prázdnou zprávu.

Popis materialized view:

Implementace zahrnuje udělení oprávnění pro dva uživatele: xosusk00 a xosusk01. Uživatel xosusk00 získal plný přístup ke všem tabulkám, stejně jako uživatel xosusk01. Uživatel xosusk01 byl v kódu okomentován, protože jsem v týmu sám a jedná se pouze o improvizaci při řešení. Databáze žádného takového uživatele nezná a kód způsobí chybu. Obecně je materializovaný pohled databázový objekt, který ukládá výsledek dotazu a pravidelně jej aktualizuje na základě zadaného plánu. Materializovaný pohled ukládá výsledek dotazu, který kombinuje data z tabulek customer a reservation uživatele xosusk00. Je aktualizován periodicky na základě určeného plánu REFRESH COMPLETE a bude aktualizován po každé transakci COMMIT.

Popis komplexního SELECT:

Tento **SELECT** dotaz používá **WITH** klauzuli k definici poddotazu s názvem ***passengerCounts***. Tento pod dotaz vypočítává počet cestujících v každé rezervaci, kterou má zákazník. Poté se provádí hlavní dotaz, který vrací několik sloupců z tabulek ***customer*** a ***reservation***, a spojuje je pomocí klauzule **LEFT JOIN**. Sloupce zahrnují jméno, příjmení, email, datum narození, číslo karty, datum rezervace, cenu, typ rezervace a počet cestujících. Pokud zákazník nemá žádné rezervace, sloupce pro rezervaci budou prázdné a počet cestujících bude 0.

Popis INDEX a EXPLAIN PLAN:

EXPLAIN PLAN FOR spustí optimalizační engine databáze, který vytvoří plán vykonávání dotazu pro druhý příkaz. Plán obsahuje informace o tom, jakým způsobem bude dotaz proveden (například jaké indexy budou použity) a jak dlouho bude trvat jeho vykonání.

Příkaz **CREATE INDEX *depDateIndex ON flight(dateOfDeparture)*** vytváří index pro sloupec ***dateOfDeparture*** v tabulce ***flight***. Index umožňuje rychlejší vyhledávání letů podle data odletu.

Použitím indexu v dotazu vede k poklesu časové náročnosti na vyk onání dotazu. Kód se dá optimalizovat přidáním dalších indexů, ale jejich příliš časté používání může mít opačný efekt.

Použití 1 indexu ***depDateIndex:***

```
-------------------------------------------------------------------------------------------------
| Id  | Operation                      | Name           | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT               |                |     2 |   162 |     8  (13)| 00:00:01 |
|   1 |  HASH GROUP BY                 |                |     2 |   162 |     8  (13)| 00:00:01 |
|   2 |   NESTED LOOPS                 |                |     2 |   162 |     7   (0)| 00:00:01 |
|   3 |    NESTED LOOPS                |                |     2 |   162 |     7   (0)| 00:00:01 |
|   4 |     NESTED LOOPS               |                |     2 |    52 |     5   (0)| 00:00:01 |
|   5 |      TABLE ACCESS FULL         | FLIGHT         |     2 |    36 |     3   (0)| 00:00:01 |

PLAN_TABLE_OUTPUT

|-----------------------------------------
|   6 |      TABLE ACCESS BY INDEX ROWID| AIRCRAFT       |     1 |     8 |     1   (0)| 00:00:01 |
|*  7 |       INDEX UNIQUE SCAN         | SYS_C004214994 |     1 |       |     0   (0)| 00:00:01 |
|*  8 |     INDEX UNIQUE SCAN           | SYS_C004214988 |     1 |       |     0   (0)| 00:00:01 |
|   9 |    TABLE ACCESS BY INDEX ROWID  | AIRLINE        |     1 |    55 |     1   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------------
```

Přidáním druhého indexu ***flightAcidIndex*** který zlepšuje spojení tabulek letadla a letu by měla způsobit zlepšení výkonu databáze

Přidání 2. indexu

```
Plan hash value: 1515577360

-----------------------------------------------------------------------------------------------------
| Id  | Operation                      | Name           | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT               |                |     2 |   162 |     8  (13)| 00:00:01 |
|   1 |  HASH GROUP BY                 |                |     2 |   162 |     8  (13)| 00:00:01 |
|   2 |   NESTED LOOPS                 |                |     2 |   162 |     7   (0)| 00:00:01 |
|   3 |    NESTED LOOPS                |                |     2 |   162 |     7   (0)| 00:00:01 |
|   4 |     NESTED LOOPS               |                |     2 |    52 |     5   (0)| 00:00:01 |
|   5 |      TABLE ACCESS FULL         | FLIGHT         |     2 |    36 |     3   (0)| 00:00:01 |

PLAN_TABLE_OUTPUT

-------------------------------------------
|   6 |      TABLE ACCESS BY INDEX ROWID| AIRCRAFT       |     1 |     8 |     1   (0)| 00:00:01 |
|*  7 |       INDEX UNIQUE SCAN         | SYS_C004215042 |     1 |       |     0   (0)| 00:00:01 |
|*  8 |     INDEX UNIQUE SCAN           | SYS_C004215036 |     1 |       |     0   (0)| 00:00:01 |
|   9 |    TABLE ACCESS BY INDEX ROWID  | AIRLINE        |     1 |    55 |     1   (0)| 00:00:01 |
-----------------------------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------
```