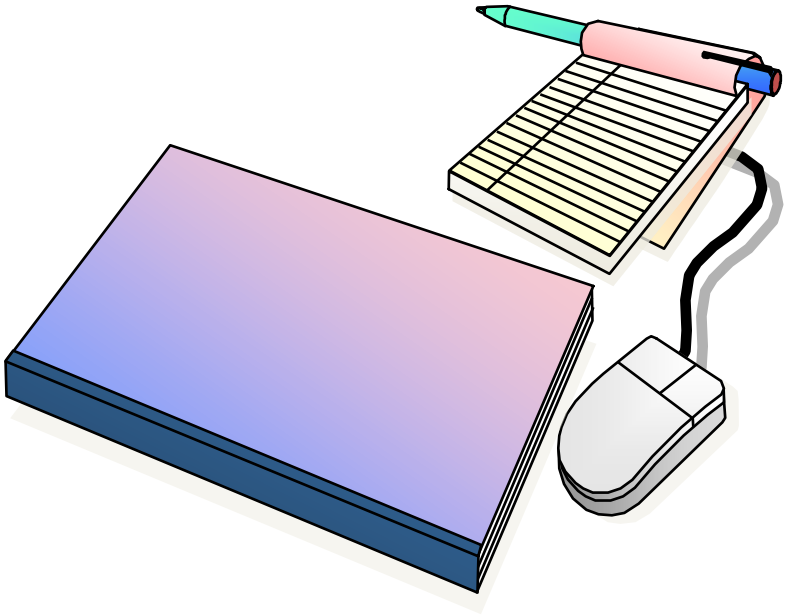
Frequent Flyer DB2



Innehållsförteckning

Innehåll

Frequent Flyer

I den här övningen kommer du att programmera ett antal COBOL-program för att skapa olika rapporter kring flera DB2-tabeller som innehåller information om personers flygningar med olika flygbolag till olika destinationer.

Uppgift 1 – Frequent Flyer DB2-tabeller

I denna uppgift så skall Du bekanta dig med de fyra DB2-tabeller som Du kommer att använda i denna övning. Används SPUFI för att göra `Select * from tabellnamn` för att kontrollera vad de olika tabellerna innehåller.

- (1) Tabellen **FLYERS** har ett utseende som du ser i nedanstående bild.

```
Kolumnnamn          Format        Not

FLYER_NUMBER        Char(10)      PK
FLIGHT_NUMBER       Char(04)      PK
FLIGHT_DATE         Date          PK yyyy-mm-dd
CLASS_OF_TRAVEL     Char(01)      T(Turist) C(Business)
                                  F(First Class)
```

- Cobol-beskrivningen FLYERS har följande utseende:

```
01  Dclflyers.
    10 Flyer-Number          Pic X(10).
    10 Flight-Number         Pic X(4).
    10 Flight-Date           Pic X(10).
    10 Class-of-Travel       Pic X(1).
```

- (2) Tabellen **FLIGHTS** har ett utseende som du ser i nedanstående bild.

```
Kolumnnamn          Format        Not

FLIGHT_NUMBER       Char(04)      PK
CITY_PAIR           Char(07)      xxx-xxx ex: DFW-ATL
AIRLINE_ID          Char(02)      01-99
```

- Cobol-beskrivningen FLIGHTS har följande utseende:

```
01  Dclflights.
    10 Flight-Number        Pic X(4).
    10 City-Pair            Pic X(7).
    10 Airline-Id           Pic X(2).
```

- (3) Tabellen **AIRLINES** har ett utseende som du ser i nedanstående bild.

Kolumnnamn	**Format**	**Not**
AIRLINE_ID	Char(02)	PK 01-9
AIRLINE_NAME	Char(25)	ex: American Airlines

- Cobol-beskrivningen AIRLINES har följande utseende:

```
01  Dclairlines.
    10 Airline-Id           Pic X(2).
    10 Airline-Name         Pic X(25).
```

- (4) Tabellen **MILEAGES** har ett utseende som du ser i nedanstående bild.

Kolumnnamn	**Format**	**Not**
CITY_PAIR	Char(07)	xxx-xxx ex: DFW-ATL
MILEAGE	Dec(5)	

- Cobol-beskrivningen AIRLINES har följande utseende:

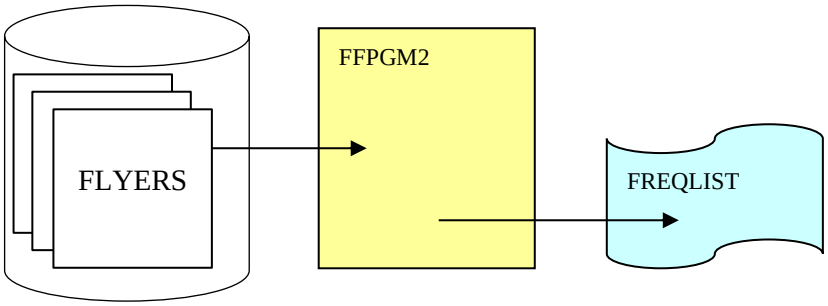
```
01  Dclmileages.
    10 City-Pair            Pic X(7).
    10 Mileage              Pic S9(5)Packed Decimal.
```

- COBL-beskrivningarna finns i biblioteket *lärar-userid*.FREQ.DCLGEN.

Uppgift 2 – Frequent Flyer Program

I denna uppgift kommer du att skriva ett program som skapar en rapport från information i de olika DB2-tabellerna som tillhör Frequent Flyer.

- Skriv ett program, `FFPGM2`, som läser samtliga rader i tabellen FLYERS och skriver en rapport liknande den Du ser nedan:



- Rapporten skall ha följande utseende:

```
*** Frequent Flyer Report ***

 Date        Al#  Fl#   Cl   C-Pair   Mileage
 22-01-07    01   0222  Y    DFW-ATL      731
 22-01-10    02   0322  Y    LAS-DFW    1 056
 22-01-11    07   0351  Y    BNA-DFW      631
 . . . .
 . . .
 . .
 .
 22-02-23    04   0633  C    SXM-DFW    2 339
 22-02-24    05   0730  Y    RDU-DCA      227
 22-03-10    07   0049  Y    DFW-LAX    1 235
 22-03-16    12   0495  C    SJU-SXM      192
 . . .
 . .
           .
 Total Mileage                        nn nnn

 Page 1
```

- Kompilera och korrigera tills du har en felfri kompilering.
- Exekvera programmet och kontrollera resultatet.

Uppgiften är avslutad.

Uppgift 3 – Bonus Report

- Komplettera ditt program, eller skapa ett nytt med namnet `FFPGM3` så att rapporten kommer att ha följande utseende:

```
*** Frequent Flyer Report ***

 Date        Al#  Fl#   Cl   C-Pair   Mileage Inc Bonus

 22-01-07    01   0222  Y    DFW-ATL      731       731
 22-01-10    03   1609  Y    ATL-BNA      214       500
 22-01-11    07   0351  Y    BNA-DFW      631       631
 . . .
 . . .
 . .
 .
 22-03-03    02   0769  C    SXM-DFW      227       500
 22-04-03    01   1263  Y    RDU-DFW    1 061     1 061
 22-04-10    07   0049  Y    DFW-LAX    1 235     1 235
 22-05-15    11   0428  C    LAX-DFW    1 235     1 544
 . . .
 . .
 .
 Total Mileage                        nn nnn    bb bbb

 Page 1

End of Report
```

- Som Du ser så har det tillkommit en kolumn, *Inc Bonus*. För varje resa så skall en extra bonus tillkomma enligt följande regler:
  - Samtliga klasser skall alltid få minst 500 miles när reslängden är lägre än 500, men ingen ytterligare bonus skall då tillkomma.
  - Turistklass, ’Y’, skall inte ha någon extra bonus.
  - Business Class, ’C’ skall erhålla 125% på aktuell reslängd.
  - First Class, ’F’ skall erhålla 150% på aktuell reslängd.

- Kompilera och korrigera tills du har en felfri kompilering.
- Exekvera programmet och kontrollera resultatet.

Uppgiften är avslutad.

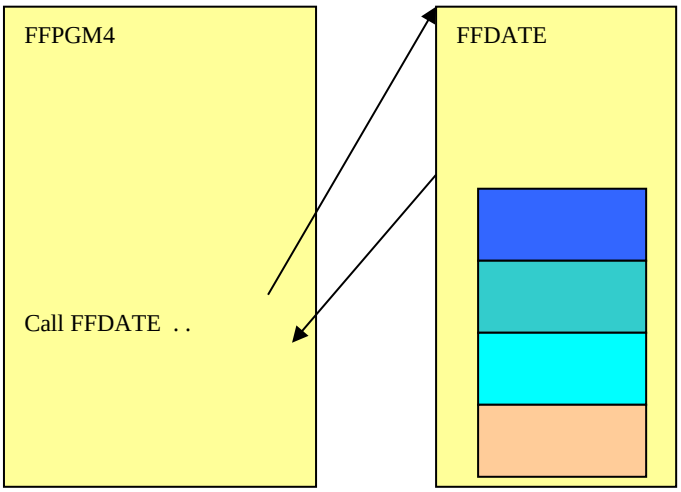
Uppgift 4 – Date-and-Day Program

I den här uppgiften skall du skriva ett program som skall kunna skapa datum i olika format. Programmet skall kunna anropas med en parameter som beskriver i vilket format datumet skall levereras.

- Skriv ett program, `FFDATE`, som skall kunna anropas som ett subprogram, där du i anropet skall kunna ange att det returnerade datumet skall vara i något av nedanstående format.

```
DATE=S    ( 6/4, 20xx )
DATE=M    ( 6 April, 20xx )
DATE=L    ( Måndag 6 April, 20xx, kl hh:mm )
```

- Programmet skall konstrueras så att det består av ett ”huvudprogram” som analyserar att anropsparametern är korrekt. Sedan skall de olika datumformaten konstrueras av ”inbakade” program i samma laddmodul/källkod.



- Lämpligen skapar du ett huvudprogram som innehåller enskilda ’inbakade’ program för att skapa de olika datumformaten.
- Bestäm själv formatet för interna anropsparametrar etc. Datum som programmet skall returnera är dagens datum samt ett framtida datum om 30 dagar.
- Komplettera sedan huvudprogrammet FFPGM4, så att det kan anropas med en parameter vid exekveringen (`DATE=S/DATE=M/DATE=L`). Programmet skall även skriva ut ett meddelande om exekveringsparameter utelämnats eller är felaktig och avslutas med lämplig returkod.
- Uppdatera programmet, så att rapporten får ett utseende som liknar denna:

```
Frequent Flyer Report

Printed        :     Monday, 6 April, 20xx
Valid Until    :     Wednesday, 6 May, 20xx

Date       AL#   Fl#   Cl C-Pair   Mileage   Inc Bonus

22-01-07   01    222   Y  DFW-ATL      731        731
22-01-10   02    322   Y  LAS-DFW     1056       1056
22-01-11   07    351   Y  BNA-DFW      631        631
22-01-15   11    428   C  LAX-DFW     1235       1543
22-01-16   05    459   F  TPA-SJU     1237       1855
. . . .
. . .
. .
.
22-03-24   06  1088   Y  DFW-RDU     1061       1061

Total Mileage                      13430      17129
```

- Kompilera och korrigera tills du har en felfri kompilering.
- Exekvera programmet och kontrollera resultatet.

Uppgiften är avslutad.

Uppgift 5 – Airline Name

- Skapa en kopia av programmet `FFPGM4`. Det nya namnet skall vara `FFPGM5`. Komplettera programmet så att det i rapporten skriver ut namnet på flygbolaget i stället för *Airline-Id*.
- Tabellen AIRLINES innehåller namnen på flygbolagen.

- Kompilera och korrigera tills du har en felfri kompilering.
- Exekvera programmet och kontrollera resultatet.

Uppgiften är avslutad.

Uppgift 6 – Date Search Summary Report

- Skriv ett program, `FFPGM6` som kan skriva ut en rapport över vilka resor som gjorts under en viss tidsperiod.
- Läs in fr.o.m. datum – t.o.m. datum från en datumfil. Bestäm själv namn på filen.
- Formatet skall vara yyyy-mm-dd +yyyy-mm-dd.
- Programmet skall kontrollera att formatet är korrekt.
- Om formatet är felaktigt, så skall programmet avsluta med returkod 99.
- Om formatet är korrekt, men sökvärdena är felaktiga, så skall programmet avslutas med returkod 98.
- Skulle filen saknas, eller vara tom, så skall detta accepteras, och då skall samtliga poster skrivas ut i rapporten.
- Rapporten skall visa vilka flygturer som gjorts under en viss period med tidpunkt enligt angivna sökdatum.
- Rapportens utseende skall ha likheter med den tidigare rapporten.

- Kompilera och korrigera tills du har en felfri kompilering.
- Exekvera programmet och kontrollera resultatet.

Uppgiften är avslutad.

Uppgift 7 – Destinations Report

- Skriv ett program, `FFPGM7` som kan skriva ut en rapport över vilka resor som gjorts med ett visst City-Pair.
- Läs in `City-Pair` via `SYSIN`. Flera `City-Pair` skall kunna anges som olika poster/rader.
- Programmet skall även kunna hantera situationen när endast avrese- eller destinations-ort anges, så att man kan se vilka resor som gjorts från eller till en viss stad.

- Kompilera och korrigera tills du har en felfri kompilering.
- Exekvera programmet och kontrollera resultatet.

Uppgiften är avslutad.

Uppgift 8 – Day-of-Week-Name

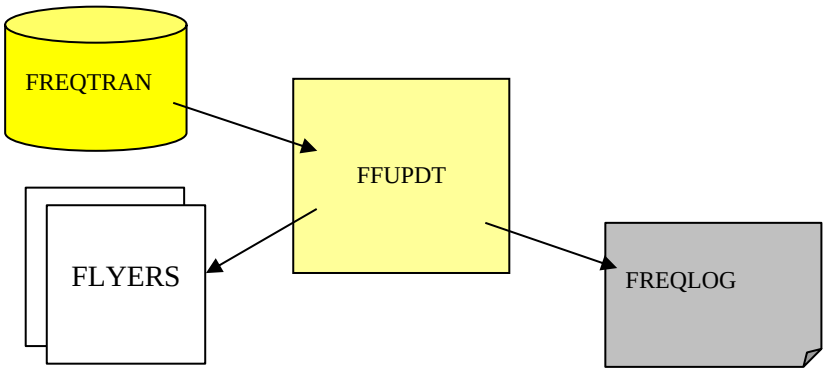
- Komplettera rapporten så att den innehåller dagnamn när resan gjorts.
- Du kan använda programmet `FFPGM5` som mall. Ge programmet namnet `FFPG8`.
- Du kommer att behöva komplettera datum-programmet, `FFDATE`, så att det kan leverera ett dagnamn med ledning av datum. Ge programmet namnet `FFDATEX`.

- Kompilera och korrigera tills du har en felfri kompilering.
- Exekvera programmet och kontrollera resultatet.

Uppgiften är avslutad.

Uppgift 9 – File Update Program

I denna uppgift så kommer du att skriva ett program som läser in en transaktionsfil med information om förändringar som skall göras i resetabellerna. Du behöver skriva ett program (FFBACK) som läser samtliga rader i FLYERS-tabellen och sparar dessa som en backupkopia i en ny generation i ett GDG.



- Skriv ett program, `FFUPDT`, som läser in filen `FREQTRAN`, *`userid`*`.FREQ.FREQTRAN`. Denna skall innehålla poster med transaktioner som skall appliceras på den databastabellerna.
- Bestäm själv vilka transaktioner som skall användas. Skapa några som är korrekta, några som är felaktiga för att kunna testa avvikelser.
- Transaktionerna har följande utseende:

```
Pos 1     : A (Add)
Pos 2-80  : Flight Date etc, en ny flygresa

Pos 1     : R (Remove)
Pos 2-6   : Flight Date

Pos 1     : U (Update)
Pos 2-6   : Flight Date
Pos 7     : Flight Class
```

- Programmet skall kontrollera att datumet i transaktionerna är korrekt. (se:LE Call CEEDAYS alt COBOL Intrinsic Function(s)).
- Programmet skall även skriva en loggfil, `FREQLOG`, med information om gjorda transaktioner.
- Felaktiga transaktioner skall även skrivas till loggfilen, så att dessa kan korrigeras/följas upp.
- När ett fel upptäcks vid applicering av transaktioner, så skall programmet abendas. Kontrollera vilken LE Callable Serveice som kan användas.

- Kompilera och korrigera tills du har en felfri kompilering.
- Exekvera programmet och kontrollera resultatet.

- Kontrollera resultatet och jämför med en tidigare rapport samt transaktionerna i `FREQTRAN`.

- Uppgiften är avslutad.