

EasyAnalyx - Produktbeschreibung

EasyAnalyx ist ein Analyse-Werkzeug für die gezielte quantitative und qualitative Untersuchung beliebiger Datenbestände. Die Prüfvorschriften und die ermittelten Resultate werden dem Benutzer als Report, Tabelle (DBF) und/oder Excel-Sheet zur Verfügung gestellt und stets in der internen Resultat-Tabelle gespeichert. Das heisst, dass gleichzeitig mit der Prüfung alle relevanten Angaben dazu archiviert werden.

Spätere Wiederholungen derselben Abfragen resp. Modifikationen davon sind also jederzeit gewährleistet. Es besteht auch die Möglichkeit, Resultate verschiedener Perioden elektronisch miteinander zu vergleichen.

EasyAnalyx wurde in Visual FoxPro – der schnellsten Datenbanksprache der Welt – entwickelt und zeitigt auch entsprechende Leistungsmerkmale: Pro Million Input-Datensätze muss mit einer Verarbeitungszeit von 15 Sekunden gerechnet werden, d.h. wenn eine Datei mit 150'000 Datensätzen nach 15 verschiedenen Kriterien geprüft werden soll, dauert dies nur rund 25 Sekunden !

Anwendungsbereich

EasyAnalyx das Analyse-Werkzeug für

- Den Manager: **Speed Research**
- Das Marketing: **Info-Analyzer**
- Den Revisor: **Case Finder**
- Das Migrationsteam: **Quality Checker**
- Den Entwickler: **Query-Generator**

Was kann geprüft werden ?

EasyAnalyx kann alle X-Base-Tabellen und alle zugänglichen Views prüfen. Die zu prüfenden Daten werden zuerst importiert und für die Analyse optimiert; zur Laufzeit werden je nach Bedarf fixe oder temporäre Indizes angelegt. Zudem können CSV-Daten importiert und analog Tabellen/Views verarbeitet werden.

Gleichartige Datenbestände verschiedener Perioden – z.B. ein monatlich erstelltes Umsatzfile – können jeweils mit den gleichen Prüfsätzen bearbeitet werden, was ein massives Einsparungspotenzial bedeutet.

Wie kann geprüft werden ?

EasyAnalyx unterstützt folgende Prüfkategorien:

- Präsenzprüfungen: **Obligatorisch, Fakultativ, Verboten**
- Alfa- und Zeichenfelder: **Reine Buchstaben, Lower-String, Upper-String**
- Ziffernfelder: **Reine Ziffern, Positive Zahlen, Negative Zahlen**

- Inhaltsprüfungen: **Muss-Inhalt resp. Maske, Inlist, Schalter**
- Bereichsprüfungen: **Charakter-Bereich**
Datum-Bereich: Char-Felder werden umgewandelt
Numerisch: Char-Felder werden umgewandelt

- Bedingungen: Das Feld muss eine **Bedingung** in Bezug auf ein oder mehrere Fremdfelder erfüllen

- Quertest-1: Das Feld muss in Bezug auf ein oder mehrere Fremdfelder **exklusiv gefüllt** sein resp. **mindestens ein Feld** einer Feldgruppe muss **Inhalt** haben
- Quertest-2: Ein Feld muss in Bezug auf exakt ein anderes Feld eine **logische Inhaltsbedingung** (.NULL., Leer resp. Gefüllt) erfüllen

- Key-Prüfungen: **Sekundärkey:** Existiert er als Primary in einer zu definierenden anderen Tabelle (Index Bedingung)
Reihenfolge: Ist das Feld auf- resp. absteigend mit resp. ohne Duplikate
UNIQUE: Ist das Feld eindeutig (Sort nicht nötig)
DUPLIKAT: Kommt jeder Schlüssel **exakt zweimal** vor im ganzen File (Sort nicht nötig)
TRIPLIKAT usw. bis 9: Analog DUPLIKAT

- Informationen aller Art: **Kontrollzahlen:** Wichtigste Kennwerte zu den Typen Charakter, Numerisch und Datum ermitteln
Gruppieren: Auszählungen nach Inhalt absolut, nach Häufigkeit, nach Wertebändern fix oder variabel und mengenabhängig (total 7 Varianten)
Hierarchie: Auswertung von expliziten und impliziten Substrings eines Feldes
Kreuztabelle: Zweidimensionale Auswertung
Statistikfile: Verdichtung auf wählbarer Ebene
Längenanalyse: Signifikante Belegung in Charakter-Feldern inkl. Berücksichtigung „interner“ Leerstellen

- Spezial-Funktionen: **Externes Modul:** Aufruf eines Prüfprogramms resp. Einsatz von EXECSCRIPT (in Planung)
Testfile-Generator: Wahl der Felder, Verschlüsseln von Feld-Inhalten, Wahl der Datenqualität
Filter: Setzen von permanenten Vor-Filtern resp. temporären Filterbedingungen inkl. Invers-Filtern
Vorkalkulation: Setzen von fixen Vor-Berechnungen

Welche Techniken setzt EasyAnalyx ein ?

Beispiele

1. Feldinhalte auf Zeichen-Sets prüfen

Verschiedene Prüfungen (Alfa, Lower, Upper) untersuchen, ob sich in einem Datenfeld nur erlaubte Zeichen befinden. Wie können solche aufwändigen Prüfungen überhaupt und dann erst noch effizient durchgeführt werden ?

a) Die Prüfung an sich:

- Es macht offensichtlich keinen Sinn, jedes Byte auf seine Präsenz in einem erlaubten String zu untersuchen mit z.B. dem Befehlssatz

```
FOR n = 1 TO LEN("prueffeld")
  IF NOT SUBSTR(prueffeld,n,1) $ "0123456789"
    lnFehler = lnFehler + 1
  EXIT
ENDIF
ENDFOR
```

- Besser ist es, eine geeignete FoxPro-Funktion einzusetzen, z.B. für eine Prüfung auf ausschliesslich Ziffern (ohne Blanks):

```
IF !EMPTY(CHRTRAN(prueffeld,"0123456789",""))
  lnFehler = lnFehler + 1
ENDIF
```

- Doch auch diese Abfrage verschlingt – obschon die VFP-Funktion um einiges schneller abläuft als der obige „handgestrickte“ Befehlssatz – für eine Million Datensätze viel Zeit, und wir sind ja alle sehr ungeduldig ...

b) Optimierung:

- Je mehr Datensätze eine Tabelle enthält, desto grösser ist die Wahrscheinlichkeit, dass dieselben Inhalte mehrfach (dutzendfach oder gar tausendfach) vorkommen

- Nützen wir es also aus:

```
SELECT prueffeld, COUNT(*) AS anzahl
FROM inputdatei GROUP BY 1 INTO CURSOR summary
SCAN
  IF !EMPTY(CHRTRAN(prueffeld,"0123456789",""))
    lnFehler = lnFehler + anzahl
  ENDIF
ENDSCAN
```

- Wenn nun noch die Fehler markiert werden sollen, brauchen wir unter die Zeile mit dem Fehlerzähler nur noch zu schreiben:

```
REPLACE inputdatei.errorlog WITH <errorcode> ;
FOR inputdatei.prueffeld = summary.prueffeld
```

2. Filter-Handling

Individuelle Filter auf Feldebene sind trivial, der Filterausdruck ist ein üblicher FoxPro-Filterausdruck, z.B.

```
!EMPTY (Prueffeld)  
Prueffeld >= anderesFeld  
BETWEEN (Prueffeld, DATE () - 3, DATE () - 3 * 365)
```

EasyAnalyx bietet jedoch mehr, nämlich vordefinierte **Filter-Komponenten**, die sowohl individuell als auch fix auf Feldebene eingesetzt werden können.

Was sind Filter-Komponenten ?

Das sind ebenfalls logische Ausdrücke – wie obige Beispiele – die sich aber beliebig verbinden lassen. Filterkomponenten werden mit Unterstrich gefolgt von Buchstabe A bis Z bezeichnet, z.B.

```
_A ("feldx >= feldy")  
_B ("feldz $ felda")  
_C ("NVL (felddb, DATE [9999-12-31] ")
```

Filterkomponenten können auch andere Filterkomponenten enthalten, z.B.

```
_D ("&_A AND !EMPTY [feldd] ")  
_E ("&_B AND [&_D OR &_C] ")
```

Die Ausführung solcher Ausdrücke gelingt jedoch nur, wenn sie vorher in Ausdrücke mit maximal einer &-Stufe umgewandelt werden, dies geschieht durch einen iterativen Prozess, der am Beispiel der Filterkomponente **_E** erläutert wird:

a) Original-Ausdruck:

```
_E ("&_B AND [&_D OR &_C] ")
```

b) Erste Umwandlung:

```
_E (" [feldz $ felda] AND [ [&_A AND !EMPTY [feldd] ]  
OR [ NVL (felddb, DATE [9999-12-31] ] ] ")
```

c) Zweite Umwandlung:

```
_E (" [feldz $ felda] AND [ [ [feldx >= feldy] AND  
!EMPTY [feldd] ] OR [ NVL (felddb, DATE [9999-12-31] ] ] ")
```

Wenn nun dieser Filter mit einem expliziten Ausdruck auf Feldebene – der wiederum ein symbolisches Feld &feldname enthalten kann – verbunden wird, bleibt er für FoxPro immer noch ausführbar.

Filter-Komponenten sind an allen Orten einsetzbar, wo logische Ausdrücke erlaubt sind, also z.B. in anderen Filtern, in Prüfkategorie B (Berechnung), in Prüfkategorie E (Externes Modul resp. EXECSCRIPT) und in allen IIF-Ausdrücken.

Das aktuelle Feld wird übrigens immer durch **&_0** repräsentiert, d.h. dass geeignete Filter dynamisch mit dem aktuell zu prüfenden Datenfeld kombiniert werden.

Kontaktadressen

Maurhofer-Informatik AG, Fritz Maurhofer, Hinwil
043 / 843 04 44

eMail: murmi@maurhofer-informatik.ch

I-2000 AG, Jürg Hertli, 9524 Zuzwil-SG

eMail: jhertli@leunet.ch

Supportadresse: Je nach Kunde und Anwendungsbereich

04.07.2005/he/fm
EAProduktebeschrieb.doc

EASY ANALYX