# FEDAS®.Fakturengenerator

## Finanz Management System (fms)

## Handbuch V1.1

# Inhaltsverzeichnis


1. Einleitung....4
   1.1 Zusammenfassung Verfahrensbeschreibung....5
2. Fakturengenerator - Einsatz im betrieblichen Umfeld....6
3. Fakturengenerator Datenmodell....7
   3.1 Datenmodell (Klienten, Aufträge, Rechnungen, Prüfregeln, Logging)....8
   3.2 Datenbank-Objekte (Klienten, Aufträge, Rechnungen, Prüfregeln, Logging)....10
   3.3 Datenmodell, Datenbank-Objekte (Metadaten, Einsatzmittel)....14
   3.4 Datenmodell Rechnungen und Datenpüfung....14
4. Fakturengenerator zur Laufzeit....15
   4.1 Fakturengenerator Programmübersetzung....15
   4.2 Übersetzungsfehler Fakturierungsprogramm....16
   4.3 Ablaufschema....16
5. Fakturierung Dialogbedienung....19
   5.1 Dialog Übersicht....19
      5.1.1. Dialog Massenfakturen....19
      5.1.2 Dialog Massenfakturen Bedienung....20
      5.2.1. Dialog Einzelfakturen....22
      5.2.1. Dialog Einzelfakturen Bedienung....22
6. Rechnungsnummern - Vergabe....23
   6.1 Fakturierungsnummer....23
   6.2 Rechnungsnummer (Buchungskreiszähler, Belegnummer)....23
   6.3 Sammelrechnungsnummer....23
   6.4 Blattnummer....24
7. Funktion Write....25
   7.1 Write Datenmodell....25
   7.2 Write Funktionsbeschreibung....29
   7.3 Write Laufzeitverhalten....30
      7.3.1 Write Anwendungskontext....30
      7.3.2 Write und Blattnummer (Sammelrechnung)....30
      7.3.3 Write und Auftragsdaten....31
   7.4 Fakturierung von Guppenveranstaltungen....32
8. Fakturierungsprogramm Aufbau (FEDAS.Basic Programm)....32
9. Funktionsbeschreibung (Referenz)....37
   AddString....37
   Avg....37
   Date$....37
   DateAdd....38
   DateDiff....38
   DateOf....39
   Day....39
   Filter....39
   FStr$....40
   Get....40
   Init....41
   Input....43
   IsValid....43
   Max....43
   Min....44
   MinutesDiff....44
   Next....44
   Now:DATE....45

Print ..... 45
Searchdetails ..... 46
Sum ..... 46
Time ..... 46
TimeDiff ..... 46
Weekday ..... 47
Write ..... 47
10. Datenprüfung (Prüfregeln, Logging) ..... 48
10.1 Datenqualität Auftragsdaten ..... 48
10.2 Datenmodell, Datenbank-Objekte (Prüfregeln) ..... 49
10.3 Protokoll automatische Datenprüfung ..... 50
10.4 Datenmodell, Datenbank-Objekte (Logging-Protokolldaten) ..... 51
10.4.1 Logging-Initialisierung ..... 51
10.4.2 Logging-Datenbankfilter (Get, Filter) ..... 51
10.4.3 Logging-Trace ..... 51
10.5 Eingabe Prüfregeln ..... 52
11. Dialoge Systemdaten Designer ..... 55
11.1 Systemdatenauftrag ..... 55
11.2 Einsatzmittel (Ressourcen) ..... 60
11.3 Konfigurationsbeispiel (Hausnotruf) ..... 60
12. Fakturierungsprogramm Beispielübersicht ..... 67
12.1 Empfehlung zur Erstellung von Fakturierungsprogrammen ..... 68
12.2 Gemeinsame Datenbank-Objekte (Fakturierungsprogramm Beispiele) ..... 69
13. Beispiel1: Automatikfakturen im Hausnotruf ..... 70
13.1 Allgemeine Auftragsbeschreibung ..... 70
13.2 Programm Auftragsdaten Dialogverarbeitung ..... 71
13.3 Programm Datenbank-Objekte Hausnotruf ..... 74
13.4 Programmdaten Beschreibung Hausnotruf ..... 75
13.5 Programmbeschreibung Hausnotruf ..... 77
13.6 FEDAS.Basic Programmcode Hausnotruf (Fakturierungsprogramm) ..... 80
14. Beispiel2: Automatikfakturen im Krankentransport ..... 86
14.1 Allgemeine Auftragsbeschreibung ..... 86
14.2 Programm Auftragsdaten Dialogverarbeitung ..... 87
14.3 Programm Datenbank-Objekte Krankentransport ..... 89
14.4 Programmdaten Beschreibung Krankentransport ..... 90
14.5 Programmbeschreibung Krankentransport ..... 91
14.6 FEDAS.Basic Programmcode Krankentransport (Fakturierungsprogramm) ..... 94
15. Beispiel3: Automatikfakturen in der Pflege ..... 101
15.1 Allgemeine Auftragsbeschreibung ..... 101
15.2 Programm Auftragsdaten Dialogverarbeitung ..... 102
15.3 Programm Datenbank-Objekte Pflege ..... 103
15.4 Programmdaten Beschreibung Pflege ..... 104
15.5 Programmbeschreibung Pflege ..... 107
15.6 FEDAS.Basic Programmcode Pflege (Fakturierungsprogramm) ..... 114

Alle Rechte, insbesondere das Recht der Vervielfältigung und Verbreitung sowie der Übersetzung, vorbehalten. Kein Teil des Werkes darf in irgendeiner Form (durch Fotokopie, Mikrofilm oder ein anderes Verfahren) ohne schriftliche Genehmigung des Herstellers reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.

# 1. Einleitung

Dieses Handbuch richtet sich an jene Personen, die sich über technische Details unserer Software informieren und mit unserem System Programme zur Neuorganisation in der Betriebsführung erstellen möchten [1)].

Der Fakturengenerator ist Baustein unseres „**Finanz Management Systems**“, ein Werkzeug zur Erstellung von Anwendungssystemen mit Hilfe von Datenbank und Software. Anwendungssysteme zur Betriebsführung werden von uns branchenspezifisch hergestellt.
Unternehmen führen Aufträge aus. Dabei entstehen Daten, wie beispielsweise gefahrene Kilometer, Zeitaufwand, gelieferte Stückzahlen, durchgeführte Therapien, usw. Diese Daten sind als Auftragsdaten zusammen mit den Vereinbarungen (Lieferkonditionen) die Grundlage zur Erstellung einer Rechnung. Von der Erfassung der Daten zur Erstellung einer Rechnung sind also im Unternehmen eine Reihe administrativer Vorgänge erforderlich. Die Kette ***Auftragsdaten-Fakturierung-Buchhaltung*** ist Grundstruktur unserer Anwendungssysteme. Der Fakturengenerator eignet sich gut für betriebliche Abläufe, bei denen Rechnungen (in großer Anzahl) nach flexiblen Vereinbarungen (Regeln) zu erstellen sind oder, ganz allgemein, für eine Neuorganisation der Abläufe im Unternehmen. Die basierend auf Auftragsdaten erstellten Rechnungen stehen dann automatisch auch in unserer Buchhaltung (Debitoren- Kreditorenbuchhaltung) zur Verfügung.

Bausteine (Komponenten) Finanz Management System (fms):

1. FEDAS.DebitorKreditor (Debitoren, Kreditoren, Zahlungsverkehr) als Programm, welches alle Funktionen zur Verarbeitung von Eingangs- und Ausgangsrechnungen sowie des Zahlungsverkehrs realisiert (Debitoren- Kreditorenbuchhaltung).
2. Ein Fakturengenerator als Komponente, welche mit Hilfe von flexiblen Regeln laufend Daten der Auftragsabwicklung automatisch in Debitorenfakturen (Automatikfakturen) umwandelt. Zur Formulierung der Regeln steht eine spezialisierte Programmiersprache (FEDAS.Basic) zur Verfügung.

[1)] ist in Kooperation möglich.

## 1.1 Zusammenfassung Verfahrensbeschreibung

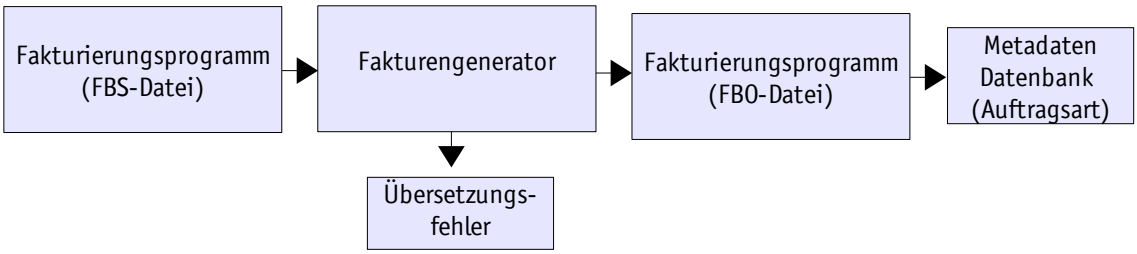
Unsere Software verfolgt ein generalisiertes Konzept, indem sich die Fakturierung beliebiger Rechnungen standardisieren lässt. Damit wird der Aufwand in der Administration reduziert und hochwertige Daten in der Datenbank erzeugt. Das System verfolgt ein hohes Maß an Integration. Die Kapitel 2 bis 15 informieren Sie über unsere Konzeption.

- Kapitel 2 „**Fakturengenerator – im betrieblichen Umfeld**“ liefert einen Überblick über verschiedene Programmkomponenten, die wir zur Verfügung stellen können und wie diese in der betrieblichen Umgebung zusammenwirken.
- Kapitel 3 „**Fakturengenerator Datenmodell**“ beschreibt die vom Fakturengenerator benötigten Daten der Datenbank.
- Kapitel 4 „**Fakturengenerator zur Laufzeit**“ vermittelt ein Verständnis über die interne Arbeitsweise des Systems.
- Kapitel 5 „**Fakturengenerator Dialogbedienung**“ dokumentiert den Dialog zur Bedienung des Systems. Dieser Dialog ist, wenn er in eine Klienten-Auftragsverwaltung eingebaut ist, eine Realisierung unseres Finanz Management Systems für ein Anwendungssystem.
- Kapitel 6 „**Rechnungsnummern – Vergabe**“ legt das Verhalten des Fakturengenerators fest, das für die Vergabe von Rechnungsnummern und Sammelrechnungsnummern maßgeblich ist. Dieses Konzept ist für die Gestaltung der Automatikfakturen wichtig, indem durch ein Fakturierungsprogramm festgelegt werden kann, welche (Teil-)Rechnungen eines Klienten für einen Zeitabschnitt als Sammelrechnung und welche als Einzelrechnung ausgegeben werden sollen.
- Kapitel 7 „**Funktion Write**“ definiert die Funktion „Write“ des Fakturengenerators. Diese Funktion schreibt zum Zeitpunkt der Fakturierung die Rechnungszeilen-Informationen in die Datenbank.
- Kapitel 8 „**Fakturierungsprogramm Aufbau (FEDAS.Basic)**“ beschreibt den Aufbau eines Fakturierungsprogramms. Dieses Programm wird in der Programmiersprache Basic eingegeben, die wir um einige spezifische Funktionen erweitert haben.
- Kapitel 9 „**Funktionsbeschreibung (Referenz)**“ listet alle Funktionen, die in einem Fakturierungsprogramm verwendet werden können, alphabetisch auf und beschreibt sie.
- Kapitel 10 „**Datenprüfung (Prüfregeln, Logging)**“ definiert das Verhalten des Systems bei der Prüfung von Auftragsdaten der Klienten-Auftragsverwaltung. Auftragsdaten müssen eine hohe Qualität aufweisen, damit daraus korrekte Rechnungen entstehen können. Dazu können Prüfregeln definiert und das Fakturierungsprogramm durch Logging beim Ablauf überwacht werden.
- Kapiel 11 „**Dialoge Systemdaten Designer**“ dokumentiert den Systemdaten Designer. Er ist für die Parametrierung des Systems zur Verarbeitung der Auftragsdaten zuständig.
- Kapitel 12 „**Fakturierungsprogramm Beispielübersicht**“ beschreibt die Art und Weise, wie wir die Beispielprogramme in Kapitel 13 – Kapitel 15 dokumentieren.
- Kapitel 13 bis Kapitel 15 ist die Dokumentation drei konkreter Anwendungsbeispiele des Fakturengenerators (Beschreibung der Auftragsdaten, der Dialoge der Klienten-Auftragsverwaltung, die Datenbankinformationen und das Fakturierungsprogramm - inklusive Programmcode).

# 4. Fakturengenerator zur Laufzeit

## 4.1 Fakturengenerator Programmübersetzung

Wie bereits erwähnt, werden Fakturengenerator-Regeln in Form von Fakturierungsprogrammen für die Verarbeitung von Auftragsdaten erstellt. Fakturierungsprogramme repräsentieren Textdateien. Sie werden vom Fakturengenerator durch Datenprüfung und Programmübersetzung in eine binären Form übergeführt und anschließend in der Datenbank gespeichert (als sogenannte „binäre objects“ [1)]). Die binäre Form des Fakturierungsprogramms repräsentiert einen ablauffähigen (interpretierbaren) Code zur Erstellung der Fakturen (Automatikfakturen).




[1)] FBO Dateien repräsentieren die binäre Form eines Fakturierungsprogramms.

## 4.2 Übersetzungsfehler Fakturierungsprogramm

Bei der Programmübersetzung können vom Fakturengenerator folgende Übersetzungsfehler ausgegeben werden, wenn die Eingabe des Fakturierungsprogrammes fehlerhaft ist.

Anmerkung: Gewöhnlich steht bei dem Übersetzungsfehler die Zeilennummer der Zeile des Fakturierungsprogrammes, in der der Fehler auftritt, dabei.

- Keine Einsprungmarke für Absprungbefehl GOTO *Sprungmarke* vorhanden!
- Ungültige Zahl: ..., *Zeile*
- Ungültiger Text:..., *Zeile*
- Ungültiger Datumswert:..., *Zeile*
- Variable xxx wurde mit dem Datentyp yyy deklariert. Mit Datentyp yyy darf nur eine Variable deklariert werden!
- Inkompatible Typen, Zeile
- Operanden dieses Datentyps können nicht mit dem aktuellen Operator verrechnet werden, *Zeile*
- Konstanten dieses Datentyps können nicht negiert werden, *Zeile*
- Variable xxx wurde nicht deklariert, *Zeile*
- Variable xxx wurde mehrfach deklariert, *Zeile*
- Aufragsvariable xxx wurde nicht deklariert, *Zeile*
- Aufragsvariable xxx wurde mehrfach deklariert, *Zeile*
- Einsprungmarke *Sprungmarke* mehrfach vorhanden, *Zeile*
- Parameteranzahl der Funktion ist nicht korrekt (Minimal:xxx, Maximal:yyy, Aktuell:xxx), *Zeile*
- Ungültiger Aufruf der Prozedur, Zeile
  Anmerkung zum Übersetzungsfehler: bspw. Funktion Init, Write, etc- darf nicht als Parameter eingesetzt werden.
- Die Typen der eingesetzten Parameter sind ungültig, *Zeile*
- Inkonsistenter Zustand bei Übersetzung aufgetreten, *Zeile*
  Anmerkung zum Übersetzungsfehler: Die interne Variablenliste hat während der Übersetzung einen inkonsistenten Zustand eingenommen. Schwerwiegender Fehler!
- Linke Klammer fehlt, *Zeile*
- Logischer Operator innerhalb eines NOT-Ausdruckes nicht möglich, *Zeile*
- Parameter xxx der Funktion SearchDetails ist ungültig! , *Zeile*
- Keine weitere RECORD-Definition möglich, *Zeile*
- Schwerwiegender Ausnahmefehler bei der Übersetzung. Abbruch in *Zeile*
  Anmerkung zum Übersetzungsfehler: Dieser Fehler tritt auf, wenn der Fakturengenerator bei der Übersetzung ein internes Problem feststellet (Ausnahmebehandlung). Schwerwiegender Fehler!
- (Warning) Folgende Variable(n) wurde(n) deklariert, aber nie verwendet: ...

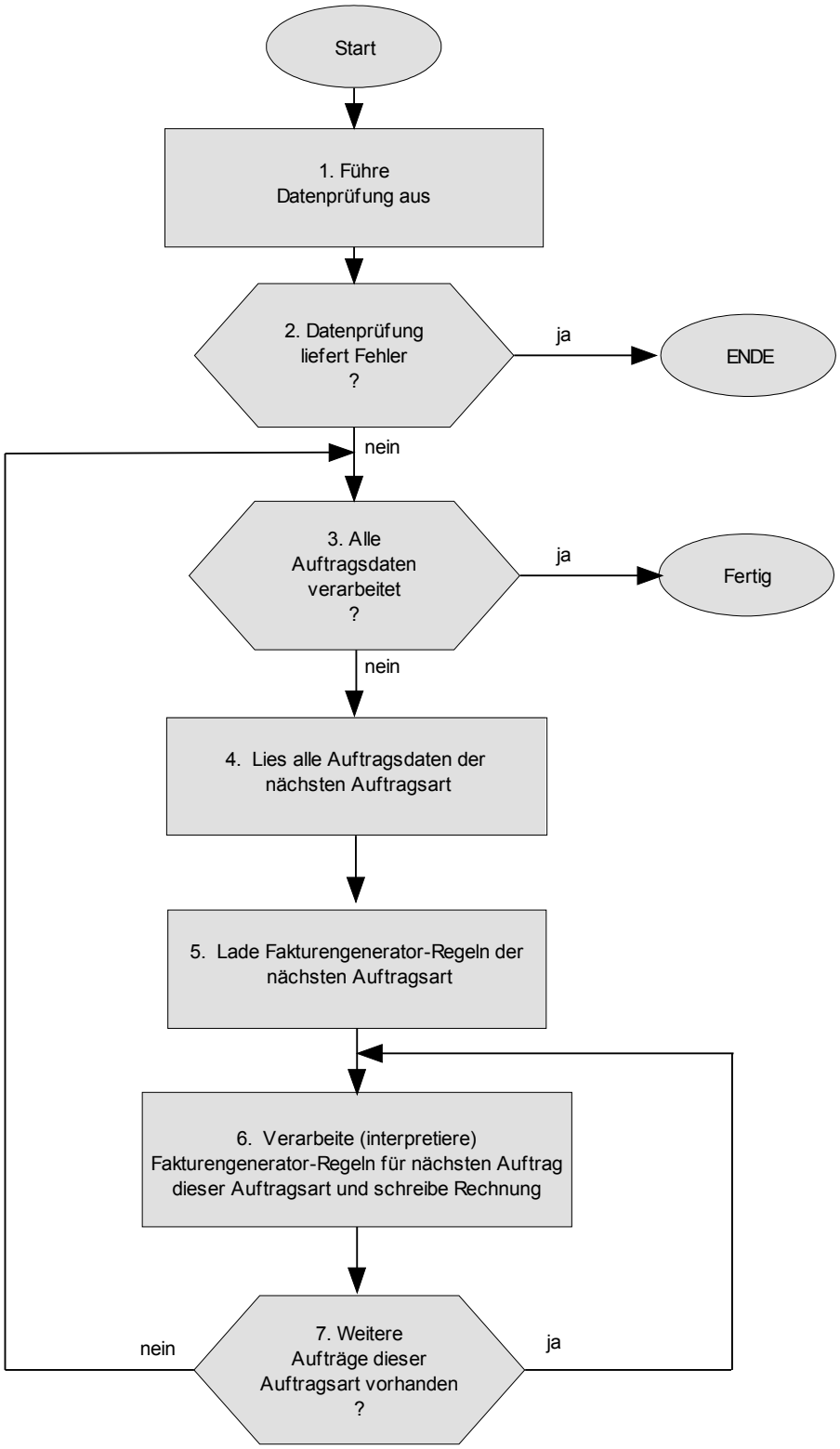
Für jede Fehlermeldung wird die Zeile in der Textdatei des Fakturierungsprogramms angegeben, in der der Fehler zu finden ist.

## 4.3 Ablaufschema

Die binäre Form der gespeicherten Fakturierungsprogramme wird zur Laufzeit, nach dem Start des Fakturengenerators, ausgeführt. Das bedeutet, dass noch nicht fakturierte Aufträge fakturiert und für diese Rechnungen erstellt werden (Fakturierung). Das Laufzeitsystem des Fakturengenerators (eingebaut in die Klienten-Auftragsverwaltung) führt das Fakturierungsprogramm für jeden Auftragsdatensatz, der dem Klienten zugeordnet ist, einmal aus [1)]. Bei diesem Vorgang werden die Rechnungszeilen je Klienten-Auftrag fortgeschrieben. Für alle gespeicherten Auftragsdaten werden also automatisch die Rechnungen erstellt (Automatikfakturen). Das Verfahrensschema sehen Sie in der folgenden Abbildung.

[1)] Durch Interpretation der Anweisungen im „binären Fakturierungsprogramm“. Die Anweisungen werden vor Abarbeitung durch die Programmübersetzung mittels „Compilation“ auf binäre Form gebracht. Der Fakturengenerator verfügt daher über einen Laufzeitinterpreter zur Interpretation der Anweisungen.

Fakturengenerator zur Laufzeit:




(Erklärung siehe nächste Seite)

Erklärung Fakturengenerator zur Laufzeit:

1. Führe Datenprüfung aus
   Als Voraussetzung für einen Fakturierungslauf muss sichergestellt sein, dass alle Auftragsdaten korrekt erfasst sind. Dazu ist zunächst obligatorisch die Funktion „Datenprüfung durchführen" zuständig (siehe Kapitel 5. „Fakturierung Dialogbedienung"). Hierbei wird jeder Auftrag überprüft und der entsprechende Status für den Auftrag gesetzt (FE_AUFTRAG.BESTA).
2. Datenprüfung liefert Fehler?
   Falls die Datenprüfung einen Auftrag findet, der die Prüfregeln verletzt, so wird die weitere Fakturierung unterbunden und der Anwender ist gezwungen die Auftragsdaten zu vervollständigen (zu korrigieren).
3. Alle Auftragsdaten verarbeitet?
   Das Laufzeitsystem des Fakturengenerators prüft, ob für diesen Lauf (weitere) Auftragsdaten vorhanden sind, deren Status „geprüft" ist.
4. Lies alle Auftragsdaten der nächsten Auftragsart (mit Status „geprüft")
   Hier werden die Auftragsdaten gelesen, die dieselbe Auftragsart haben und noch nicht fakturiert sind.
   Anmerkung: Intern werden zu Beginn des Laufs alle Aufträge aller Auftragsarten gelesen, nach Auftragsart sortiert und Auftragsart für Auftragsart vom Fakturengenerator bearbeitet.
5. Lade Fakturengenerator-Regeln der nächsten Auftragsart
   Hier ladet der Fakturengenerator jene Regeln (jenes Fakturierungsprogramm), die für die Fakturierung der Auftragsdaten dieser Auftragsart festgelegt wurden. Wechselt die Auftragsart, so muss der Fakturengenerator das für die Auftragsart vorgesehene Fakturierungsprogramm aus der Datenbank nachladen. Die Auftragsdaten werden dann mit dem neuen Fakturierungsprogramm im nächsten Schritt in Automatikfakturen übergeführt.
6. Verarbeite (interpretiere) Fakturengenerator-Regeln für nächsten Auftrag dieser Auftragsart und schreibe Rechnung
   Hier arbeitet der Fakturengenerator jede Zeile des Fakturierungsprogramms ab. Je nach Programmlogik werden Rechnungszeilen bzw. Rechnungsdaten in Form von Datensätzen in die Datenbank geschrieben (INVOICE bzw. INVOICEDATA). Dabei kann es sich pro Auftrag auch um mehrere (verschiedene) Rechnungen für ein und denselben Klienten an unterschiedliche Rechnungsempfänger handeln. Das bezeichnen wir dann als automatisches Rechnungssplitting. Nach Bearbeitung durch das Fakturierungsprogramm sind die Rechnungen in Form von „Offenen Posten" gespeichert (und in unserem FEDAS.DebitorKreditor Programm verfügbar, wenn mit der Anwendung die Massenbuchung bestätigt wird). Außerdem kann hier das Logging von Ereignissen einsetzen (dabei werden Fehlersituationen signalisiert/gespeichert).
   Mehrere Aufträge für denselben Klienten:
   Während ein und desselben Fakturierungslaufs kann es vorkommen, dass für einen Klienten mehrere unterschiedliche Aufträge fakturiert werden müssen. Für solche Aufträge des Klienten werden dann auch unterschiedliche Fakturengenerator-Regeln (Fakturierungsprogramme) verwendet und dabei wird jeweils ein neuer Eintrag in dem Datenbank-Objekt FE_RECH erstellt. Dies bedeutet, dass bei jedem Fakturierungslauf pro Klient mehrere Rechnungen entstehen könnten. Nun besteht eine Möglichkeit zur Zusammenfassung dieser Rechnungen zu einer gemeinsamen Sammelrechnung. Dazu gibt es im Fakturengenerator das Konzept der Sammelrechnungsnummer und Blattnummer. Das Konzept ist im Kapitel 6. „Rechnungsnummern-Vergabe" beschrieben.
7. Weitere Aufträge dieser Auftragsart vorhanden?
   Hier entscheidet der Fakturengenerator, ob weitere Aufträge abgearbeitet werden sollen. Wenn ja, wird die gleiche Fakturengenerator-Regel (das gleich Fakturierungsprogramm) nochmalig für den nächsten Auftrag abgearbeitet und die Rechnung(en) werden für den nächsten Kunden erstellt.

# 5. Fakturierung Dialogbedienung

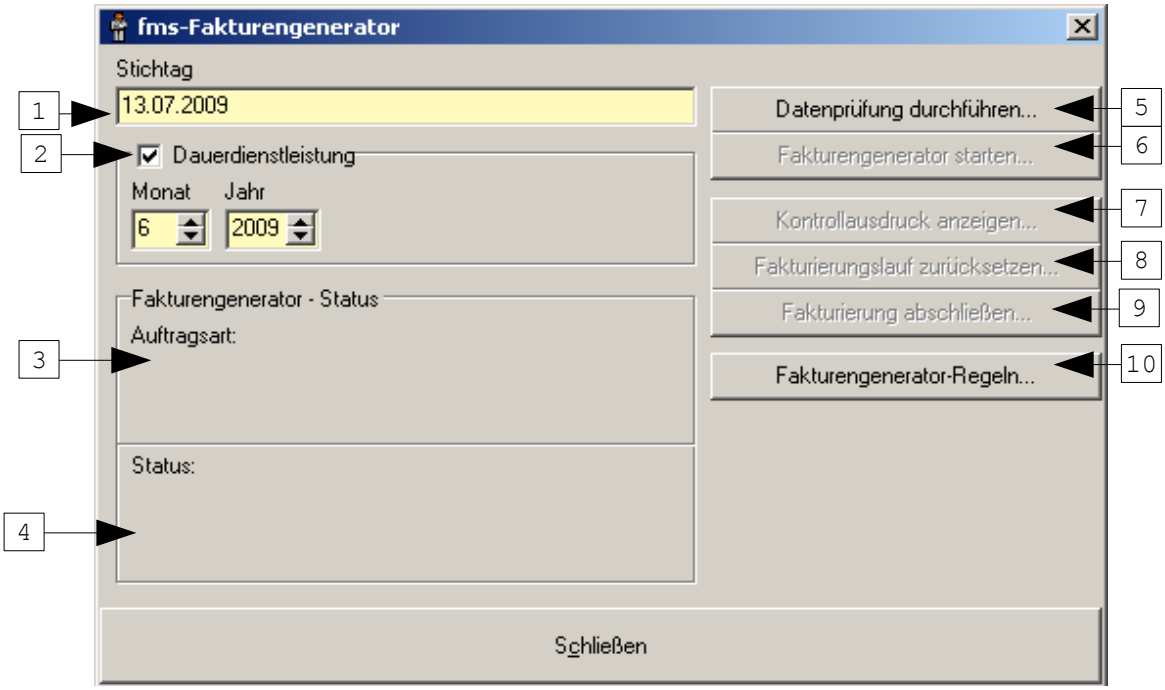
## 5.1 Dialog Übersicht

Die Fakturierung wird mit dem unten dargestellten Dialog gestartet. Bei diesem Dialog handelt es sich um eine Programmfunktion, die in jede Klienten-Auftragsverwaltung eingebaut werden kann und flexibel auf individuelle Anforderungen angepasst werden kann. Weiter unten bringen wir zwei konkrete Beispiele, wie der Dialog im Kontext von Klienten-Auftragsverwaltungen funktioniert. In diesem Kapitel erklären wir, wie der Anwender mit dem Fakturengenerator arbeitet. Es gibt zwei Modi für den Betrieb des Fakturengenerators:

- Mode für Einzelfakturen und
- Mode für Massenfakturen

Im Modus für Einzelfakturen wird der Fakturengenerator aufgerufen, um einen einzigen Auftrag zu fakturieren. Bei Massenfakuren werden entsprechend viele Aufträge fakturiert. Dementsprechend ist die Bedienung des Dialogs unterschiedlich.

### 5.1.1. Dialog Massenfakturen



## 5.1.2 Dialog Massenfakturen Bedienung

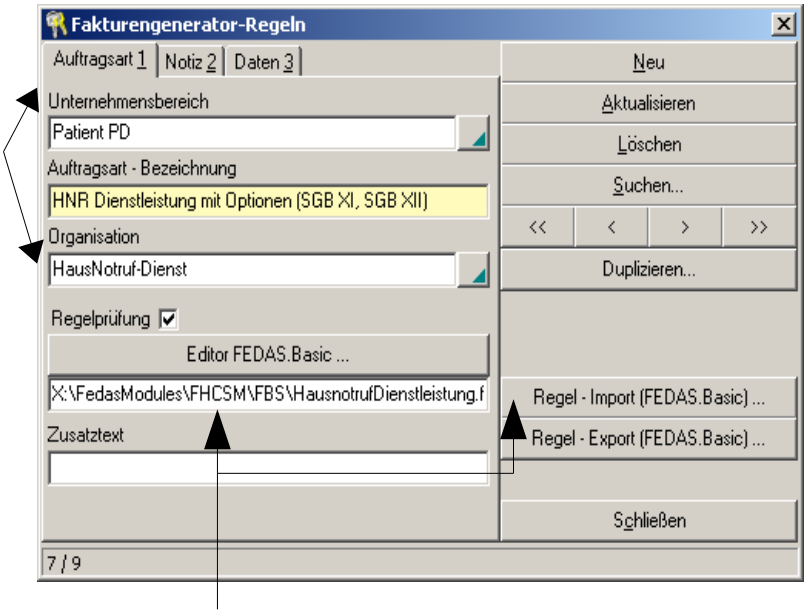
Wir kommentieren in diesem Abschnitt die Bedeutung der Felder, die in der vorigen Abbildung mit Nummern gekennzeichnet wurden.

1. Stichtag
   Dies ist ein Datum zur Steuerung der Fakturierung. Auftragsdaten werden in der Datenbank angelegt und das Datum der Datenspeicherung automatisch auf das Tagesdatum der Erfassung gesetzt. Dabei handelt es sich somit um das Speicherdatum mit der ein Auftrag versehen ist (Tabellenspalte: FE_AUFTRAG.SPEIDA). Die Fakturierung geht nun auf dieses Speicherungsdatum und vergleicht dieses Datum mit dem Stichtag. Liegt das Speicherungsdatum vom Stichtag aus gesehen weiter in der Vergangenheit, so wird der Auftrag zur Fakturierung herangezogen. Wenn das Speicherungsdatum des Auftrags gleich dem des Stichtags ist, so wird der Auftrag ebenfalls fakturiert. Liegt das Datum vom Stichtag aus gesehen in der Zukunft, so wird der Auftrag nicht zur Fakturierung herangezogen. Gewöhnlich wird man für den Stichtag immer das Tagesdatum eingeben.
2. Dauerdienstleistung
   Mit diesem Merkmal legen Sie fest, ob der Fakturengenerator Daueraufträge fakturieren soll. Ein Dauerauftrag ist in der Datenbank als solcher gekennzeichnet (Tabellenspalte: DAUERAUFTRAGKEN). Daueraufträge sind dadurch charakterisiert, dass sich ihre Auftragsdaten in der Regel von Monat zu Monat nicht ändern. Aus diesem Grund werden die Auftragsdaten nur einmalig erfasst und selten verändert. Als „Daueraufträge“ werden sie daher unabhängig vom eingegebenen Stichtag fakturiert und die aktuell erfassten (gleich bleibenden) Auftragsdaten werden zur Fakturierung verwendet. Beispiel für einen Dauerauftrag: Der einmalig eingerichtete Hausnotrufauftrag (entspricht einem Mietverhältnis).
3. Auftragsart
   Während der Fakturengenerator arbeitet wird hier angezeigt, welche Auftragsart gerade aktuell bearbeitet wird. Die Aufträge werden sortiert nach Auftragsart hintereinander abgearbeitet.
4. Status
   Während der Fakturengenerator arbeitet wird hier angezeigt, der wievielte Auftrag der aktuellen Auftragsart gerade fakturiert wird.
5. Datenprüfung durchführen
   Es ist erforderlich, dass vor dem eigentlichen Fakturengenerator-Start die gespeicherten Auftragsdaten nach bestimmten (inhaltlichen und strukturellen) Regeln geprüft werden. Jeder Auftrag wird nach erfolgreicher Prüfung auf „geprüft“ gesetzt. Erst wenn die Datenprüfung für jeden Auftrag erfolgreich abgeschlossen werden kann, können Sie die eigentliche Fakturierung starten. Im Kapitel 10. „Datenprüfung“ erfahren Sie zur Datenprüfung weitere Details.
6. Fakturengenerator starten
   Wenn die Datenprüfung in den Auftragsdaten keine Fehler bei der Datenprüfung festgestellt hat, kann eine Fakturierung erfolgen (gestartet werden). Die Fakturierung kann selbst auch Fehler zutage fördern, indem der Fakturengenerator feststellt, dass die zur Fakturierung verwendeten Daten nicht konsistent sind (strukturelle Fehler). Beispielsweise könnte ein Kostenfaktor einer bestimmten Kasse zur Bestimmung eines Kassenpreises nicht erfasst sein. Im Kapitel 10. „Datenprüfung“ erfahren Sie zur Datenprüfung weitere Details.
7. Kontrollausdruck anzeigen
   Hier können Sie eine Übersicht über alle vom Fakturengenerator erstellten Fakturen anzeigen lassen. Erstmalig durch das System fakturierte Klienten-Auftragsdaten werden im Kontrollausdruck (in der angezeigten Tabellenübersicht) speziell gekennzeichnet. Hier können Sie die Plausibilität der Fakturierung kontrollieren. Das Anzeigen des Kontrollausdrucks ist nur möglich, wenn der Fakturengenerator beim vorangehenden Lauf fehlerfrei beendet wurde.
8. Fakturierungslauf zurücksetzen
   Wenn Sie bei der Überprüfung des Kontrollausdrucks feststellen, dass eine Rechnung nicht korrekt (nicht plausibel) ist, so können Sie durch diese Funktion den vollständigen Fakturierungslauf zurücknehmen. Intern werden die zu diesem Zeitpunkt noch nicht gebuchten

Fakturen gelöscht und die Auftragsdaten auf den Status „nicht fakturiert“ zurückgesetzt (FE_AUFTRAG.BESTA).

9. Fakturierung abschließen
   Nachdem Sie die Überprüfung des Kontrollausdrucks abgeschlossen haben und alle Fakturen akzeptieren, können Sie die Fakturierung abschließen. Intern werden dabei die bis vor diesem Zeitpunkt noch nicht gebuchten Fakturen auf gebucht gesetzt. Ab diesem Zeitpunkt können die Fakturen als offene Posten weiter verarbeitet, jedoch nicht mehr verändert werden (Programm FEDAS.DebitorKreditor).
10. Fakturengenerator-Regeln
    Hier rufen Sie die Funktion zur Verwaltung aller Fakturengenerator-Regeln auf. Wie bereits dargestellt sind das Fakturierungsprogramme, welche die Auftragsdaten in Rechnungsdaten umwandeln (FEDAS.Basic). Im Dialog unten ist dargestellt wie Sie den Editor zur Eingabe des FEDAS.Basic Programms aufrufen.

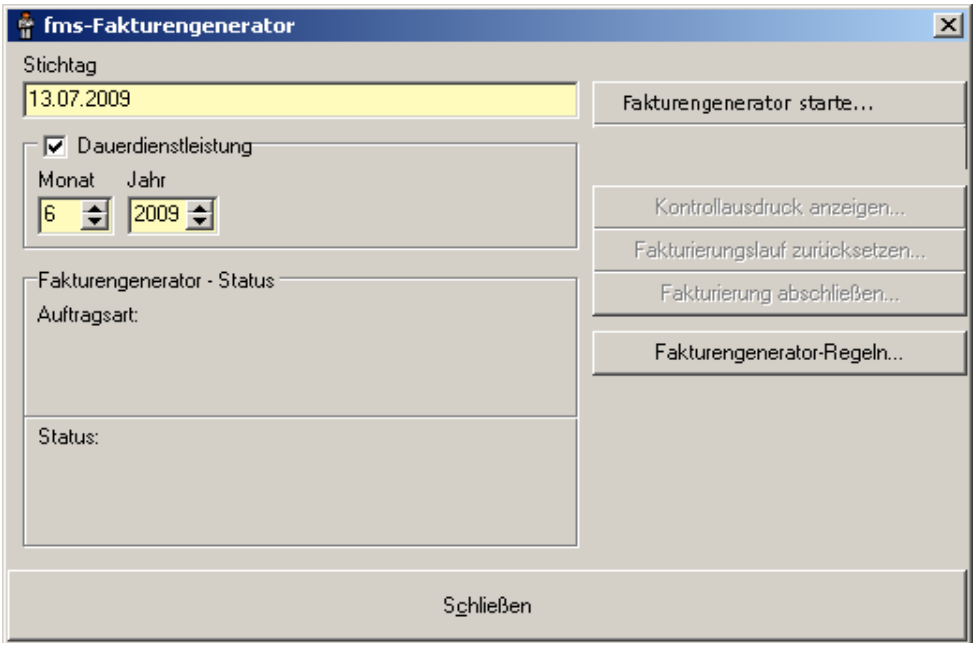
Ein FEDAS.Basic Programm wird für eine bestimmte Auftragsart (also für gleich bleibende Daten) erstellt. Innerhalb der Organisation ist die Auftragsart einem „Unternehmensbereich“ (d.h. einer Spartenorganisation) zugeordnet. Eine Spartenorganisations(einheit) ist bestimmt durch Unternehmensbereich (Dienstleistungsbereich) und die Organisation.
Die „Regelprüfung“ legt fest, ob fehlerhafte Eingaben für das FEDAS.Basic Programm dem Programmbenutzer gemeldet werden sollen.




Hier kann das Fakturierungsprogramm (FEDAS.Basic) ausgewählt, in den Text-Editor geladen und bearbeitet werden. Beim Schließen des Editors erfolgt automatisch die Übersetzung durch den Fakturengenerator (und ggf. erfolgt dabei die Ausgabe der Fehlermeldungen). Beim Drücken von „Aktualisieren“ wird das Fakturierungsprogramm in binärer (übersetzter) Form in der Datenbank gespeichert. Das gespeicherte Fakturierungsprogramm kann vom Fakturengenerator dann später bei der Fakturierung ausgeführt werden. Siehe dazu Kapitel 4. „Fakturengenerator“ (Programmübersetzung, Laufzeit) .

## 5.2.1. Dialog Einzelfakturen

Die folgende Abbildung stellt den Fakturierungsdialog für Einzelfakturen dar. Dieser unterscheidet sich von dem Dialog für Massenfakturen dadurch, dass der Schalter „Datenprüfung durchführen“ nicht vorhanden ist.




## 5.2.1. Dialog Einzelfakturen Bedienung

Die Bedienung dieses Dialogs funktioniert analog zur Bedienung des den Massenfakturen Dialogs. „Datenprüfung durchführen“ wird automatisch aktiviert und nach erfolgreicher (impliziter) Datenprüfung wird der Kontrollausdruck angezeigt.

# 6. Rechnungsnummern - Vergabe

Rechnungen (Fakturen) sind immer mit Nummern versehen. Diese Nummern (Rechnungsnummern) realisieren eine ordnungsgemäße Buchhaltung. Für den Fakturengenerator sind dabei die vier folgenden Nummern von Relevanz (siehe Datenbank-Objekt FE_RECH):

1. Fakturierungsnummer (FAKTURIERNUM)
2. Rechnungsnummer (Belegnummer – BELEGNUM, Buchungskreiszähler – BUCHKREISZAEHLER [1] )
3. Sammelrechnungsnummer (SAMMELRECHNUM)
4. Blattnummer (BLATTNUM)

[1] BELEGNUM und BUCHKREISZAEHLER sind Felder der Datenbank, die im Zusammenhang mit Rechnungen wichtig sind.

In den folgenden Ausführungen wollen wir auf die Vergabe dieser Nummern näher eingehen.

## 6.1 Fakturierungsnummer

Diese Nummer wird für jeden Fakturierungslauf neu vergeben. Jede Rechnung, die bei einem Fakturierungslauf entsteht, hat die Fakturierungsnummer dieses Laufs. Diese Nummer ist eine organisatorische Größe und hat für die Buchhaltung eine untergeordnete Bedeutung.

## 6.2 Rechnungsnummer (Buchungskreiszähler, Belegnummer)

Rechnungsnummern müssen (gemäß den Grundsätzen ordnungsgemäßer Buchhaltung) fortlaufend sein. Ein Buchungskreiszähler kann, neben der Belegnummer, ein Bestandteil der Rechnungsnummer sein und wird vom Fakturengenerator als laufende Nummer für jede Ausgangsrechnung in die Datenbank eingetragen. Für den Buchungskreiszähler kann ein Nummernkreis festgelegt werden. Er dient hauptsächlich der Belegorganisation in der Ablage. Die Belegnummer ist für Ausgangsrechnungen eine unveränderbar fortlaufende Nummer und dient der Identifikation der Rechnung bei der Verarbeitung von Eingangszahlungen mit dem Programm FEDAS.DebitorKreditor.
Es ist möglich einen einzigen Buchungskreiszähler-Nummernkreis für sämtliche Ausgangsrechnungen zu verwenden, die mit unserer Software erstellt werden. Alternative Vorgehensweisen in der Buchhaltung setzen hier für unterschiedliche „Spartenrechnungen“ unterschiedliche Nummernkreise ein. Wenn Ausgangsrechnungen zentral von einer Stelle erstellt werden (bspw. mit unserer Software), sind unterschiedliche Nummernkreise allerdings nicht notwendig.

<u>Nummernkreise:</u>
Nummernkreise für „Buchungsobjekte“ wie Ausgangsrechnungen (um diese geht es hier), können im Programm FEDAS.DebitorKreditor für unterschiedliche Arten von Rechnungen festgelegt werden. Näheres zur Konfiguration dieser Nummernkreise (Buchungskreiszähler) erfahren Sie im Handbuch für FEDAS.DebitorKreditor.

## 6.3 Sammelrechnungsnummer

Die Sammelrechnungsnummer wird vom Fakturengenerator automatisch vergeben, wenn ein Rechnungsempfänger (Kostenträger) die Kosten für mehrere Klienten übernimmt. Sie ist eindeutig und kann auch durch Programmoptionen nicht gesteuert werden (sie wird intern in der Datenbank als Zähler gespeichert, fortlaufend hochgezählt und für eine neue Sammelrechnung vergeben). Eine Sammelrechnung kann selbst direkt nicht eingegeben werden, sondern diese wird immer automatisch vom Fakturengenerator erstellt. Im Zusammenhang mit der Sammelrechnungsnummer spielt die Blattnummer eine wichtige Rolle. Das Konzept der Blattnummer erklären wir im nächsten Kapitel.

# FEDAS.Basic (Fakturierungsprogramm)

*Fakturierungsprogramm:*
Einleitender Kommentar
**START**
*Einfache Benutzervariable*
*Komplexe Benutzervariable*
*Auftragsvariable Datenbank*
**PROGRAM BERECHNUNG**
*Fakturierungsanweisungen*
**END**
**END**

Ein Fakturierungsprogramm hat die dargestellte Struktur

Eine Variable muss mit einem Buchstaben beginnen und kann Zahlen beinhalten.

*Einfache Benutzervariable*
: **DIM** Variablenname **AS NUMBER**
| **DIM** Variablenname **AS STRING**
| **DIM** Variablenname **AS DATE**
| **STATIC** Variablenname **AS NUMBER**
| **STATIC** Variablenname **AS DATE**
| **STATIC** Variablenname **AS STRING**
;

Einfache Benutzervariablen können das Schlüsselwort STATIC vorangestellt haben. Damit kann die Variable nur einmalig pro Auftragsart zugewiesen werden (weitere Zuweisungen werden ignoriert). Der Variableninhalt bleibt für alle Aufträge (der Auftragsart) erhalten, die hintereinander fakturiert werden. Jegliche weitere Zuweisungen innerhalb derselben Auftragsart für denselben oder einen anderen Auftrag werden ignoriert.

*Komplexe Benutzervariable*
**: DIM** Variablenname **AS TABLE**
| **DIM** Variablenname **AS SYMTABLE**
| **DIM** Variablenname **AS SYMKEYTABLE**
| **DIM** Variablenname **AS INVOICE**
| **DIM** Variablenname **AS INVOICEDATA**
| **DIM** Variablenname **AS COMMENT**
| **DIM** Variablenname **AS MATRIX**
;

Komplexe Benutzervariablen werden dazu benutzt Daten bestimmter Struktur aus der Datenbank zu lesen:
TABLE, SYMTABLE, SYMKEYTABLE, MATRIX
bzw. zu schreiben:
INVOICE, INVOICEDATA, COMMENT.

*Auftragsvariable Datenbank*
: **RECORD** Auftragsvariable-Name
Auftragsvariable-Name-Alias
*Auftragsvariable-Liste*
**END**

Die Auftragsvariable Datenbank repräsentiert die Daten des aktuellen Auftrags einer Auftragsart. Auftragsvariable-Name-Alias ist das gemeinsame Merkmal für „Auftragsdatenkopf" und „Auftragsdatendetails". Auftragsdaten sind analog zu einer Rechnung aufgebaut (Rechnungskopf, Rechnungszeilen).

*Auftragsvariable-Liste*
: **DIM** Variablenname **AS NUMBER**
| **DIM** Variablenname **AS STRING**
| **DIM** Variablenname **AS DATE**
;

Diese Daten sind in der Datenbank gespeichert und entsprechen den Tabellenspalten von Datenbankobjekten.

*Fakturierungsanweisungen*
: Label *Fakturierungsanweisung*
| *Fakturierungsanweisung*
;

*Fakturierungsanweisung*
: **IF** *Ausdruck* **THEN GOTO** Label
| Variable = *Ausdruck*
| **GOTO** Label
| *Funktion*
;

*Ausdruck*

: *Rekursiver-Ausdruck* **OR** *Rekursiver-Ausdruck*
| *Rekursiver-Ausdruck* **AND** *Rekursiver-Ausdruck*
| **NOT** *Rekursiver-Ausdruck*
| **(** Rekursiver-Ausdruck **)**
| *Rekursiver-Ausdruck* **LIKE** *Rekursiver-Ausdruck*
| *Rekursiver-Ausdruck* **>** *Rekursiver-Ausdruck*
| *Rekursiver-Ausdruck* **<** *Rekursiver-Ausdruck*
| *Rekursiver-Ausdruck* **=** *Rekursiver-Ausdruck*
| *Rekursiver-Ausdruck* **>=** *Rekursiver-Ausdruck*
| *Rekursiver-Ausdruck* **<=** *Rekursiver-Ausdruck*
| *Rekursiver-Ausdruck* **<>** *Rekursiver-Ausdruck*
| *Rekursiver-Ausdruck* **+** *Rekursiver-Ausdruck*
| *Rekursiver-Ausdruck* **–** *Rekursiver-Ausdruck*
| *Rekursiver-Ausdruck* ***** *Rekursiver-Ausdruck*
| *Rekursiver-Ausdruck* **/** *Rekursiver-Ausdruck*
| *Funktion*
| *Einfacher-Ausdruck*
;

Logische Operatoren: OR AND (OR mit Priorität)
Vergleichsoperatoren: LIKE > < = >= <= <>
Binäre Operatoren: + - * /

Die Anwendung der Operatoren unterliegt Regeln
**Für Ausdrücke mit dem Datentyp NUMBER:**
- alle binären Operatoren
- alle Vergleichsoperatoren (außer LIKE)
**Für Ausdrücke mit dem Datentyp STRING:**
- binäres + (zum Verknüpfen von Texten)
- alle Vergleichsoperatoren
**Für Ausdrücke mit dem Datentyp DATE:**
- alle Vergleichsoperatoren (außer LIKE)

Binäre Beispielausdrücke:
`A + B`
(A + 123.45) > 200
10 + -5

Beispielausdrücke mit logischem Operator:
(A >= B) AND (C < 100)
(A = Get(...)) OR (D > 01.01.2004)
Anmerkung: eine Funktion darf in einem log. Ausdruck nur einmal (im linken Teilausdruck) angewendet werden!

Ein Teilausdruck kann TRUE oder FALSE ergeben (TRUE=0)

Rekursiver-Ausdruck

: Ein Ausdruck, der beliebig
oft wiederholt werden kann.
;

*Einfacher-Ausdruck*

: **-** *Zahlenkonstant*e
| - Variable
| *Zahlenkonstante*
| Variable
| *Stringkonstante*
| Auftragsvariable-Name.Variablenname
| *Datumskonstante*
;

Beispielausdrücke mit LIKE:

| | |
|---|---|
| Tanne LIKE T* | ergibt TRUE |
| Tee LIKE T* | ergibt TRUE |
| T LIKE T* | ergibt TRUE |
| Auto LIKE T* | ergibt FALSE |
| Haus LIKE ?aus | ergibt TRUE |
| Maus LIKE ?aus | ergibt TRUE |
| Klaus LIKE ?aus | ergibt FALSE |

Diese Variablen als „Auftragsvariable“ speichern automatisch immer die Inhalte der Datenbank für den aktuellen Auftrag, den das Fakturierungsprogramm gerade bearbeitet. Auf solche Variablen ist keine Zuweisung möglich.

*Zahlenkonstante*

: **TRUE**
| **FALSE**
| **SUNDAY**
| **SATURDAY**
| **NULL**
| **LOG_OFF**
| **LOG_HINT**
| **LOG_WARNING**
| **LOG_ERROR**
| HH:MM (Minuten seit 1.1.1800)
| Fließkommazahl
;

Die Begriffe TRUE, FALSE,...LOG_ERROR sind Schlüsselwörter, die in einem Fakturierungsprogramm verwendet werden können. Sie repräsentieren verschiedene Konstantwerte.
HH:MM ist 06:05 als 6 Uhr 5 Minuten ist eine „NUMBER“ Konstante (intern als Minuten seit 01.01.1800 gespeichert).
Eine Fließkommazahl hat einen Punkt zwischen den „Vorkommastellen“ und „Nachkommastellen“.

*Stringkonstante*

: **NULL**
| Freitext
;

Freitext ist ein beliebiger Text bestehend aus Ziffern, Buchstaben und Sonderzeichen. Am Beginn und Ende steht ein doppeltes Hochkomma.

*Datumskonstante*

: **NULL**
| TT.MM.JJJJ:HH:MM
;

Folgende Datumsangabe ist möglich:
TT.MM.JJJJ bspw. 01.01.2004
Folgende Datumsangabe mit Zeit ist möglich:
DD.MM.YYYY:HH:MI 01.01.2004:12:30

*Funktion*

```
: AddString ( )
| Avg ()
| Date$ ()
| DateAdd ()
| DateDiff ()
| Day()
| DateOf()
| Filter ()
| FStr$ ()
| Get ()
| Init ()
| Input ()
| IsValid ()
| Max ()
| Min ()
| MinutesDiff ()
| Next ()
| Now
| Print ()
| Searchdetails ()
| Sum ()
| Time ()
| TimeDiff ()
| Weekday ()
| Write()
;
```

Die Parameter der Funktionen sind im Kapitel 9. „Funktionsbeschreibung (Referenz)“ beschrieben.

# 9. Funktionsbeschreibung (Referenz)

## AddString

( Text1 :**STRING**, Text2: **STRING ):STRING**

Verknüpft die beiden Texte „Text1“ und „Text2“ zu einem resultierenden Text.

1. **Text1**:Vorangestellter Text
2. **Text2**:Nachfolgender Text

---

## Avg

(Datenquelle: **TABLE**, Feldname: **STRING ):NUMBER**

Liefert den durchschnittlichen Wert der Spalte „Feldname“ für alle Einträge der Tabelle „Datenquelle“. Feldname muss sich auf eine Tabellenspalte mit numerischen Werten beziehen.

1. **Datenquelle**: Beliebiges Tabellenobjekt, welches im Fakturierungsprogramm definiert ist.
2. **Feldname**: Name der Tabellenspalte, von welcher der Durchschnittswert ermittelt werden soll.

---

## Date$

**(**Datum: **DATE**, Formatstring: **STRING ):STRING**

Liefert das übergebene „Datum“ als formatierten Text zurück. Die formatierte Textausgabe wird durch die Variable „Formatstring“ gesteuert.

1. **Datum**: beliebiges Datum.
2. **Formatstring**: Formatierungsstring mit folgenden Möglichkeiten für die Steuerung der formatierten Textausgabe:

| | |
|---|---|
| %d | Aktuelles Tagesdatum ohne vorangestellte Null (1 – 31) |
| %02d | Aktuelles Tagesdatum mit vorangestellter Null (01-31) |
| %D | Aktueller Wochentag ("Montag" – "Sonntag") |
| %m | Aktuelles Monat ohne vorangestellte Null (1 – 12) |
| %02m | Aktuelles Monat mit vorangestellter Null (01-12) |
| %M | Aktuelles Monat ausgeschrieben ("Jänner" – "Dezember") |
| %y | Aktuelles Jahr, 4-stellig |
| %h | Aktuelle Zeit, Stunden ohne vorangestellte Null (0 – 23) |
| %02h | Aktuelle Zeit, Stunden mit vorangestellter Null (0-23) |
| %i | Aktuelle Zeit, Minuten ohne vorangestellte Null (0 – 59) |
| %02i | Aktuelle Zeit, Minuten mit vorangestellter Null (00-59) |

Leer- und Trennzeichen können beliebig vorgesehen werden.

Beispiele:

| | |
|---|---|
| "%02d.%02m.%y" | 01.01.2004 |
| "%02d-%02m-%y %02h:%02i" | 01-01-2004 12:30 |
| "%d.%m.%y" | 1.1.2004 |
| "%02d. mmmm yyyy" | 01. Jänner 2004 |
| "dddd, d. mmmm yyyy" | Montag, 1 Jänner 2004 |