Generierung guter Testdaten für das BI-System

Felix Krul
areto consulting gmbh
Köln

max. 9000 Zeichen

Schlüsselworte
Testdaten, Datenverfälschung, randomisierte Datenauswahl, manuelle Testdatenerzeugung, Pseudozufallszahlen, zueinander passende Daten erzeugen

Einleitung: Kriterien für gute Testdaten

Gute Testdaten für ein BI-System müssen einige sehr unterschiedliche Prüfungen erlauben, zum Beispiel

- ob die richtigen Daten korrekt verarbeitet werden,
- ob die Verarbeitung von Massendaten zuverlässig und korrekt erfolgt,
- ob fehlerhafte Daten auf die vorher definierte Weise behandelt werden,
- ob die Performance der Verarbeitung den Anforderungen entspricht,
- ob die Anforderungen an die Zugriffssicherheit erfüllt werden.

Darüber hinaus werden Testdaten auch im Berichtswesen verwendet, um (ggf. bevor Echtdaten bereitstehen) Standardberichte initial zu erstellen und ihr Abfrage- und Darstellungsverhalten zu überprüfen.

Daraus folgen Kriterien für gute Testdaten:

- Sie müssen aussagekräftig sein, also den Echtdaten inhaltlich ähnlich sein, da sie deren „Stellvertreter“ sind. Um zu testen, ob korrekte Daten korrekten verarbeitet werden, müssen die Testdaten daher inhaltlich und strukturell ähnlich sein. Beispiele hierfür: In einem Elektrofachmarkt werden keine Bananen verkauft, Internetlogs sollten nicht aus dem Jahr 1960 stammen, ein Mehrwertsteuerbetrag ist nie größer als der zugehörige Bruttowert.
  In einem Datumsfeld sollten echte Datumswerte stehen, in einem Verkaufsbetrag nur Zahlen mit zwei Nachkommastellen. Um Zeilenumbrüche eines Textfeldes zu prüfen, werden andere Texte als „Test“ benötigt.
- Sie müssen alle relevanten Fälle vollständig abdecken: In einem BI-System gibt es häufig mehrere verschiedene Verarbeitungsregeln, je nach der Charakteristik des Einzelsatzes. Beispielsweise müssen im Lagersystem eines Supermarktes für Lebensmittel Verfallsdaten berücksichtigt werden, für Schreibwaren aber nicht. Alle im ETL definierten Fälle müssen auch in den Testdaten vorkommen.
- Sie müssen zueinander passen: Daten können in sehr unterschiedlichen Beziehungen zu einander stehen, wie z.B. Bewegungsdaten, die nur zu gültigen Stammsätzen erzeugt werden dürfen, aber auch inhaltlich, wie Straße, Hausnummer, PLZ und Ort, die in Adressdaten zueinander passen müssen, oder Kontensalden, die gleich der Summe aller Einzelbuchungen seit Kontoeröffnung sind.

- Sie müssen ausreichend sein: Für die Tests von Massendatenverarbeitungen müssen entsprechende Datenmengen zur Verfügung stehen; um die Historisierung, sei es als SCD1 oder 2, einer Dimension zu testen, müssen einige Sätze in mehreren Versionen geliefert werden; um in einem Bericht die Zeilenumbrüche zu testen, müssen mindestens Daten für zwei Seiten abgefragt werden; für einen Bericht, der zwölf Monate darstellt, sollten mindestens zwölf Monate mit Daten abfragbar sein.
- Sie müssen alle relevanten Fehlerfälle enthalten: Auch die Frage, wie fehlerhafte Daten verarbeitet werden, ist im BI-System von großer Bedeutung. Entsprechende Testfälle (auch diese müssen vollständig sein) sind zu definieren und mit entsprechenden Testdaten zu versorgen.

Echtdaten als Quelle für Testdaten

Sofern ein Quellsystem schon Daten liefern kann, werden Testdaten oft als randomisierter Teilabzug generiert: Eine zufällige Teilmenge (z.B. 1% der Daten) wird ausgewählt und verarbeitet.

Dies kann in der Praxis komplexer sein, als zunächst angenommen. Wie oben erwähnt, müssen die Daten zusammenpassen, d.h. werden die Stamm- und Bewegungsdaten unabhängig voneinander randomisiert ausgewählt, so werden sie mit hoher Wahrscheinlichkeit nicht gut zusammenpassen. Es ist zielführender, in ein oder zwei Stammdatentabellen randomisiert auszuwählen und aus den dazugehörigen Bewegungsdaten Teilmengen auszuwählen. Danach werden in den restlichen Stammdaten die dazugehörenden Sätze ausgewählt.

Diese Daten sind nach den oben beschriebenen Kriterien aussagekräftig und passen zueinander, ob sie vollständig und ausreichend sind, ist aber nicht automatisch gegeben, und alle relevanten Fehlerfälle sind fast sicher nicht enthalten.

Randomisierte Auswahlen sind außerdem nicht deterministisch (wenigstens nicht nachvollziehbar), dieselbe Auswahl ist also beispielsweise unter Einsatz der RAND()-Funktion nicht ein zweites Mal identisch möglich.

Der Umgang mit Echtdaten bedeutet oft auch, dass Daten aktiv verfälscht und/oder anonymisiert werden müssen, was eigene Verfahren erfordert. Texte zur Anonymisierung zu leeren, ist einfach, es wird aber schwierig, wenn sie konsistent anderweitig befüllt werden sollen. Hier könnten beispielsweise Hash-Funktionen eingesetzt werden, diese erzeugen allerdings keine echt repräsentativen Einträge (der MD5-Hash von „Müller“ ist e35bc0a78f1c870124dfc1bbbd23721f), was aber oft nicht problematisch ist.
Die sinnvolle Verfälschung von Zahlen erfordert wiederum, dass die Verfälschung nicht umgekehrt werden kann, weil sonst das Original wiederhergestellt werden könnte. Weiter unten werden einige Verfahren zur Erzeugung von vielen unterschiedlichen Werten beschrieben.

Insgesamt zeigt sich aber, dass die oben gestellten Anforderungen an Testdaten oft nur mit selbst erzeugten Daten erfüllt werden können. Dies wird im folgenden beschrieben.

Eigene Stammdaten erzeugen

Dies ist der mühsamste, aber auch wichtigste Schritt in der Testdatenerzeugung, weil er viel manuelle Arbeit erfordert:

<Sollen beispielsweise Einkaufsdaten für hundert Produkte erzeugt werden, so müssen hundert Produkte als Stammdatensätze erzeugt werden. Soll die SCD2-Historisierung dieser Produkte geprüft werden, so müssen wenigstens für einige der Produkte mehrere Sätze erzeugt werden.

Kundendaten sind ähnlich aufwändig zu erzeugen, da beispielsweise die Adressdaten zusammenpassen sollten. Es kann auch schwierig sein, mehrere hundert unterschiedliche Namen zu erfinden.

Diese Mühe lohnt sich aber: Zum einen kann so sichergestellt werden, dass die Daten vollständig (alle interessanten Fälle sind abgedeckt) und ausreichend sind, zum anderen können explizit Fehler eingebaut werden, die die Testfälle der Fehlerverarbeitung bedienen. Beispiele: Nicht passende Adressdaten, die von der Adressprüfung korrigiert werden müssen; hierarchische Zuordnung von Produkten zu nicht existierenden Produktgruppen; Lücken oder Überlappe in der Historisierung usw.

Ein wichtiger Hinweis:

Zu den Stammdaten müssen später Bewegungsdaten erzeugt werden, dazu müssen die Schlüssel der Stammdaten zu Tupeln kombiniert werden. Dieser Vorgang ist über Integer-Schlüssel sehr viel einfacher als über alphanumerische Schlüssel zu erreichen! Daher ist die dringende Empfehlung, den natürlichen Stammdatenschlüssel um einen zugehörigen künstlichen Integerschlüssel zu ergänzen. Dieser kann in einem zweiten Schritt auch wieder in den „echten“ Schlüssel umgewandelt werden, aber zur initialen Erzeugung der Sätze eignen sich Integer deutlich besser, siehe unten.

Ein Beispiel sind Sätze in der folgenden Produkttabelle:

INTEGER_ID	PRODUKT_NR	PRODUKT	MENGEN-EINHEIT	STUECK-PREIS	MWST_SATZ	PRODUKT-GRUPPE	GUELTIG_AB	GUELTIG_BIS
1	AL00001	Pils	Flasche	1,19	7	AL	01.01.2012	31.12.2013
1	AL00001	Pils	Flasche	1,19	19	BIE	01.01.2014	31.12.9999
2	AL00002	Export	Flasche	1,29	7	AL	01.01.2012	31.12.2013
2	AL00002	Export	Flasche	1,29	19	BIE	01.01.2014	31.12.9999
3	AL00003	Weizenbier	Flasche	1,19	7	AL	01.01.2012	31.12.2013
3	AL00003	Weizenbier	Flasche	1,29	19	BIE	01.01.2014	31.12.9999
4	AL00004	Weizenbier geschmacksfrei	Flasche	1,09	7	LA	01.01.2012	03.12.2011
4	AL00004	Weizenbier alkoholfrei	Falsche	1,09	70	LALL	01.01.2012	05.12.2011
4	AL00004	Weizenbier alkoholfrei	Flasche	1,09	7	AL	01.01.2012	31.12.2013
4	AL00004	Weizenbier alkoholfrei	Flasche	0,99	19	BIE	01.01.2014	31.12.9999
...								

Funktionen zur Erzeugung von „deterministischen Zufallszahlen“

Bevor die zugehörigen Bewegungsdaten erzeugt werden, müssen wir zunächst über Funktionen nachdenken, die interessante Werte und Zuweisungen erzeugen.

Mit Hilfe der schon erwähnten RAND()-Funktion lassen sich zwar sehr gut „echte“ Zufallszahlen erzeugen, diese werden aber bei einer erneuten Berechnung unterschiedlich sein. Gibt es eine Möglichkeit, zufallsartige, aber deterministische Zahlen zu erzeugen? Dafür werden nicht umkehrbare Funktionen benötigt, so dass verschiedene Argumente ähnliche Werte erzeugen können. Periodische Funktionen sind hierfür besonders gut geeignet.

Eine erste Möglichkeit ist die Modulo-Funktion: Diese gibt den Rest bei Division durch den angegebenen Divisor an. Beispiel: $Mod_5(7) = 2$.

Wird nun als Divisor eine Primzahl, z.B. 19, genommen und als Argument sehr große Zahlen, wie Vielfache von 100, so ergeben sich folgende Werte:

Argument	**Rest**
100	5
200	10
300	15
400	1
500	6
600	11
700	16
800	2

Die Funktion ist zwar periodisch ist (ab 2000 wiederholen sich die Ergebnisse), die Werte sind aber einfach vorauszusagen.

Dies ließe sich durch nicht-regelmäßige Argumente verhindern, beispielsweise die Fibonaccizahlen:

Argument	**Rest**
1	1
2	2
3	3
5	5
8	8
13	13
21	2
34	15
55	17
89	13

Eine andere interesante Funktion ist sin(x) mit folgenden Eigenschaften:

- Irrationale Periode (2π): Damit erzeugen alle ganzzahligen Argumente immer unterschiedliche (meist irrationale) Zielwerte. Die Periode kann durch Multiplikation geändert werden.
- Der Wertebereich liegt immer zwischen -1 und 1.
- Sie ist linear genug, um eine gute Verteilung zu erreichen, aber gleichzeitig nicht-linear genug, um absichtliche Verzerrungen zu erzeugen.
- Die Periode ist klein genug, dass große Eingabewerte quasi zufällige Ergebnisse erzielen.

In Kombination mit Multiplikationen, Runden, Betrag usw. lassen sich die gewünschten Ergebnisse erzielen, hier sind beispielsweise die Ergebnisse für die Funktion Round(10*ABS(sin(x)),0), die ganzzahlige Werte zwischen 0 und 10 erzeugt:

Argument	Wert
15	7
30	10
45	9
60	3
75	4
90	9
105	10
120	6
135	1
150	7

Bewegungsdaten erzeugen

Nun können auch Bewegungsdaten erzeugt werden. Da die (selbst erzeugten) Stammdaten bekannt sind, ist auch der Wertebereich der künstlichen Schlüssel (z.B. Produkte zwischen 0 und 100, Kunden zwischen 0 und 250,...) bekannt.

Erzeugen wir nun eine Tabelle mit mehreren 100.000 Zeilen und einem Zeilenindex in Ganzzahlschritten, so wird

Round(100*ABS(sin(INDEX *80)),0)

Werte zwischen 0 und 100 Erzeugen, also die Produktzuordnung.

Round(250*ABS(sin(INDEX *80)),0)

würde die entsprechenden Werte zwischen 0 und 250 (Kunden) erzeugen, allerdings mit exakt demselben Verlauf und derselben Periode, was langweilig wäre.

Eine Änderung der Periode führt zu unterschiedlichem Verhalten:

Round(250*ABS(sin(INDEX *240)),0)

Nun können noch Umsatzwerte generiert werden: Beispielsweise mit der Funktion

Round(600*ABS(sin(INDEX*12)),2)

Die Ergebnistabelle sieht wie folgt aus:

Index	Produkt	Kunde	Umsatz
1	99	236	321,94
2	22	154	543,35
3	95	136	595,07
4	43	243	460,95
5	85	22	182,89
6	62	228	152,29
7	71	171	439,91
8	77	117	590,15

9	54	247	556,09
10	89	44	348,37
...			

Dies ist nur ein kleiner Ausschnitt, weil natürlich auch ein Datum und weitere Stammdatenbeziehungen zu pflegen sind, das Verfahren ist aber analog.

Wenn in diesem Beispiel der Umsatz gleich dem Bruttopreis ist und die MWST zu berechnen ist, deren Satz aber produktspezifisch ist, so kann über den Lookup des Produkts die Berechnung erfolgen.

Zueinander passende Kennzahltabellen werden erzeugt, indem die inhaltliche Berechnungsvorschrift angewandt wird, beispielsweise Kontensalden als Stichtagsgenaue Summe aller Kontobuchungen des Kunden bis zum Stichtag.

Ein solches selbsterzeugtes Testdatenmodell kann auch zu Schulungszwecken verwendet werden, weil damit auch Verarbeitungsverfahren exemplarisch modelliert und umgesetzt werden können.

Kontaktadresse:
Felix Krul

areto consulting gmbh

Schanzenstr. 6-20

51063 Köln

Telefon: +49 (221) 66 95 75-0

Fax: +49 221 66 95 75-99

E-Mail felix.krul@areto-consulting.de

Internet: www.areto-consulting.de