Laufzeitoptimierung bei der Verknüpfung großer Datenmengen Ein Vergleich zwischen MERGE und JOIN

Cerstin Erler
Institut für Arbeitsmarkt- und Berufsforschung
der Bundesagentur für Arbeit (IAB)
Regensburger Straße 104
Nürnberg
cerstin.erler@iab.de

Zusammenfassung

Ziel ist es, für das Datenmanagement großer und sehr großer Datensätze aussagekräftige Handlungsempfehlungen abzuleiten, mit welchem Vorgehen (Prozeduren etc.) die Laufzeit optimiert werden kann und wie die Inputdaten (und Indizes) organisiert sein sollten, um diese schnellstmöglich verarbeiten zu können.

Schlüsselwörter: SAS Base, MERGE, JOIN, PROC SQL, Index, Laufzeit, Performance, große Datenmenge

1 Datenbasis und Server

Verwendet wurden die Integrierten Erwerbsbiografien (IEB) des IAB. Dabei handelt es sich um einen Datensatz, der Forschungsfragen der Volkswirtschaft und Sozialwissenschaft beantworten kann.
Die IEB enthalten prozessproduzierte Daten der Beschäftigtenstatistik, der Bundesagentur für Arbeit und der Grundsicherungsträger nach dem SGB II.
Insgesamt sind aktuell Informationen zu 82 Mio. Personen in 1.827 Mio. Zeilen und 39 Spalten gespeichert. Der Platzbedarf beträgt 427 GB (352 GB Daten und 75 GB Index).

Für die hier beschriebenen Arbeiten steht SAS Base 9.2 unter UNIX/Solaris mit 4 Prozessoren und 32 GB RAM zur Verfügung. Dort ist auch der Speicherort der Daten. Die Größe des Workspace ist auf 1 GB beschränkt. Das Front-End ist der SAS Enterprise Guide 4.2 unter Windows Server 2003.

2 Einleitung

Mit dem zugrunde liegenden sehr großen Datensatz werden verschiedene Varianten des Zusammenspielens zweier Datensätze durchgeführt.

1. MERGE im DATA STEP mit und ohne Verwendung von Indizes
2. JOIN unter PROC SQL mit und ohne Verwendung von Indizes, sortiert und unsortiert.

Als Vorbereitung werden Stichproben der Größe 1%, 2%, 5% 10%, …, 95%, 98%, 99% aus den 82 Mio. Personen gezogen und jeweils nur die Personen-Identifikatoren in der Stichprobendatei behalten.

Eine typische Aufgabe aus der Praxis besteht darin, zu diesen Stichproben die in den IEB enthaltenen Personeninformation zu ziehen. Basierend auf dieser für das Datenmanagement im IAB zentralen Fragestellung erfolgen weitere Auswertungen, so dass häufig täglich mehrere solcher Abfragen laufen.

Trotz guter Infrastruktur müssen die Skripte ressourcenschonend bzgl. Festplattenspeicher und Arbeitsspeicher angelegt sein.

3 SAS-Code

Verglichen werden alternative Vorgehensweisen bei der Ziehung von erwerbsbiografischen Konten zu Personenstichproben unterschiedlicher Größe. Jede der genannten Varianten ist als SAS-Makro kodiert. Damit wird sichergestellt, dass für jede Stichprobengröße der identische Code ausgeführt wird und der Compiler dadurch den gleichen Auftrag erhält.

Die Makros sind so aufgebaut, dass die Daten zuerst vorbereitet werden, wie z.B. sortieren nach Zufall, um später den für den MERGE benötigten Sortierschritt in der Zeitmessung zu dokumentieren. Anschließend erfolgt der Start der Stoppuhr und damit das Füllen der Makrovariable `beginn`[1].

```
%LET beginn = %NRBQUOTE(%SYSFUNC(DATETIME(),10.));
```

Danach folgt der zentrale Verknüpfungsschritt, der auch einen erforderlichen Sortiervorgang enthalten kann. Schließlich wird die Zeitmessung mit Zuweisung einer Makrovariable `ende` beendet, die verarbeiteten Zeilen gezählt und diese zusammen mit der Differenz aus Stopp- und Startzeit in einem Datensatz gesammelt, der außerhalb des Makrocodes am Anfang leer angelegt wurde. Eine übersichtliche Darstellung der einzelnen Messwerte ist im Abschnitt 5 unter Ergebnisse enthalten.

4 Versuchsanordnung

Der unter 3 genannte zentrale Verknüpfungsschritt soll nun genauer beschrieben werden und die Unterschiede der Alternativen detailliert herausgearbeitet werden.

1. MERGE
   Der MERGE verlangt sortierte Input-Daten. Deshalb wird die Stichprobe nach dem Identifikator sortiert und der MERGE über ID mit dem gesamten zentralen Datensatz, der sortiert und indiziert vorliegt, ausgeführt.
2. MERGE ohne Index

Ähnlich zu 1. wird die Stichprobe nach dem Identifikator sortiert. Jedoch wurde der zentrale Datensatz vorher ohne Index gespeichert, nur die Sortierung besteht bereits.

3. SORT + JOIN
   Um gleiche Bedingungen wie beim MERGE zu schaffen, wird hier die Stichprobe nach dem Identifikator sortiert und anschließend erfolgt der JOIN über ID mit dem gesamten Datensatz, der bereits sortiert und indiziert vorliegt.
4. SORT + JOIN (inkl. Optionen)
   Dieser Schritt entspricht genau dem vorherigen, jedoch werden zusätzlich die Optionen _METHOD und _TREE angegeben.
5. JOIN
   Anders als bei den bisherigen Schritten sind alle Sortierungen und Indexes aufgehoben. Der JOIN erfolgt zwischen der unsortierten Stichprobe mit dem unsortierten großen zentralen Datensatz.
6. INDEX + JOIN
   Neu ist in diesem Schritt, dass die Stichprobe mit einem Index versehen wird. anschließend erfolgt wie bei der dritten Variante der JOIN über ID mit dem gesamten Datensatz, der bereits sortiert und indiziert vorhanden ist.

Alle Varianten sind so angelegt, dass die Sortierung und/oder Indexerstellung für den zentralen Datensatz nicht berücksichtigt wird. Das ist bewusst so angelegt, da alle Überlegungen darauf beruhen, einen Datensatz effizient zu speichern, um alle notwendigen Abfragen ressourcenschonend erstellen zu können.
Unsortiert bedeutet, dass eine Sortierung nach Zufallszahl erfolgt ist, ohne bei der gemessenen Dauer berücksichtigt zu werden.
Im Rahmen der Zusammenfassung wird später noch auf die für Sortierung und Indexerstellung benötigten Zeiten eingegangen.

5 Ergebnis

In Tabelle 1 sind die gemessenen Werte (in hh:mm:ss) für die Dauer der beschriebenen Alternativen für die Ziehung von Informationen aus einem zentralen Datensatz zu Stichproben der Größe *rate*. Die für die jeweilige Stichprobengröße schnellste Zeit ist grau hinterlegt.

Eindeutig stellt sich heraus, dass die beiden Varianten des MERGE (Varianten 1 und 2) durchgehend für alle Stichprobengrößen die schnellste Verknüpfung zweier Datensätze erstellen. Favorisiert werden kann auch der JOIN mit einer indizierten Datei (Variante 3), wenn die kleinere der beiden Dateien bis zu 10-15% der Zeilen der Größeren enthält. Interessant ist auch, dass bei der Angabe der SAS-SQL-Optionen _METHOD und _TREE (Variante 4) die Laufzeit für den gleichen Code so lang ist, als wäre keine Indizierung und/ oder keine Sortierung vorhanden.

Tabelle 1: Einzel-Ergebnisse der Zeitmessungen

	1	2	3	4	5	6
rate	MERGE	MERGE ohne Index	SORT + JOIN	SORT + JOIN (inkl. Optionen)	JOIN	INDEX + JOIN
1	01:52:48	01:48:57	01:50:47	02:14:24	01:53:24	07:05:36
2	02:16:20	01:45:17	01:56:05	08:44:07	07:50:08	13:59:23
5	02:10:03	01:47:28	02:12:15	09:42:18	09:09:37	06:30:47
10	01:50:20	01:53:35	01:44:04	09:28:05	Abbruch	08:43:48
15	02:30:48	01:55:06	07:21:46	07:59:54	Abbruch	10:48:14
20	01:56:45	01:53:02	09:47:02	08:47:22	Abbruch	10:31:14
25	01:58:40	02:13:33	09:15:19	13:45:57	Abbruch	Abbruch
30	01:59:29	02:11:58	07:44:06	09:11:35	Abbruch	Abbruch
35	02:03:12	02:29:58	09:31:36	08:09:11	Abbruch	Abbruch
40	03:26:56	02:15:17	07:37:07	08:11:26	Abbruch	Abbruch
45	03:40:43	02:16:44	09:18:16	Abbruch	Abbruch	Abbruch
50	03:26:22	02:30:56	09:12:24	Abbruch	Abbruch	Abbruch
55	02:11:39	02:10:26	08:06:52	Abbruch	Abbruch	Abbruch
60	02:14:01	02:02:28	09:28:37	Abbruch	Abbruch	Abbruch
65	02:26:22	02:58:23	08:05:01	Abbruch	Abbruch	Abbruch
70	03:46:28	02:04:20	08:07:46	Abbruch	Abbruch	Abbruch
75	02:23:29	02:17:40	08:14:58	Abbruch	Abbruch	Abbruch
80	02:24:37	02:57:56	10:33:28	Abbruch	Abbruch	Abbruch
85	02:26:37	02:58:34	08:25:33	Abbruch	Abbruch	Abbruch
90	02:29:55	02:48:43	11:03:28	Abbruch	Abbruch	Abbruch
95	02:32:47	02:45:05	08:44:29	Abbruch	Abbruch	Abbruch
98	04:57:24	02:19:06	10:47:14	Abbruch	Abbruch	Abbruch
99	04:35:05	03:01:12	09:08:57	Abbruch	Abbruch	Abbruch

Ein Abbruch ist bei den beschriebenen Varianten erfolgt, wenn der Workspace (im IAB 1 GB) übergelaufen ist. Das alleinige Auftreten bei den JOIN-Varianten ergibt sich aus der Bearbeitung eines JOINS in SAS: Aus den zu verknüpfenden Tabellen wird zuerst das kartesische Produkt bestimmt, welches als utility-File im Workspace beobachtet werden kann. Erst dann entscheidet sich, welche der Zeilen in das Ergebnis übernommen oder verworfen wird. Das utility-File nahm in den hier erfolgten Durchläufen eine Größe von bis zu 512 GB an, das heißt über 45% größer als die zu verknüpfende zentrale Tabelle. Da das vorliegende Skript so angelegt ist, dass die Ergebnisdatei eine temporäre Tabelle ist und damit im Workspace angelegt wird, behindert das utility-File das Entstehen des Ergebnisses aus dem JOIN.
Wichtig zu erwähnen ist natürlich, dass die Zeitmessungen immer im Normalbetrieb durchgeführt wurden, also einmal mehr und einmal weniger andere SAS-Arbeiten auf dem hier verwendeten Server parallel erfolgten. So ist zu erklären, dass der Anstieg der Verknüpfungsdauer mit wachsender Stichprobengröße schwankt.

6 Zusammenfassung

Unter Berücksichtigung der Dauer für die Erzeugung der Sortierung (hier 06:11:40) oder eines Index (hier 03:51:54) für den zentralen Datensatz ist bei sehr häufiger Verknüpfung mit diesem Datensatz die Speicherung mit Sortierung nach der Verknüpfungsvariable zu empfehlen. Dann wird die schnellste Verknüpfung mit MERGE erreicht.
Hat eine der zu verknüpfenden Dateien häufig eine Größe bis zu 15% des anderen Datensatzes, dann lohnt es sich, das File mit Index auf die Verknüpfungsvariable zu speichern. Hier wird die schnellste Art der Verknüpfung mit JOIN, jedoch ohne Angabe von Ausführungsoptionen, erreicht.

Das Ergebnis für die JOIN-Varianten folgt genau dem Muster, das Michael A. Raithel [2] bei der SUGI 29 darstellt. Dort sind auch weitere Konstellationen für SAS-Codierung beschrieben, die einen Index verwenden und somit zu schnelleren Ergebnissen führen.

Literatur

[1] SAS Online Documentation SAS Base 9.2.

[2] Michael A. Raithel, Creating and Exploiting SAS Indexes, SUGI 29, Hands-on Workshops, Paper 123-29.

ANHANG

Beispiel für den Code der Variante 1:

```
%LET anz = MAX;
/* Leeres File LIB.merge_info_ohne_ind erstellen, um alle
Informationen zu sammeln */

/* IEB ist sortiert nach person_id, Makro zur Ausführung für alle
Stichproben wird erstellt. */

%MACRO M_merge_no_ind(rate, no);

/* Step 1: Vorbereitung */
/* Zuerst die Stichprobe nach Zufall sortieren */
/* Bilde Zufallsvariable */
DATA work.neu_zuf;
     SET LIB.stichprobe_&no.  (OBS = &anz.);
     zufall = RANUNI(0);
RUN;

/* Sortierung nach Zufallsvariable */
PROC SORT
     DATA = work.neu_zuf
```

```
     OUT = work.neu_zuf_s  (DROP = zufall);
     BY  zufall;
RUN;

/* Step 2 */
/* Hier beginnt die Zeitmessung */
%LET beginn = %NRBQUOTE(%SYSFUNC(DATETIME(), 10.));
/* wieder "richtig" sortieren für MERGE */
PROC SORT
     DATA = work.neu_zuf_s
     OUT = work.stichprobe_&no._sort;
     BY   person_id;
RUN;

DATA work.merge_ieb_&no.;
     MERGE     work.stichprobe_&no._sort(OBS = &anz.  IN = in_stpr)
               LIB.&ieb._s              (OBS = &anz.  IN = in_ieb)
     ;
     BY person_id;
     IF in_stpr THEN OUTPUT;
RUN;
/* Hier endet die Zeitmessung */
%LET ende = %NRBQUOTE(%SYSFUNC(DATETIME(), 10.));

/* Infos über die Daten sammeln */
DATA work.merge_info_&no. (KEEP = filename rate dauer);
     SET work.merge_ieb_&no.;
     ATTRIB
     filename  FORMAT = $32.  LABEL = "MERGE o Ind: Filename Konten"
     rate      FORMAT = 4.2        LABEL = "Stichproben-Rate"
     dauer     FORMAT = TIME.      LABEL = "Dauer Konten-Ziehung"
     ;
     filename  = "merge_no_index_ieb_&no.";
     rate      = &rate.;
     dauer     = %EVAL(&ende. - &beginn.);
RUN;

/* Abschließend wird noch die Anzahl der Beobachtungen ermittelt,
die Ergebnisse per PROC APPEND gesammelt, die Zwischenergebnisse
gelöscht, das Makro beendet und für jede der einzelnen Stichproben
aufgerufen. */
```