Reentrant Logic Table Technology

Colin James III, Principal Scientist, CEC Services, LLC, 1613 Morning Dr, Loveland CO 80538-4410, ltt@cec-services.com, C: 719.210.9534 , F: 970.593.1350

Keywords

Reentrant Logic Table Technology, RLTT, Layered Logic Tables, LTT

Abstract

Reentrant logic table technology (RLTT) uses one input to spawn a cascading series of subsequent steps. RLLT may be represented in non RLTT by multiple inputs that initiate the respective steps. Layers of two dimensional logic tables may also effectively emulate RLTT as three dimensional logic tables. This has implications for data warehouse cubes which include the time dimension.

Introduction

Reentrant logic table technology makes use of tasks which link to other tasks. The query of a task in a column position returns switches and those row numbers associated with the switches. Some switches may also refer to other column tasks which when queried return more column tasks. This progression is termed cascading tasks because one task may ultimately invoke many other tasks. The implementation problem with cascading tasks is keeping track of the order of switches which apply to operations on a row number or to another task. Below are the problem conditions.

Logic Table:

Row	**Column Position Number**		
Number	Task 1	Task 2	Task 3
Account 4	Dr	--	--
Account 5	Cr	--	--
Account 6	2	Cr	--
Account 7	--	3	--
Account 8	--	Dr	--
Account 9	--	--	Cr

For task 1 in column position number 1, a query returns the switch debit (Dr) for account 4, the switch credit (Cr) for account 5, and the switch 2 (meaning task 2). The operations are performed sequentially and in that order. The last operation of switch 2 for task 2 is different from the switches Dr or Cr for account operations and is indifferent to the account number associated with it.

For task 2, a query returns the switch Cr for Account 6, the switch task 3, and the switch Dr for Account 8. The difficulty is that after Account 6 is credited, another query for Task 3 follows next before the debit to Account 8 is made.

The solution is to keep track of the order place of switch operations on accounts and for tasks. However, the complexity of this graph order defeats the whole purpose of using logic table technology in the first place for speed and simplicity.

A further implementation problem with cascading tasks is that *recursive* triggers which contain the queries are not supported by the relational database vendors because of the complexity of execution in SQL engines. The only way to implement cascading tasks is to nest in successive levels a separate query for each column task. This again defeats the purpose of using logic table technology by introducing arbitrary complexity.

Layers of Logic Tables

Another representation of the problem of cascading tasks is to think of multiple logic tables in layers.

Logic Table 1:		**Logic Table 2:**		**Logic Table 3:**	
Row	**Column**	**Row**	**Column**	**Row**	**Column**
Number	Task 1 …	**Number**	Task 1 …	**Number**	Task 1 …
Account 4	Dr	Account 4	--	Account 4	--
Account 5	Cr	Account 5	--	Account 5	--
Account 6	2	Account 6	Cr	Account 6	--
Account 7	--	Account 7	3	Account 7	--
Account 8	--	Account 8	Dr	Account 8	--
Account 9	--	Account 9	--	Account 9	Cr

The question again is the order place of switches. For example, beginning at table 2 is the order of the tasks:

account 6 credit, account 9 credit, account 8 debit; or
account 6 credit, account 8 debit, account 9 credit.

While the net effect may be the same, the order of execution is important if the output of account 9 depends upon the input of account 8 or if intermediate reports are required before completion of a task.

Where layered logic tables (LLT) may help is in data warehouse cubes where each logic table corresponds to a certain time constraint. However, layered tables may also be folded into one table consisting of sequential logic tables and indexed as described in James [1998.1].

Conclusion

Any tasks performed by reentry to a logic table may be performed as multiple, sequential, simpler tasks. The business value of RLLT is not apparent as yet.

References

James, C. 2002.3, "Static and Dynamic Driver Triggers", unpublished, CEC Services, LLC, Loveland CO.

James, C. 2002.2, "Additional Information", unpublished, CEC Services, LLC, Loveland CO.

James, C. 2002.1, “Implementation Details for Multiple Billing”, CEC Services, LLC, Loveland CO.

James, C. 2001.2, "Report Accounts [RA] v 1.2 Inventory / Point of Sale", unpublished, CEC Services, LLC, Loveland CO.

James, C., 2001.1, "Report Accounts [RA] v 1.2", unpublished, CEC Services, LLC, Loveland CO.

James, C. 1999.1, “Recent Advances in Logic Tables for Reusable Database Engines”, Proceedings of the American Society of Mechanical Engineers International, Petroleum Division, 75th Anniversary Conference, Energy Sources Technology Conference & Exhibition, Houston, Texas.

James, C., 1998.5, "Multiple and Self-Modifying Logic Tables with Queries", unpublished, CEC Services, LLC, Loveland CO.

James, C. 1998.4, “A Reusable Database Engine for Accounting Arithmetic”, Proceedings of The Third Biennial World Conference on Integrated Design & Process Technology, Vol. 2, pp. 25-30, Berlin, Germany.

James, C., 1998.3, "Competency test for CEC Services, LLC", unpublished, CEC Services, LLC, Loveland CO.

James, C. 1998.2, “Theory and Application of Logic Tables in Relational Database Engines”, Doctoral Dissertation, Pacific Western University, Los Angeles.

James, C., 1998.1, "Ticket Reservations [TR] ver 1.1", unpublished, CEC Services, LLC, Loveland CO.

James, C., 1997.2, "User Documentation", unpublished, CEC Services, LLC, Loveland CO.

James, C., 1997.1, "Logic Table Design for Reports in RA", unpublished, CEC Services, LLC, Loveland CO.