Optimizing Temporal Database using Temporally Homogeneous Characteristics

LALIT GANDHI, RESEACH SCHOLAR, UIET, M.D. UNIVERSITY ROHTAK (HARYANA), INDIA

Abstract: Time-varying features of real time applications cannot be implied utilizing two-dimensional Codd's relational model. Temporal data models save past records utilizing their timestamps characteristics. Various temporal data models intend to express temporally static and dynamic characteristics, utilizing attributes and tuple time stamping. This paper center around the thought of homogeneous and heterogeneous nature of temporally dynamic attributes while outlining temporal data models. By gathering the temporally homogeneous attributes on premise of their homogeneous and heterogeneous nature, memory or space cost required can lessened and execution time can improved.

I. Introduction

Relational model does not monitor past or future database states and time-differing highlights of real life. Real time applications must store time shifting qualities and development data. It should focus around time differing information administration. Commercialy accessible relational databases focus around the customary bivalent Boolean rationale and is insufficient to think about such ongoing data changes. Real life problems and decision-making associated data, is not precise and hence cannot be represented using relational data model only. We require database, which can save past records alongside their timestamps and required data can be retrieved when needed.

II. Need of temporal database

Database, which can keep up past, present and future information, is known as a temporal database. Temporal data put away in a temporal database varies from the information put away in non-temporal database. As, time periods are appended to the temporal information, so temporal database principally consider time measurement to timestamp the information.

Two sorts of time documentations are there in temporal databases-Valid time and Transaction time. Timestamps can be allied with either tuples

or attributes. Timestamps can be connected on particular estimations of attributes (characteristics), utilizes Non First Normal Formof relations (N1NF) or on the complete row of the relations called tuple itself, utilizes First Normal frame (1NF).Both valid time (VT) and transaction time (TT), timestamps can be connected on attributes and tuples.

III. Static and Dynamic Attributes of a Relation

The attributes of a relation which stays same and do not change with the traverse of time are called as temporally static properties. The characteristics of a relation, which change with the traverse of time, are dynamic attributes. Dynamic properties can be temporally homogeneous or heterogeneous. In the event that two or more attributes changes inside same interim of time then they are known as temporally homogeneous and if these attributes change inside different interim of time are called as temporally heterogeneous. Appropriately, corresponding relations are also homogeneous or heterogeneous Relations. Time-differing data is articulated by adding timestamp to the qualities. The distinctive timestamps utilized might be time points, intervals or set of time intervals.

IV. Application area

Temporal database is required for most of real time applications. Grow-up area covered is *financial applications;*itincludes the history of stock markets and share prices. Real time r*eservation system*has also grown-up field, which comprise of airlines reservations, train reservations and hotel bookings etc. Insurance industry is the next bigger sector, which is utilizing the advantages of temporal data, for policy interest calculations, maturity dates etc. Latest growing area is medical sector, which is using temporal database for keeping up medical records. IT industry utilizes the power of temporal database by keeping backup records and archive files. Archive management system is completely based on timelines for archival of files, directories etc.

V. Temporal data models

Enormous part of research done in this field has concentrated on execution of time varying features on the main database. The contemporary examinations created temporal models in view of open source database framework. Around two dozen time varying data models were presented, amid 1990s. Most recent couple of years back once more, number of temporal data models were presented.

Bi-temporal Conceptual Data Model (BCDM): C. S. Jenson and R. T. Snodgrass, utilized the basic characteristics of relational model anproposed Bi-temporal data model. It utilizes the two sorts of time stamping techniques– Valid Time and Transaction Time.

Temporal Functionality in Objects with "Role" models (TF-ORM): TF-ORM is a temporal object oriented data model with utilization of "Role" characteristic. A few "Roles" might be characterized for each class.

Temporal Relational model for overseeing Patient Data (TempR-PDM): Implemented on patients database parts – Entity, Attributes and relationships. This model uses tuple time stamping. The valid time is replenished by activation_start, and activation_end time.

Temporal Data Model of SQL-2011: Microsoft released SQL-2011 in December 2011. Prior to this version, time-varying properties were portrayed by means of use rationale, which was intricate and tedious. The systemtime period is determined by the word SYSTEM_TIME while the identifier for the application-time period is indicated by the user. The integrity constraint isindicated by using PRIMARY KEY utilizing WITHOUT OVERLAPS alternative.

Temporal Data Model of IBM DB2: The implementation of time periods in tables for IBM DB2 is like that in SQL-2011 with the exception of classification of tables, for example, BUSINESS_TIME in DB2. IBM DB2 does not characterize name for the predetermined time period in the PERIOD statement and hence, BUSINESS_TIME keyword is utilized.

Tuple Timestamp Single Relation Data Model: A tuple timestamp single relation data model stores its time varying dynamic characteristics and in addition non-time varying static attributes in a solitary relation. This component brings about data redundancy in the database model amid offbeat updates, as the dynamic characteristics may not differ at the same time. Tuple timestamp single relation data model is represented by key attributes, static characteristics, dynamic traits and timestamp. Every one of these attributes are put away in a solitary relation. The space prerequisites are high because of the excess caused by the tuple time stamping. Accordingly, total cost of the model is high.

Tuple Time stamped Historical Relation Model: In tuple timestamp historical relation data model, two relations are kept up to catch the advancement of objects with traverse of time. One of them stores the present part of the Object (Current Relation) though the other is utilized to store the past occasions of the object at various instances of time (History Relation). A present relation can be separated into four subclasses: key, static characteristic, dynamic attributes and timestamp. Every one of the characteristics of these classes are available in current relation of historical relation data model. A history relation can be separated into three subclasses named key, dynamic qualities or timestamp.On the event of any refresh in current relation, the past part of that occasion is consequently exchanged (Triggers) to the history relation for keeping up the entire history of object. There is a successful decrement in the redundancy of data when contrasted with tuple timestamp single relation model.

Tuple Timestamp Multiple Historical Relation Model: In tuple timestamp multiple historical relational data model, different relations are kept up to catch the development of an object with traverse of time. One of these relation stores the present part of the object which are static in nature though rest of these are utilized to store the dynamic part of the object which differs with the traverse of time. The relations which stores the present part of an object demonstrates the current situation with the object regarding time and speaks to depiction.

Then again, history relations are utilized to catch the past instances of temporally heterogeneous dynamic attributes. Only time varying attributes are caught in the history relations. For putting away non-temporal attributes of a relation, a different static table is utilized which holds the estimation of static characteristics and the estimation of these traits doesn't change amid their lifetime. Onlyinsert or retrieve task is executed on static relation. Then again, a different table is shaped for every unique characteristic and these are utilized to store the estimations of dynamic traits independently. Redundancy of the data diminishes by the utilization of various relation for every temporal attribute.

History relation stores the past part of an object and it holds the estimation of only temporal properties and no non-temporal attributes. On the event of any refresh in current relation, the past part of that occasion is consequently exchanged to the history relation for keeping up the entire history of an object. This inclusion of past viewpoint into the history relation is finished by the utilization of temporal functions like trigger, whichis executed consequently on the event of refresh activity on current relation.

The general memory cost of the tuple timestamp multiple historical relation (TTMHR) data model is very intricate when contrasted with the tuple timestamp historical relation (TTHR) data model. As the quantity of non-temporal attributes builds, the costing of data model as far as space increments and consequently the execution diminishes. This model executes just the dynamic characteristics; the quantity of history and current relations are required to be kept up and space necessity is higher.

VI. Optimizing temporal data model

Tuple Time stamped Multiple Historical Relation Model (TTMHR), if a relation has n-time varying traits, and m non-time varying attributes. The relation is expressed as single relation with m static attributes and all the time varying characteristics of a fundamental relation are decayed into n-relations (for n time varying traits) alongside timestamps regardless of their temporally homogeneous or heterogeneous nature.Homogeneous and heterogeneous temporal nature of the attributes can be utilized, to execute the tuple timestamp. Different historical relations for temporally homogeneous dynamic attributes uses only single temporal relation as these attributes change synchronously. However, usage of temporally heterogeneous dynamic characteristics utilize various relations as these properties changes nonconcurrently, inside same model. Single relation use for temporally homogeneous attributes, will essentially enhance space or memory usage cost. Furthermore, impressive enhancement as far as query execution time for a solitary relation can be uncovered. In this manner, data model will have the capacity to deal with both temporally homogeneous and heterogeneous dynamic properties. By applying only single relation for temporally homogeneous dynamic characteristics, space usage cost will be enhanced, as lesser memory will be required.

VII. Conclusion

Real time applications must store time varying traits and development data. It is request of time to store that data in temporal database, utilizing attributes or tuple timestamping. With traverse of time, size of data increments with the quantity of transactions performed. Temporally dynamic homogeneous and heterogeneous characteristics of a relation can be gathered to store the temporally homogeneous dynamic attributes in a solitary relation rather than number of relations. To fetch the data stored in temporal database, there will be no need of joining different relations, ordering them and it is conceivable to get required data from just single relation. Henceforth, gathering temporally homogeneous credits prompts execution change of temporal data models.

VIII. References

1] Kvet, M., "Temporal data approach performance." New developments in Circuits, systems, signal processing, communications and computers: proceedings of the International Conference Circuits, systems, signal processing, communications and computers (CSSCC 2015), Vienna, Austria. 2015.

[2] Kumar, S., Rishi, R. “A Relative Analysis of Modern Temporal Data Models,” Proc. of 3rd IEEE International Conference on “Computing for Sustainable Global Development”, pp 8361-8365, 2016.

[3] Kumar, S., Rishi, R. “Retrieval of Meteorological Data using Temporal Data Modeling”. Indian Journal of Science and Technology, 9(37), pp. 1-10, 2016. doi: 10.17485/ijst/2016/v9i37/99875.

[4]http://www.postgresql.org/files/documentation/pdf/9.5/postgresql-9.5-A4.pdf.

[5]http://www.ibm.com/developerworks/data/library/techarticle/dm-1204db2temporaldata.

[6] C. Atay, "An attribute or tuple timestamping in bitemporal relational databases," Turkish Journal of Electrical Engineering & Computer Science, pp 4305-4321, 2016. doi:10.3906/elk-1403-39.

[7] Kumar, S., Rishi, R., Kumar, R. “Implementation of Temporal Functionality in Object with Roles Model”. IEEE India International Conference on Information Processing, 12-14 August 2016, Delhi.

[8] Petkovic, Dušan. "Temporal Data in Relational Database Systems: A Comparison." New Advances in Information Systems and Technologies. Springer International Publishing, pp. 13-23, 2016. doi: 10.1007/978-3-319-31232-3_2.

[9] Hu, Y. et al., "Temporal View Maintenance in Wide-Column Stores with Attribute-Timestamping Model." East European Conference on Advances in Databases and Information Systems. Springer International Publishing, 2016. doi: 10.1007/978-3-319-44039-2_17.

[10] Song, S. et al., "Cleaning timestamps with temporal constraints." Proceedings of the VLDB Endowment 9.10, pp. 708-719, 2016. doi: 10.14778/2977797.2977798.

[11] Chomicki, Jan, and Jef Wijsen. "Consistent Query Answering for a temporal Constraints over Temporal Databases." 23rd IEEE International Symposium on Temporal Representation and Reasoning (TIME), pp. 149-156, 2016. doi: 10.1109/TIME.2016.23.

[12] Kumar, S, Rishi, R., et al. “Performance Evaluation of Tuple Timestamp Multiple Historical Relation Data Model”.

[13] M. Pomarlan, "Visibility-based planners for mobile robots capable to handle path existence queries in temporal logic," Advances In Electrical and Computer Engineering, pp. 55-64, Feb. 2014. doi:10.4316/AECE.2014.01009.

[14] Kvet, M. et al., "Complex time management in databases." Open Computer Science 4.4, pp. 269-284, 2014. doi: 10.2478/s13537-014-0207-4.

[15] Kvet, Michal, and Monika Vajsová. "Extended Column Level Temporal System Indexing." European Modelling Symposium (EMS), 2014. doi: 10.1109/EMS.2014.43.

[16] Petković, Dušan. "Performance Issues Concerning Storage of Time-Variant Data." Egyptian Computer Science Journal 38.2, 2014.

[17] M. Kaufmann et al., "Bi-temporal Timeline Index: A data structure for processing queries on bi-temporal data," 31st IEEE International Conference on Data Engineering, pp. 471-482, 2015. doi: 10.1109/ICDE.2015.7113307.

[18] Yang, C., Wang, X., Zhang, M., Zheng, R., Wei, W., & Lou, Y., "Standardization on bitemporal information representation in BCDM," IEEE International Conference on Information and Automation, 2015, pp. 2052-2057. doi: 10.1109/ICInfA.2015.7279627.