MASTER OF COMPUTER APPLICATIONS

SEMESTER 1

RELATIONAL DATABASE MANAGEMENT SYSTEM

Unit 14

Database Application

Table of Contents

1. INTRODUCTION

In the previous unit, we studied XML Query processing and several associated aspects such as XML query languages, approaches for XML query processing, query processing on relational structure and storage schema, and XML DBMS. In this unit, we will cover applications related to database.

An active database system is a technique that allows you to take action automatically to the events that are happening within or outside the database system itself. It includes event driven architecture. The main uses of active database system include statistics gathering, authorisation, security monitoring and alerting.

In this unit, you will study in detail about active databases, temporal database and multimedia database. You will also learn about video database management system (VDBMS) which offers an integrated support for queries on storage management for video, semantic and image query processing.

1.1 Objectives

After studying this unit, you should be able to:

- *Explain the concept of active database*
- *Discuss temporal database*
- *Identify and explain the text and media database*
- *Explain video database management system*

2. ACTIVE DATABASE

Active database do not form two visible classes, but they rather define two ends of database rule languages. In active databases, production style rules are utilised to give automatic implementation of database operations in response to certain events and / or conditions.

Active databases vary from conservative databases in such a way that the active databases identify predefined situations in database and trigger predefined actions when such situations occur. Actions are generally database updates.

2.1 Design Principles for Active Rules

Events that take place triggers the active rules involuntarily. Occurring events can be updating database, initiation of some actions, etc. Functionality provided by means of active databases is included in various commercial packages. The functionality is shown as triggers. There are some actions that can be recognised by certain events automatically. The rules that recognise these actions are believed to be a main enhancement to database systems.

Triggers can be defined as a method which specifies various active rules. The occurrence of triggers took place in the previous versions of SQL specification related to relational databases. At present, triggers are considered as the component of SQL 99 and later standards.

DB2, Oracle, and MS SQL Server are considered as the commercial relational databases. There are various trigger versions available in these databases. However, a lot of research has been performed regarding the appearance of common models used for active databases.

Below are active rules explained for starburst, oracle etc. in detail.

2.2 Starburst

Now, we will provide some cases to demonstrate the process of defining rules in STARBURST. Through this, we are permitted to illustrate the process of writing statement-level rules, as only these kinds of rules are permitted in STARBURST.

The three active rules R1S, R2S, and R3S are as follows:

R_1S: CREATE RULE Total_sal1 ON EMPLOYEE

```
WHEN INSERTED
IF EXISTS (SELECT * FROM INSERTED WHERE Dno IS NOT NULL)
THEN UPDATE DEPARTMENT AS D
SET     D.Total_sal = D.Total_sal +
( SELECT SUM I.Salary) FROM INSERTED AS I WHERE D.Dno - I.Dno)
WHERE         D.Dno IN ( SELECT Dno FROM INSERTED );
```

R_2S:

```
CREATE RULE Total_sal2 ON EMPLOYEE
WHEN UPDATED ( Salary )
IF EXISTS       ( SELECT * FROM NEW-UPDATED WHERE Dno IS NOT NULL )
OR EXISTS ( SELECT * FROM OLD-UPDATED WHERE Dno IS NOT NULL )
THEN UPDATE         DEPARTMENT AS D
SET     D.Total_sal = D.Total_sal +
        ( SELECT SUM ) (N.Salary) FROM NEW-UPDATED AS N
        WHERE D.Dno N.Dno) -
        ( SELECT SUM (O.Salary) FROM OLD-UPDATED AS O WHERE D.Dno = O.Dno
        WHERE         D.Dno IN ( SELECT Dno FROM NEW-UPDATED) OR D.Dno IN (
        SELECT Dno FROM OLD-UPDATED );
```

R_3S:

```
CREATE RULE Total_sal3 ON EMPLOYEE
WHEN UPDATED (Dno)
THEN UPDATE         DEPARTMENT AS D
SET     D.Total_sal = D.Total_sal +
        ( SELECT SUM (N.Salary) FROM NEW-UPDATED AS N
        WHERE D.Dno = N.Dno )
WHERE D.Dno IN ( SELECT Dno FROM NEW-UPDATED
);
UPDATE DEPARTMENT AS D
SET     D.Total_sal = Total_sal -
        ( SELECT SUM (O.Salary) FROM OLD-UPDATED AS O
        WHERE D.Dno = O.Dno )
WHERE         D.Dno IN ( SELECT Dno FROM OLD-UPDATED );
```

Now you can study the rule structure by utilising rule R1S. You can state the name of a rule by using the following statement:

CREATE RULE

That is, this statement states Total_sal1 in case of active rule R1S. A Clause ON defines the relation upon which a rule is stated, that is, EMPLOYEE for active rule R1S.

We use WHEN clause when the events (which are used to trigger a rule) are required to be specified. The IF clause, which is not compulsory, is utilised to state the conditions which are required to be verified. Lastly, we use THEN clause to state the actions that are to be performed. These actions are usually SQL statements (one or more).

The particular events in STARBURST which triggers the rules are considered as update commands for SQL. These commands include: DELETE, INSERT, and UPDATE. STARBURST document makes use of these particular keywords.

Changed tuples can be referred by a method which is incorporated by a rule designer.

STARBURST document makes use of the following keywords:

- INSERTED
- DELETED
- NEW-UPDATED
- OLD-UPDATED

By means of these keywords, you can refer to different transition tables related to these four keywords respectively.

Clearly, the availability of different transition tables depends on triggering events. These tables can be referred by the writer of a rule when it is required to write the condition of the rule as well as its action parts.

In case of statement-level semantics, designer of the rule can just refer to transition tables as well as the rule is triggered merely one time. Thus the rules should be written in a different way as compared to the rules for row- level semantics. As a particular insert statement may comprise several employee related tuples, it should be checked if any recently inserted

employee related tuples is connected to department. In case of R1S, the following condition is required to be checked:

EXISTS (SELECT * FROM INSERTED WHERE Dno IS NOT NULL)

Also if the condition appears to be true, it signifies the execution of the action. The action performs the updation in single statement DEPARTMENT tuple(s) linked to recently inserted employee(s). This is done by totalling their income to Total_sal characteristic of every linked department.

Since various recently inserted employees might link to similar department, it is required to utilise the SUM aggregate function for ensuring that every salary of the department is added.

The active Rule R2S appears to be same as the active rule R1S. However, UPDATE function is used to trigger this rule. Update function is used for updating the salaries of employees.

You can trigger R3S by updating Dno attribute which is included in EMPLOYEE. This signifies that one or more employees are shifted from one department to another.

Since R3S does not include any specification, action is performed every time on the occurrence of triggering event. The action which is performed equally updates both the old as well as new departments related to the employees who are reassigned. This is done by totalling the salaries of employees to Total_sal related to every new department. Also the salaries of all old departments are deducted from Total_sal.

Delayed consideration is being utilised in the implementation model for active rules in STARBURST. Specifically, every rule which is triggered inside a transaction is kept in a set. We call this set as a conflict set. This set is not judged for assessment of conditions as well as execution till the completion of a transaction. This is done providing its COMMIT WORK command.

In STARBURST, client is allowed to begin the rule consideration while performing a transaction. This is done by means of PROCESS RULES command. As there is a requirement to evaluate multiple rules, it is essential to provide an order between the rules.

In STARBURST, rule declaration syntax permits ordering between rules. This is done to educate the system regarding the order which considers a set of rules.

Furthermore, transition tables, that is, INSERTED, DELETED, etc. include the total result of every operation inside the transaction. The transaction influenced every table, since the transaction applies multiple operations to all tables.

2.3 Oracle

The model which is utilised to state active database rules is known as Event-Condition-Action (ECA) model. ECA model contain rules which has 3 elements.

(a) Events generate rules: Events in this rule are basically operations for DB updation. In this rule, events are openly implemented to the DB.

(b) The second element decides whether the generated rule should be executed or not? As the rule action is generated in the first step, there is an option in the condition. If any condition is not defined, the rule action has to be executed at least once when the event happens. If a condition is defined, firstly it should be evaluated and if the result is true then only the rule action is executed.

(c) Third element explains the rule action: The rule action is basically a set of SQL statements. But sometimes it acts as a DB transaction or an external program which is executed automatically.

For example: Figure 14.1 shows a company database which can be utilised for Active rules. There is an EMPLOYEE having Name, Security Number

(Ssn), Salary, Department Number (FK to DEPARTMENT) and Supervisor (A recursive FK to EMPLOYEE)

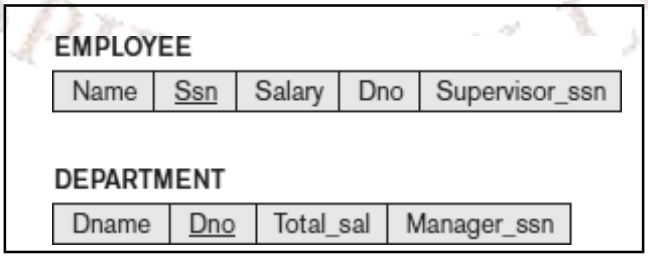

Fig 14.1: Example of Active Rule

In this example we can say that Null value is permitted for Department Number (Dno) which shows that some employee is not assigned to any department. In DEPARTMENT, there is a

Department Name (DNAME) , Department Number (Dno), Total Salary (Total_sal), Manager (Manager_ssn FK to EMPLOYEE)

An active rule is applied on the Total_sal because it has a crucial job of maintaining salaries of each employee. It should maintain a correct value. We will first see the events that are responsible for change in Total_sal:

1. New Employee
2. Increments in salary
3. Moving one employee to another department
4. Removing employee

In the first event, we have to re-calculate the Total_sal if a new employee is added and assigned to department. Therefore Dno (Department Number) does not have Null value. Same thing goes for 2nd and 4th events. In second and fourth event we have to determine the employee whose salary is incremented. For third event, we will always perform an action to maintain the value of Total Salary accurately, hence no condition is needed (the action is forever executed).

In event first, second and fourth the action updates the total salary for the department of an employee to show the added, incremented and erased salaries. In third event, two actions are needed, first for updation of total salary of past department and second for updating total salary in present department.

Now you can see the Active rules for R1, R2, R3 and R4 events:

(a) R1:
```
CREATE TRIGGER Total_sal1
AFTER INSERT ON EMPLOYEE
FOR EACH ROW
WHEN ( NEW.Dno IS NOT NULL )
        UPDATE DEPARTMENT
        SET Total_sal = Total_sal + NEW.Salary
        WHERE Dno = NEW.Dno;
```
R2:
```
CREATE TRIGGER Total_sal2
AFTER UPDATE OF Salary ON EMPLOYEE
```

```
        FOR EACH ROW
        WHEN ( NEW.Dno IS NOT NULL )
        UPDATE DEPARTMENT
        SET Total_sal = Total_sal + NEW.Salary – OLD.Salary
        WHERE Dno = NEW.Dno;
R3:     CREATE TRIGGER Total_sal3
        AFTER UPDATE OF Dno ON EMPLOYEE
        FOR EACH ROW
                BEGIN
                UPDATE DEPARTMENT
                SET Total_sal = Total_sal + NEW.Salary
                WHERE Dno = NEW.Dno;
                UPDATE DEPARTMENT
                SET Total_sal = Total_sal – OLD.Salary
                WHERE Dno = OLD.Dno;
                END;

R4:     CREATE TRIGGER Total_sal4
        AFTER DELETE ON EMPLOYEE
        FOR EACH ROW
        WHEN ( OLD.Dno IS NOT NULL )
                UPDATE DEPARTMENT
                SET Total_sal = Total_sal – OLD.Salary
                WHERE Dno = OLD.Dno;
(b) R5:         CREATE TRIGGER Inform_supervisor1
                BEFORE INSERT OR UPDATE OF Salary, Supervisor_ssn ON EMPLOYEE
                FOR EACH ROW
                WHEN ( NEW.Salary > ( SELECT Salary FROM EMPLOYEE
                WHERE Ssn = NEW.Supervisor_ssn ) )
                inform supervisor(NEW.Supervisor_ssn, NEW.Ssn );
```

The CREATE TRIGGER statement states an active rule (trigger name) Total_sal1 for R1. AFTER clause says triggering rules will happen after the triggering rule events happens.

Events triggered- Inserting new employee are defined after AFTER clause. The ON clause explains relation in which rule is defined—FOR EACH ROW explains the rule that will be triggered for every row for at least one time.

WHEN keyword is optional- It is used for any condition that need to be certified after every rule is triggered and before execution of action. Finally, action to be taken is defined as PL/SQL block with more than one SQL statements or calls to run procedures which are external.

The above mentioned active rules show numerous characteristics of active rules. Firstly, the events defined for rules to be triggered are SQL commands: INSERT, DELETE, UPDATE.

2.4 DB2

DB2 is a relational model database server developed by IBM. There are three DB2 products that are very similar, but not identical: DB2 for LUW (Linux, Unix, and Windows), DB2 for z/OS (mainframe), and DB2 for iSeries (formerly OS/400). The DB2 LUW product runs on multiple Linux and UNIX distributions, such as Red Hat Linux, SUSE Linux, AIX, HP/UX, and Solaris, and most Windows systems.

The Syntax is:

DB2-trigger: CREATE TRIGGER <trigger-name>
{BEFORE I AFTER} < trigger-event> ON
<table-name>
[REFERENCING <references>] FOR
EACH {ROW I STATEMENT}
WHEN (<SQL-condition>)
<SQL-procedure-statements>
<trigger-event> : INSERT I DELETE I UPDATE
[ON <column-names>]
<reference> : OLD AS <old-value-tuple-name> I

NEW AS <new-value-tuple-name> I OLD_TABLE AS <old-value-table-name> I NEW_TABLE AS <new-value-table-name>

2.5 Application of active database (Active DB)

You can now see the possible active database applications. Noticeably, first application permits announcement of condition to happen. Such as, an active DB might be utilised to observe the physical property of an organisational heating system.

Another application is the physical property taking records direct from physical property sensors. Active rules are triggered when physical property records are added from the condition that assures if the physical property increases from risk level. It will effect an action to happen

Another application of active rules is that it entails integrity constraints by explaining the types of events that may affect the constraint to be despoiled as well as evaluates the suitable conditions that assures constraints are despoiled or not.

Hence the composite application constraints which are also known as business rules might be enforced that way.

For example in DB of a college, 1 rule might notice Average of Grade (AG) of pupils when a new grade is inserted and may aware the consultant if AG of a pupil falls below a certain value. Another rule might assure that subject requirements are fulfilled before permitting a pupil to apply for a subject. Another application is the basic maintenance of derived data. Maintaining consistency of occurred views and operations updating.

SELF-ASSESSMENT QUESTIONS – 1

1. Events _____ by the rule action that automatically identifies.
2. The main events that are defined for rules triggered are the common SQL statements in STARBURST. (True/ False)
3. What does the CREATE TRIGGER statement states?
   a) trigger name
   b) trigger class
   c) trigger function
   d) trigger variable
4. The model which is utilised to state active database rules is known as _____ model.

3. TEMPORAL DATABASE

Temporal databases are used to record time-referenced data. Basically majority of the database technologies are temporal. For example:

- Record keeping function (inventory administration, medical-record and personnel,)
- Financial function (banking, accounting and portfolio organisation)
- Scientific function (weather monitoring)
- Scheduling function (project organisation, hotel, airline and train reservations).

All these functions trust on temporal databases.

Temporal databases are best suited for the applications where information is to organize on time constraints. Therefore, temporal database set a good example to demonstrate the requirement for development of a combined set of concepts for the use of application developers. The framing (objective, design, coding, interface and implementation) of temporal database is designed by application developers and designers.

There are numerous applications where time is an important factor in storing the information. For example:

- Insurance, to keep record of accidents and claims.
- Healthcare, to maintain patient histories.
- Reservation systems, to check the reservation and availability of seats in train, airline, hotel, car rental, and many more places.
- Scientific databases, where experiments outcome need to be stored along with the time that when it was carried out.

In case of temporal applications, even the two instances utilised might be simply expanded. For example, in COMPANY database, it may be desirable to keep PROJECT, JOB and SALARY histories of all the employees.

It can be applied to UNIVERSITY database as well, to store the grade history of STUDENT. The details about the YEAR, SEMESTER, COURSE and each SECTION are also included in the database.

Actually, it can be easily concluded that some temporal information is stored by many of the database applications. But it is also observed that many users try to ignore temporal feature as it adds complexity to the applications.

Different forms of Temporal databases

Temporal database can be distinguished into various types depending upon the different notions of time, i.e., valid time and transaction time. Valid time is the time for which a fact is true in the real world. Transaction time is the time at which a transaction was made.

A **historical database** stores data with respect to valid time.

A **rollback database** stores data with respect to transaction time.

A **bitemporal database** stores data with respect to both valid time and transaction time.

As an example, a **snapshot database** store only a single state of the real world, usually the most recent state in the context of both valid time and transaction time. A TimeDB database stores the history of data with respect to both valid time and transaction time.

SELF-ASSESSMENT QUESTIONS – 2

5. Temporal databases are the technique which record _____ data.
6. Financial function of temporal database include:
   a) Banking
   b) accounting
   c) portfolio organisation
   d) all of the above

Activity 1

Explain how temporal databases include all database applications to organise their information.

4. MULTIMEDIA DATABASE

Multimedia databases facilitate the users to store as well as generate query for retrieving multimedia information. This information can demand for:

- Documents (like articles/ books/journals)
- Images (like drawings/pictures)
- Video clips (like newsreels/ movies/home videos)
- Audio clips (like speeches/ phone messages/songs),

The primary type of database query generally tries to locate multimedia sources comprising of particular objects of interest. Such as, one user wants to locate all the video clips regarding a specific person, say Michael Jackson in a video database.

Another scenario may be as of someone willing to retrieve video clips grounded on specific activities like, video clips where a soccer goal is scored by a certain player or team. These types of queries are mentioned as content based retrieval, as they retrieve information based on a certain activity/object from the multimedia sources.

To make this retrieval fast the multimedia database must make use of some model to index and manage multimedia sources grounded on the contents. But identifying the contents of multimedia sources is lengthy and difficult task. To accomplish this task two approaches can be followed as defined below:

- Based on automatic analysis of the multimedia sources. It is done to recognise the contents mathematical characteristics.
- Based on manual identification of the objects and activities of interest in each multimedia source. And later on depending on this information; index the sources.

SELF-ASSESSMENT QUESTIONS – 3

7. The main type of database query those are required to include locating multimedia sources consists of certain objects of interest. (True/ False)
8. Identifying the contents of multimedia sources is an easy task. (True/ False)

5. VIDEO DATABASE MANAGEMENT

The video database management system comprises of open source system. The video database management system research group came up with the extensions and adaptations that were required to support full functionality of the database.

The extensions of the key database consists of store management for video, video pre-processing for representation of content and indexing, image and semantic-based query processing, real time buffer management.

5.1 Storage management for video

Massive amount of data having real-time restrictions are carried by the buffer managers as well as video database storage. In Video database management, the database buffer zone and the streaming zone are split by the buffer pool.

Numerous page requests by segment allocation for huge streaming requests done by means of stream manager are taken care of by the extended buffer management. Interchanging information, which is used to direct buffer caching, is performed by means of interface. This interface takes place among stream manager as well as buffer manager.

Storage manager provides long duration for the execution of essential video operation in addition to accomplish the requests for real-time as well as for non real-time. Video database management technique is utilised to take care of expanded hierarchies for storage that assist real-time, translucent access to disk, buffer, as well as tertiary storage.

Caching level series on disk storage as well as buffer makes the process of accessing better for data which is referenced most often. Also the server used for tertiary storage helps in managing access to tertiary data. This is done to make the tertiary data accessible straightforwardly to VDBMS.

In tertiary storage, the important items are recognised by the use of committed disk partition. Cache disk manager performs the maintenance of these items. Also he creates the report of these items.

5.2 Video Pre-Processing for Content Representation and Indexing

Metadata, in addition to index video content take a disk space to a greater extent as compared to the video itself. This is because the data having high- dimensional characteristic are gathered for all video frames. Also the data are combined for all video shots.

In executing and optimising the characteristic-based queries, there occur problems in severe indexing as well as searching. These problems are presented by the extent of metadata in addition to their storage in database as multi-dimensional vectors.

The research set included in VDBMS increased the potentiality of Shore's indexing. This is done by including the implementation of GiST v2.0. Also the predator's layer of query processing is changed for utilising the index of the Shore or GiST.

VDBMS inserted vector ADT which is to be utilised by every field. The following statement is implemented:
CREATE GSR INDEX <table> <?eldname> <table>

for generating an example of GiST SR-tree for utilising all fields as access path in the queries which are characteristic-matching.

High-dimensional characteristic vectors which are generated by means of visual characteristic extraction as well as utilised in the image resemblance searches are managed by means of multi-dimensional structure of indexing.

At present an interface is being built for supporting add-in constituent for the methods of indexing. This is done to implement, check, and compare different methods of indexing within VDBMS.

Query processor was changed widely for managing new indexing plan. Also it was changed for managing latest operators concerned with video query in addition to their incorporation into the plan of query execution.

Query processing in case of Video DBMS is required to consider video techniques in addition to operators in producing, and carrying out query plans. The process of providing nearest queries to the access path (which is high dimensional) performs image resemblance search.

In case of image resemblance queries having multiple features, users usually display sample image. In addition, database is queried for images. Determination of outcomes should take place as per a combined order of similarity.

5.3 Image and Semantic-Based Query Processing

The Video Database Management System toolkit employs image and semantic-based query processing for separating raw videos into shots then combining the shots obtained from visuals and semantic signifiers and filing the content of video for searching.

Image and semantic-based pre-processing algorithms identifies the limitations of video scene which will cut the video into significant shots. This can be done by utilising a procedure that calculates colour changes in bar graph.

Video shots are processed to take out MPEG7-supporting low-level visual characteristic signifiers, spatial and temporary segmentation, camera movement categorisation, illustration key frames, and additionally semantic notations of domain proficient. The video features and content are confined in Video Database Management System along with physical metadata.

The new method of presenting video indices in an XML is same as MPEG7 multimedia indices signifiers.

5.4 Real-Time Buffer Management

Content based search is supported by continuous media servers and main memory get recovery. After that it stores the demanded media prior to transfer it to the user.

Several studies notified the investigations of buffering policies for media streaming. Chang and Garcia-Molina initiates a prefetching plan for memory efficiency which depends on determining the time displacement among the requesting prefetching.

In media streaming, dynamic buffer allocation reduces the memory requirement. The essential practicalities of buffer management for delay- sensitive multimedia data, defines alterations required by DBMS to sustain multimedia data.

Buffer allocation is very goal-oriented utilised for various DB (database) workloads in which targeted objective/goal is allocated for every workload.

Caching parts of media streams enhances streaming performance in two ways which may be referred in the near future:

1. It decreases numerous addresses to the disk storage
2. It minimises waiting time for initialising streaming.

You know accurate caching conclusions are very hard to make. Before initial reference, replacement of data will be barred and not referenced for maximum time in optimum pre fetch and replacement methods.

In the normal streams there is a trouble in policy faith on information which is usually not available. However, there is an inherent connection between query processing and streaming.

Streaming options are generally grounded on query outcomes, and buffer manager employ this type of link to prefetch as well as cache pages awaited for reference. An effective buffer management policy should be used which takes feedback from the search engine and helped in making more accurate replacement and prefetching decisions.

To predict about future video streaming requests top graded query outcomes of query processor should be utilised in which a weight function further determines candidates for caching.

VDBMS is an option that can achieve better caching of media streams by incorporating knowledge of the query as well as streaming elements and further minimises the initial latency and reduces disk I/O.

SELF-ASSESSMENT QUESTIONS – 4

9. Video database management method is used to take care of storage _____ hierarchies.
10. In Video database management, the database _____and the streaming zone are split by the buffer pool.
11. Video shots are processed to take out MPEG7-supporting low-level visual characteristic signifiers, (True/ False)
12. Which processing is used by VDBMS video pre-processing toolkit to partition raw video?
    a) Semantic based query processing
    b) Distributed query processing
    c) Both
    d) None of the above

Activity 2

The Video Database Management System toolkit employs image and semantic-based query processing to separate raw video into shots. Analyse this statement and make a note of it.

6. SUMMARY

Let us recapitulate the important points discussed in this unit:

- By the occurrence of events, active rules must be automatically triggered. The events may be like database accessed or updated.
- ECA (Event-Condition-Action) model is used to state active database rules.
- The application can periodically insert in the database the temperature reading records directly from temperature sensors, and active rules can be written that are triggered.
- The main type of database query those are required to include locating multimedia sources consisting of objects of interest.
- Massive volumes of data with real-time limitations are carried by the video database storage and buffer managers.
- Video pre-processing toolkit in VDBMS utilises semantic processing and image for creating various shots by partitioning the raw video streams.

7. GLOSSARY

- ***Active database:*** An active database system is a technique that allows you to take action automatically to events
- ***Active rules:*** Events that take place, automatically triggers the active rules.
- ***Dynamic buffer allocation:*** Minimises the memory requirement for concurrent media streams.
- ***ECA:*** Event-Condition-Action.
- ***Multimedia databases:*** It gives features that let users to store as well as query various kinds of multimedia information
- ***VDBMS:*** Video Database Management System

8. TERMINAL QUESTIONS

1. What is active database? Explain what are the active rules for oracle?
2. What is the difference between temporal and multimedia database?
3. Explain the storage management for video.
4. Discuss real time buffer management.
5. Write short notes on:

   (a) Starburst

   (b) Oracle

   (c) DB2

9. ANSWERS

Self-Assessment Questions

1. Triggered
2. True
3. (a) Trigger name
4. Integrity
5. Time-referenced
6. d. all of the above
7. True
8. False
9. Extended
10. Buffer zone
11. True
12. a. Semantic based query processing

Terminal Questions

1. Active database do not form two visible classes, but somewhat they define two ends of database rule languages. Refer Section 2 for more details.
2. Temporal database works on time-referenced data and multimedia database let users to store and query various types of multimedia information. Refer Section 3 and 14.4 for more details.

3. Massive volumes of data with real-time limitations are carried by the video database storage and buffer managers. Refer Section 5 for more details.
4. Content based search is supported by continuous media servers and main memory get recovery which further buffer to store the requested media streams before transferring them on to the user. Refer Section 5 for more details.
5. DB2, Oracle, and STARBURST are considered as the commercial relational databases. There are various trigger versions available in these databases. Refer Section 2 for more details.

10. REFERENCES

Reference:

- Ramakrishnan, R. & Gehrke, J. (2003) *Database Management Systems,* Third Edition, McGraw-Hill, Higher Education.
- Rob, P. & Coronel, C. (2006)*Database Systems: Design, Implementation and Management*, Seventh Edition, Thomson Learning.
- Silberschatz, Korth & Sudarshan (1997) *Database System Concepts,* Fourth Edition, McGraw-Hill
- Navathe, E. (2000) *Fundamentals of Database Systems,* Third Edition, Pearson Education Asia

E-reference

- portal.aauj.edu/.../database/delphi_database_application_developers_... retrieved on May 14, 2012.
- deep.yweb.sk/dbs/p63-paton.pdf retrieved on May 14, 2012.
- www.cs.arizona.edu/~rts/pubs/LNCS639.pdf retireved on May 14, 2012.