22CSA506-ADVANCED DBMS

Unit - 5

INTELLIGENT DATABASES

Active Databases

❖ Active Databases

- Active Database is a database consisting of set of triggers.
- These databases are very difficult to be maintained because of the complexity that arises in understanding the effect of these triggers.
- In such database, DBMS initially verifies whether the particular trigger specified in the statement that modifies the database) is activated or not, prior to executing the statement.

Active Databases

- If the trigger is active then DBMS executes the condition part and then executes the action part only if the specified condition is evaluated to true.
- It is possible to activate more than one trigger within a single statement.
- In such situation, DBMS processes each of the trigger randomly.
- The execution of an action part of a trigger may either activate other triggers or the same trigger that Initialized this action.
- Such types of trigger that activates itself is called as ‘recursive trigger’.
- The DBMS executes such chains of trigger in some pre-defined manner but it effects the concept of understanding.

Active Database Systems

- An integrated facility for creating and executing production rules from within a database system.
- A typical database production rule:

  when event

  if condition

  then action

Active Database Systems (Cont.)

- Powerful and uniform mechanism for:
  - Constraint enforcement
  - Derived data maintenance
  - Alerting
  - Authorization checking
  - Version management
  - Resource management
  - Knowledge management

Oracle

- Supports general-purpose triggers, developed according to preliminary documents on the SQL3 standard.
- Actions contain arbitrary PL / SQL code.
- Two granularities: row-level and statement-level.
- Two types of immediate consideration: before and after.
- Therefore: 4 Combinations:

  BEFORE ROW
  BEFORE STATEMENT
  AFTER ROW
  AFTER STATEMENT

Syntax

<Oracle-trigger> ::= **CREATE TRIGGER** <trigger-name>

{**BEFORE | AFTER}** <trigger-events>

ON <table-name>

[[**REFERENCING** <references>]

FOR EACH ROW

[WHEN (<condition>)]] <PL/SQL block>

<trigger event> ::= **INSERT | DELETE | UPDATE**

[OF <column-names>]

<reference> ::= **OLD AS** <old-value-tuple-name> |

NEW AS <new-value-tuple-name>

Trigger Processing

1. Execute the BEFORE STATEMENT trigger.
2. For each row affected:
    - (a) Execute the BEFORE ROW trigger.
    - (b) Lock and change the row.
    - (c) Perform row-level referential integrity and assertion checking.
    - (d) Execute the AFTER ROW trigger.
3. Perform statement-level referential integrity and assertion checking.
4. Execute the AFTER STATEMENT trigger.

Example Trigger in Oracle: Reorder Rule

```
CREATE TRIGGER Reorder
AFTER UPDATE OF PartOnHand ON Inventory
WHEN (New.PartOnHand  < New.ReorderPoint)
FOR EACH ROW
DECLARE NUMBER X
 BEGIN
                SELECT COUNT(*) INTO X
                FROM PendingOrders
                WHERE Part = New.Part;
        IF X=0
        THEN
                INSERT INTO PendingOrders
                VALUES (New.Part, New.OrderQuantity, SYSDATE)
        END IF;
 END;
```

INTELLIGENT DATABASES

❖ Applications

- An intelligent database is a full-text database that employs artificial intelligence (AI), interacting with users to ensure that returned items (hits) contain the most relevant information possible.
- This is in contrast to a traditional database, which is searchable only by keywords and verbatim phrases connected by Boolean operations such as AND, OR, and NOT.
- Intelligent database technology is in its infancy, and is evolving as AI becomes more advanced.

INTELLIGENT DATABASES

❖ Design Principles for Active Rules

Problems:

- Complex interactions due to cascading of triggers.
- Unknown side effects of inserting, deleting and modifying rules.
- Recursive trigger: A trigger TR1 is recursive when an application updates table T1, which fires trigger TR1 updating table T1.
- Nested triggers: If a trigger changes a table on which there is another trigger, the second trigger is then activated and can then call a third trigger, and so on. Maximum number of cascading allowed in Oracle is 32; When it exceeds 32, all database changes as a result of original SQL are rolled back.

Temporal Databases

- A temporal database stores data relating to time instances. It offers temporal data types and stores information relating to past, present and future time.
- Temporal databases could be uni-temporal, bi-temporal or tri-temporal.

- More specifically the temporal aspects usually include valid time, transaction time or decision time.
- Valid time is the time period during which a fact is true in the real world.
- Transaction time is the time at which a fact was recorded in the database.
- Decision time is the time at which the decision was made about the fact.

TSQL2

- TSQL2 is a modification and extension of SQL-92.
- The functionality of user-defined time support in SQL-92 is enhanced.
- This required replacing the DATETIME and INTERVAL types with alternative timestamp types.

- SQL-92 was the third revision of the SQL database query language.
- Unlike SQL-89, it was a major revision of the standard.
- Aside from a few minor incompatibilities, the SQL-89 standard is forward-compatible with SQL-92.

Deductive Databases

- Deductive databases are an extension of relational databases which support more complex data modeling.
- A deductive database is a database system that can make deductions (i.e. conclude additional facts) based on rules and facts stored in the (deductive) database.
- Datalog is the language typically used to specify facts, rules and queries in deductive databases.
- Deductive databases have grown out of the desire to combine logic programming with relational databases to construct systems that support a powerful formalism and are still fast and able to deal with very large datasets

Recursive

- When there is a relationship between two entities of the same type, it is known as a recursive relationship.
- This means that the relationship is between different instances of the same entity type.

For Example:

- An employee can supervise multiple employees. Hence, this is a recursive relationship of entity employee with itself.
- This is a 1 to many recursive relationship as one employee supervises many employees.