Examrace: Downloaded from examrace.com [https://www.examrace.com/]
For solved question bank visit doorsteptutor.com [https://www.doorsteptutor.com] and for free video lectures visit Examrace YouTube Channel [https://youtube.com/c/Examrace/]

NET, IAS, State-SET (KSET, WBSET, MPSET, etc.), GATE, CUET, Olympiads etc.: The Programming Language Modula-2

Get unlimited access to the best preparation resource for competitive exams : get questions, notes, tests, video lectures and more [https://www.doorsteptutor.com/]- for all subjects of your exam.

The developments in software engineering, as described above, have led to the emergence of a new class of programming languages over the past decade, which provide sophisticated facilities for data abstraction, modularity and concurrency. Two of the most important of these languages are Ada (Ichbiah et al, 1979) and Modula-2 (Wirth, 1977 and 1982) . A comparison of these languages highlights striking similarities, particularly in the constructs they provide for modular programming and data abstraction. Concurrency is also provided by both languages, though at a rather lower level in Modula-2. Thus the two language are very comparable, if one is primarily concerned with the extent to which the principles of modern software engineering may be applied (Pomherger, 1984) .

A great advantage of Modula-2 over Ada is that it is a small language. This small size has led to the development of efficient Modula-2 compilers for a wide variety of computers ranging from desk-top microcomputers to mainframes. Also, in addition to its support for modularization and data abstraction, the language contains a range of low-level programming features which make it suitable for systems programming as well as applications programming. As described, implementing database systems requires a multilevel approach, ranging from the organization of physical storage at the low level to providing high-level user views. Modula-2 supports such a multilayered approach to software development by providing separate compilation of modules and strong type and interface checking. These facilities offer great advantages for maintaining integrity and security in database systems.

Thus Modula-2 is a language which is suitable both for the implementation of database management system, and for the development of application programs. It is also an ideal language for interacting with an existing DBMS, where a suitable module interface has been provided.

Relational Algebra and Relational Language

The relational algebra is a set of operators which can be used to define new relations from existing ones. Using these operators one may manipulate and query the database in a homogeneous manner. There are five operators from which others can be defined.

1. UNIION RuS Take the union of the two relations.
2. CARTESIAN PRODUCT Rx S

3. DIFFERENCE R-S
4. PROJECTION R [X] Project R on the attribute list X.
5. SELECTION R: F Select the topples from R which satisfy the formula F. This formula is defined by atomic formula (attribute comparison operator constant or attribute) which may be combined using logical operators (AND, OR NOT)

From these five operators we define: If JOIN R * S Construct a new relation by taking tuples from R and the corresponding tuples from S. The correspondence is expressed by comparison between common attributes. If we. Consider R (X, X2, _____ *) and S (YV Y2, … Yn) and if Xt and Yj are attributes constructed on the same domian, we have:

R * S = (RxS) : Xi = Yj which means that the join is equivalent to a Cartesian product followed by a selection.Instead of the equality operator, we have. Etc. Relational algebra can be used to express queries and updates.

One interesting aspect is that relational DBMSs offer a unique, set-oriented data definition and manipulation language. This kind of language is based on relational operators which provide a minimal power. This power is minimal in the sense that, for instance, it cannot express numeric calculations. This is the reason why relational DBMSs provide language which are complete with regards to relational algebra but which provide many other functions.In the same direction, Codd shows that relational algebra is equivalent to first order predicate calculus.

Several relational languages have been defined for different relational DBMSs. Table 3.2 shows the evolution of some of these systems. Two type prototypes arise from database research: IBM's System R and its associated language SQL (Structured Query Language) , and INGRES developed at Berkeley University with the QUEL language.

SQL is a relational language which has also been implemented in several DBMSs. It is neither a pure algebra language nor a predicative language. The basic SQL statement is the qualification block which gives as its result a new relation:

SELECT < attribute list >

FROM < relation (s) >

[WHERE < conditions (s) >]

Centralized and Distributed Relational DBMSs

As we have seen, a relational query expresses only what the user wants to extract from the database and not how to do it.In an SQL statement there is no reference to data access paths as there was for network DML operations. Furthermore, relational language are dynamic which means that at any time it is possible to define new relations and/or views, or to modify the schema, or to define or delete access paths without modifying the application programs. This feature is called data independence.

The interactive interface allows the user to type queries and updates, receiving the answers on the screen. He/she can also issue queries on the database catalogs which are particular relations describing all the schema information. Relational languages can use either a linear

syntax such as SQL or a more graphical form such as Query By Example (QBE) where several empty tables are displayed on the screen and the user fills the table columns in order to express queries and updates.

It is also possible to access the relational DBMS through application programs written in some high-level programming language (e. g. PL ‘1 or COBOL) .In this case, relational statements are embedded into the host language and a recompilation is necessary. For instance, SQL and PL’ 1 can be mixed up and one of the main problems to solve is to establish the correspondence between program variables and database objects. Also, it is necessary to provide ad-hoc mechanisms in order of process sets which come from the database in answer to a given query. For this, we find jn SQL + PL⁄1 a cursor mechanism which is illustrated below.In PL⁄1 programs SQL statements are indicated by a ‘$’ sign in order to be replaced at precompile time by DBMS calls.

Going into the internal structure of a relational DBMS, one may find several software modules. For instance, System R has two layers, namely the Relational Data System (RDS) and the Research and Storage System (RSS) . The RDS is composed with the SQL precompiled and is also responsible for query optimization, authorization control and catalog management. The RSS is more or less a low-level network DBMS, managing disk space, ensuring concurrency control and recovery through a transaction manager. The INGRES system is composed with four modules: PI: User interface, P2: Request analysis and modification, P3: Database control and P4: Database modification and recovery control.

A DBMS allows data manipulation through primitive operations which can be grouped to form database transactions. For instance, a debit-credit transaction consists of:

1. get account value X from the database
2. subtract S from X
3. put into the database new X value
4. get account value Y from the database
5. add S to Y
6. put into the database new Y value

In order for the database to remain consistent, either all of these actions must be executed or none of them. This leads to the concept of an atomic transaction. A transaction is the unit of concurrency and recovery control. SQL, for instance, provides the following statements for transaction control:

- BEGIN TRANSACTION: Initialize a transaction
- END TRANSACTION: Consider that all the SQL statements from BEGIN TRANSACTION constitute an atomic transaction
- RESTORE TRANSACTION: Roll back all the actions made by a transaction since its beginning.