~~Databases~~ 2000

p6q8
JKMM
p13q9

Question 2

Describe the basic architecture of the ODMG standard for Object Data Management.

[~~8~~ 7 marks

In what ways do these proposals enable database management to be integrated with Object-Orientated Distributed Programming?

[4 marks

What support is provided for transactions? What locking modes are available, and how are they used by the database runtime system?

[4 marks

The query language OQL is recognized as a standard by the Object Management Group (OMG). To what extent is it similar to SQL, and in what ways does it differ?

[~~7~~ 5 marks

Question 2 Solution

ODMG standard is based around an Object Data Model, which establishes types for object data, based on a complete system for type generation. Base types include standard items such as Real, Integer, Boolean and Character, as well as more specialised ones such as date and time representations associated with SQL.

A distinction is made between immutable literals and mutable objects. Literals have fixed values. Collection type generators can be associated with both objects and literals; sets, bags, lists, arrays and dictionaries are supported. Mutable collections are objects. Structures defined by a finite set of (name, type) pairs are also supported. In the case that the types are

Question 2 Solution scalar base types the Object Model subsumes the relational model through the mapping of relations <...> to bag<struct<...>>.

Methods are only represented at the interface level. The ODMG Object Model is PL independent, and method implementations must be defined within the context of a specific OOPL (typically C++, Smalltalk or Java).

In addition to the OM the ODMG standard proposes a DDL — ODL, Object Definition Language — and an DML — OQL, a query language. OQL is compatible with the query subset of the relational DBPL standard SQL.

Question 2 Solution ctd)

Database Management is supported through standard tools for schema maintenance (ODL) and a high-level functional query language (OQL). The latter includes query definition features that can fill the role of VIEWs. In this way data independent management of object data is provided.

In addition the ODMG Object Model allows persistent objects stored in databases to be bound to particular OOPL, through run-time system extensions to manage databases, transactions, locking modes etc. In this way the facilities offered by persistent programming languages can be extended to support schema management in a language independent fashion.

Question 2 Solution ctd) THIRD PART

Application programs running against ODMG databases must open each database before object data can be accessed. By default a transaction object is created at the first such access; the style of concurrency control and resource management is up to the implementation. The default suggested is 2PL with granularity at object level.

Lock modes supported are Read, Write and Upgrade. R/W locks are taken out as the system default; U locks must be managed explicitly by the application program.

Application programs must close transactions by issuing an explicit COMMIT or ABORT (via run-time system method invocations supported by the relevant ODMG language binding).

Question 2 Solution (cont'd)

OQL is a functional query language ONLY, but it includes support for method invocations, and these can have side effects. There is no support for schema definition in OQL, unlike SQL which has separate sublanguages which support DDL and DML functionalities.

SQL queries are based on the relational model, whose type system does not support higher-level constructs (i.e. tuples have scalar components, and within SQL only aggregates of tuples are supported). The ODMG Object Model allows for a complete system of type generation, and objects of any type may be interrogated through OQL.

Question 2 Solution (ctd)

SQL syntax and semantics define a set of ad hoc constructs sufficient for many purposes, but there are irritating restrictions and it is often necessary to consult the small print. Any query that can be expressed in SQL may also be expressed in OQL, usually with equivalent syntax; in addition method invocations may be evaluated during query processing, which greatly extends the expressive power of the language. Implementers of OQL do not face an easy job, since method implementations are dependent on a particular programming language binding, which raises problems both of performance and of security.