SOLUTION NOTES

Databases 2002 Paper 5 Question 8 (GMB)

(*a*) [Bookwork] The relational model of data was introduced by Codd in 1970. Codd proposed that database systems should present the user with a view of data organised as tables called relations. Behind the scenes there might be a complex data structure that allowed efficient processing of queries, but the user of the relational system would not be concerned (or aware) of this storage structure. Consequently, queries could be written in a very high-level language.

The relational model provides a simple way to represent data: as a two-dimensional table called a relation. Attributes of a relation serve as names for the columns of a relation. The rows of the relation (other than the header row containing the attributes) are called tuples. Relations are sets of tuples, so a tuple can not appear more than once in a relation. The name of a relation and the set of attributes for a relation is a called the schema for that relation.

(Another definition of a relation R, is that it is a subset of the cartesian product of the domains that define R.)

The relational model requires that each component of a tuple is atomic; that is of an elementary type (integer, char, string, etc.). (This is, of course, first normal form.)

(*b*) (*i*) The Entity Integrity Constraint states that no primary key value can be NULL.

(*ii*) A set of attributes S in relation R_1 is a Foreign Key of R_1 if
– The attributes S are the primary key attributes of another relation R_2 (which need not be distinct from R_1), and
– For every tuple t in R_1, then the values of the attributes S is either NULL, or occurs as a value for some other tuple in R_2.

A Referential Integrity Constraint between two relations is that if a tuple in one relation refers to a tuple in another relation, then it must refer to an *existing* tuple in that relation. The conditions for a foreign key given above specify a referential integrity constraint between two relation schemas R_1 and R_2.

(*iii*) Semantic Integrity Constraints are a large class of constraints that apply to the actual data values that can be stored in a database. For example, "the maximum number of hours that a lecturer can spend marking scripts is 10" (sic).

(*c*) (*i*) A functional dependency on a relation R holds if any two tuples agree on attributes $A_1, \ldots, A_n$ then they agree in another attribute B. We write this as $A_1 \ldots A_n \rightarrow B$. If a set of attributes determine more than one attribute, $B_1 \ldots B_k$, say; then we adopt the shorthand $A_1 \ldots A_n \rightarrow B_1 \ldots B_k$

(*ii*) A relation R is in BCNF iff: whenever there is a non-trivial dependency $A_1A_2 \ldots A_n \rightarrow B$ for R, it is the case the $\{A_1, A_2, \ldots, A_n\}$ is a superkey for R.

Equivalently, the left side of every nontrivial functional dependency must contain a key.

(*iii*) 3NF is a slight relaxation of BCNF:

A relation R is in 3NF iff: whenever there is a non-trivial dependency $A_1A_2 \ldots A_n \rightarrow B$ for R, either $\{A_1, A_2, \ldots, A_n\}$ is a superkey for R, or B is a member of some key.

(*iv*) From the definitions it is clear that BCNF is stricter than 3NF meaning that every relation in BCNF is also in 3NF.