Database Management - May/June 2016
Model Answers

1. (a) i. The cardinality of the Supplies relationship type is many-to-many. (**1** mark)

   ii. Both Supplier and Part participate optionally in the Supplies relationship. (**2** marks)

   iii. The relation schema would be Supplies(Sno,Pno,Price) with primary key (Sno,Pno). There would be foreign key constraints for each of Sno and PNo in Supplies, referencing the primary keys of Supplier and Part, respectively. (**6** marks)

   (b) i. One of Code $\rightarrow$ Level or Name $\rightarrow$ Level is redundant. For example, the second is redundant because we can derive it from Name $\rightarrow$ Code and Code $\rightarrow$ Level. (**3** marks)

   ii. {Code,Textbook} and {Name,Textbook} are keys. Using the closure algorithm, once we have Code in the closure, we can add Name and Level. Similarly if we start with Name. (**4** marks)

   iii. One BCNF decomposition is as follows. The LHS of the first FD (Code) is not a superkey for {Code, Name, Level, Textbook}, so decompose into {Code, Name} and {Code, Level, Textbook}. The LHS of the second FD (Code) is not a superkey for {Code, Level, Textbook}, so decompose into {Code, Level} and {Code, Textbook}. The final 3 relation schemas are therefore {Code, Name}, {Code, Level} and {Code, Textbook}. (**6** marks)

   (c) The operators = and != can only be used with normal attribute values, which NULL is not. NULL is never equal nor unequal to any value (including NULL). Hence we need special syntax to check whether the stored value is the special NULL value or not. (**3** marks)

2. (a) i. The constructs are known as aliases or tuple variables. (**1** mark)

   ii. A correlated subquery is one which refers to a term in an outer query; in this case, the comparison `L.person=V.person` refers to the `Visits` relation in the outer query. (**2** marks)

   iii. The query returns those person and store pairs such that the person visits the store but the store does not sell any items liked by the person. (**5** marks)

   (b) i. For the closure of AB, we start with AB. From the first FD, we can add C. From the third FD, we can add D, to give $ABCD$. Now the second FD allows us to add E, giving $ABCDE$ as the answer. (**3** marks)

   ii. AG is a superkey, since the last FD allows us to add B using the closure algorithm. Now that we have AB, we can add CDE as above, yielding all attributes. Neither A nor G is a key, so AG is a key. (**3** marks)

   (c) The four steps are: (1) Find a canonical cover G of F, (2) form a relation schema from each FD in G, (3) remove any schema which is a subset of another, and (4) if no schema contains a key for U, then add a schema whose attributes are a key for U. (**4** marks)

(d) Update anomalies can be insertion, deletion or modification anomalies. Insertion and deletion anomalies arise when we cannot insert or delete a tuple because of a not-null constraint (e.g. for a primary key). Modification anomalies arise when updates violate an FD whose left side is not the primary key. (Any two answers will do.) (**5** marks)

(e) Null values are ignored by aggregation operators. GROUP BY assigns all null values to the same group. (**2** marks)

3. (a) i. Attributes `s_id` and `m_code` in `Results` would both be foreign keys, referencing the primary keys of `Student` and `Module`, respectively. These constraints would ensure that every tuple inserted into Result would have `s_id` and `m_code` values which already appeared in `Student` and `Module`, respectively. (**6** marks)

ii. The query returns the student name, module name and mark for modules taken by students whose names end in 'Jones'. (**5** marks)

iii. The view is not updateable because there is more than one relation in the FROM clause. The system would not have values for `s_id` and `m_code`. Even if 'A.N. Other' and 'Databases' already occur in Student and Module, respectively, the names are not keys and so the inserted names might refer to a different student and module. Hence the system would have to use null values for `s_id` and `m_code`. This would not be allowed since both are primary keys. (**6** marks)

(b) The `while` loop uses the `fetch()` method to fetch the next row in the answer as an array. This is output as an HTML table row. The `for` loop outputs the cells of a row, each one as an HTML `td` element. It uses `columnCount()` to find out the number of columns in each row. The variable row`[i]` is the value of the i'th column of the current row. (**8** marks)

4. (a) The term "atomically" refers to the fact that transactions should execute to completion or else appear not to have executed at all. As an example, consider a banking application on relation Accounts(acctNo, balance) where we want to transfer £100 from account 123 to account 456. The transaction might first ensure there is at least £100 in account 123. In step 2, it might then deduct £100 from account 123. In step 3, it might add £100 to account 456: If there is a failure after step 2 and before step 3, then £100 would have been lost. (**8** marks)

(b) Three limitations are that (1) all data is value-based - all relationships are expressed through common values, (2) data must always be represented in "flat" (first normal form) relations, and (3) for some applications, performance is not fast enough. Some alternatives are: object-oriented, object-relational, and Nosql databases such as key-value stores, XML databases, document stores and graph databases. (**7** marks)

(c) i. Whenever the same $artist$ appears, he or she always plays the same $role$. (**2** marks)

ii. There is at least one $artist$ who appears twice, each time associated with a different $song$. (**2** marks)

iii. Because an $artist$ may have different $role$s on the same $song$. (**2** marks)

iv. Because an $artist$ may have the same $role$ on many $song$s. (**2** marks)

v. Because the same $role$ on a $song$ (e.g., singer) can be played by many $artist$s. (**2** marks)

5. (a) The three levels of abstraction are the physical level, the logical level and the view or external level. Physical data independence allows the physical layer to be changed without affecting the logical layer. Growth independence allows structures to be added to the logical layer without affecting the view layer. But the view layer may not be shielded from deletions at the logical layer. (**5** marks)

(b) i. A design is dependency preserving if the satisfaction of all functional dependencies can be checked "locally", i.e., by considering each individual relation in turn. (**2** marks)

ii. Third normal form can guarantee that dependencies are preserved. (**1** mark)

iii. If the dependency $AB \rightarrow C$ were not preserved, the database system would have to join the relations in which the attributes A, B and C appear, each time an update to one of them occurred. (**2** marks)

(c) The actions are CASCADE or SET NULL. When a primary key value is deleted or updated, CASCADE will delete/update all rows containing the corresponding foreign key value; SET NULL will instead replace the foreign key value with a null value. (**5** marks)

(d) Give a short explanation of what is meant by each of the following terms:

i. First normal form requires that attribute values in relations are atomic or "simple", e.g., not sets of values.

ii. A weak entity type is one that does not have sufficient attributes to form its own primary key. It's existence depends on another entity type.

iii. Lossless join is the property that the natural join of a decomposition of a relation r should return exactly r.

iv. When a transaction commits, all its updates are made permanent in the database.

v. Dynamic SQL refers to the ability to construct an SQL query in a host programming language at runtime.

(**10** marks)

