1. (18 points) Of the following statements, identify all that hold about XSLT

   A. Because XSLT is all about altering document structure, it is not possible to have an XML Schema for XSLT

   > **Solution:** A is false: XSLT does have an XML Schema because XSLT documents are XML documents

   B. XSLT epitomizes the imperative style of programming

   > **Solution:** B is false: XSLT is not about changing variables representing program state as in, say, Fortran or Cobol

   C. In XSLT, if a template is specified with parameters, values must be explicitly specified when that template is invoked

   > **Solution:** C is false: parameter values are empty node sequences by default

   D. The deep copy of an element (expressed appropriately in XSLT) fails to terminate whenever there are cycles, and that can happen in the more complex XML documents

   > **Solution:** D is false: no cycles in XML documents as such

   E. In XSLT, the default behavior for any text node that is encountered is to copy it to the output

   > **Solution:** E is true: that is the effect of the built-in template (if not overridden)

   F. Each XSLT template must specify the unique node on which it applies

   > **Solution:** F is false: a template would generally apply to more than one node in a document

   G. When an XSLT template matches a node, that node is treated as the context node for determining any output and further actions of the given template

   > **Solution:** G is true:

   H. If XSLT didn't include the copy-of construct, we could achieve its effect through copy and the remaining constructs

   > **Solution:** H is true:

   I. It is possible to write XSLT stylesheets that fail to terminate on some inputs

   > **Solution:** I is true:

2. (32 points) Of the following statements, identify all that hold about XML keys, integrity constraints, and other aspects of relating XML to databases:

   A. In mapping XML to tables, sometimes you would create a table with no columns

Solution: A is false: impossible in the relational model; also notice that we are not trying to store one XML document instance in a relational DBMS, but a family of XML documents

B. In mapping XML to tables, sometimes you would be forced to create a table that has no primary or candidate key

Solution: B is false: impossible in the relational model: there is always a superkey

C. If we map a relational table into an XML document, we are forced to take the risk of allowing duplicate tuples even if that was not our intent in the original table

Solution: C is false: we can state unique constraints to prevent duplicates among tuples mapped to XML elements

D. The unique element limits us to one field subelement

Solution: D is false: no such limitation for unique

E. Since XML documents may have arbitrarily many repetitions, any mapping of XML to tables without duplicates is impossible

Solution: E is false: we can introduce additional columns to distinguish different copies of an XML element

F. Since document order of elements in XML documents is significant and the ordering of tuples in the relational model is not significant, mapping of XML documents to tables is possible only if we know that document order is not significant in the particular case at hand

Solution: F is false: we can introduce additional columns to capture, if necessary, the ordering information from an XML document

G. This course advocates building a conceptual model (e.g., using UML) for the desired documents before creating XML Schema representation for them

Solution: G is true:

H. The data-centric view is better than the document-centric view for sharing well-structured information

Solution: H is true: the cleaner the structure the better the data-centric view is

I. The document-centric view is better than the data-centric view for storing information for subsequent access to establish regulatory compliance

Solution: I is true: in general, the explicit representation facilitates establishing regulatory compliance

J. A tuple-generating element is an element in an XML document schema that is mapped to a table in a relational schema

Solution: J is true: the "contents" of the TGE become the tuples in that relation

K. XML databases can naturally support key integrity constraints but not domain-specific constraints in general

Solution: K is false: use XQuery or SQL/XML queries to state additional constraints

L. In general, when we map an XML schema to a relational schema using one of the simple tuple-generating element approaches, we might not be able to map an XML key to a relational key constraint in any of the resulting relations

Solution: L is true: each relational key is limited to one table but an XML key can refer to subelements of the element on which it is asserted

M. A major practical motivation behind the interest in SQL/XML is the huge installed based on relational databases

Solution: M is true:

N. SQL/XML provides mappings between SQL types (such as date) and XML Schema

Solution: N is true:

O. If an SQL/XML query output any XML, what it outputs is a well-formed XML document

Solution: O is false: the output is often a forest; only in special cases is it a single document or tree

P. XML cannot be used to express sell and buy bids because bids are based on money, not on describing data

Solution: P is false: XML can be used to represent any information item