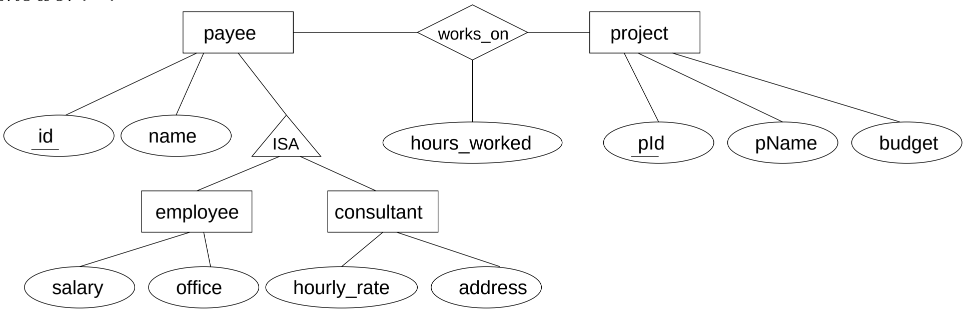
Database Systems June 2002
Sample Answers (with questions)

1. A company payroll database contains information about people who are paid (payees), projects, and what projects people work on. There are two kinds of payee: employee and consultant. Here is a relational schema for the database.

   project(<u>*pId*</u>, *pName*, *budget*)
   payee(<u>*id*</u>, *name*)
   employee(<u>*id*</u>, *salary*, *office*)
   consultant(<u>*id*</u>, *hourly_rate*, *address*)
   works_on(<u>*id*</u>, <u>*pId*</u>, *hours_worked*)

   (a) Draw an E-R diagram that represents the database and captures the information above. Underline the attributes in your diagram that are keys or parts of keys. [5 *points*]

   Answer: :



   (b) The *payee* entity set can be translated into the SQL DDL

```
CREATE TABLE payee ( id   CHAR(9),
                     name CHAR(40),
                     PRIMARY KEY (id) )
```

   Similarly translate the *works-on* and *employee* tables, specifying foreign keys (the syntax for foreign key is `FOREIGN KEY (...)  REFERENCES` *R*`(...)`. Use whatever types you think are appropriate for domains. [5 *points*]

   Answer:

```
CREATE TABLE works-on ( id CHAR(9),
                        pId CHAR(7),
                        hours_worked INTEGER,
                        PRIMARY KEY (id, pId),
                        FOREIGN KEY (id) REFERENCES payee,
                        FOREIGN KEY pId REFERENCES project )

CREATE TABLE EMPLOYEE (id CHAR(9),
```

```
salary REAL,
office CHAR(20),
PRIMARY KEY (id),
FOREIGN KEY (id) REFERENCES payee )
```

(c) Write SQL queries for the following.

i. The names of *employees* who worked on the "Widget" project. [5 *points*]

Answer:

```
SELECT name
FROM employee, payee, works_on, project
WHERE employee.id = payee.id
AND employee.id = works_on.id
AND works_on.pId = project.pId
AND project.pName = "Widget"
```

ii. The names of *payees* who have worked on all projects. [5 *points*]

Answer:

```
CREATE TABLE notworked AS  //employees and projects
  SELECT id, pId           //such that employee has not
  FROM payee, project      //worked on project
  DIFFERENCE
  SELECT id, PId
  FROM works_on ;

CREATE TABLE worked_all AS
  SELECT id
  FROM payee
  DIFFERENCE
  SELECT [DISTINCT] id
  FROM notworked;

SELECT name
FROM payee
WHERE payee.id = worked_all.id
```

2. (a) For each of the following, state whether it is true or false and briefly justify your answer.

   i. From a given initial state, any serialisable schedule will leave the database in the same state. [3 *points*]
   Answer: False. Two transactions, executed in different order could produce different states (examples below.)

   ii. Two-phase locking guarantees serialisability. *Answer*: True. There's a theorem to that effect. [3 *points*]

   iii. Two-phase locking guarantees no deadlock. False. Easy to construct a counter-example. [3 *points*]
   Answer: False

(b) Consider the following transactions

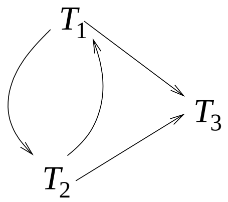
T_1:	T_2:	T_3:
read (X);	read (X);	read (X);
$X := X + 3$;	$X := X + 5$;	$X := X * 2$;
write (X)	write (X)	write (X)

   i. Given that the initial value of X is 0, write down all possible values of X that result from serialisable executions of T_1, T_2 and T_3. [6 *points*]
   Answer: 8, 11, 13, 16.

   ii. Write down a non-serialisable schedule of T_1, T_2 and T_3. [4 *points*]
   Answer: $R_1(X), R_2(X), W_1(X), W_2(X), R_3(X), W_3(X)$

   iii. What is the value of X that results from this schedule? [2 *points*]
   Answer: 10.

   iv. Draw the conflict graph for this schedule. [4 *points*]
   Answer:



(c) Given the functional dependency set

$$\{A \to B, B \to CD, C \to AE, DE \to A, EF \to G\}$$

   i. Give a candidate key for $ABCDEFG$ (a candidate key is a minimal set of attributes that determines $ABCDEFG$). [4 *points*]
   Answer: AF

   ii. Is your answer to 2(c)i unique? If not, give another candidate key. [3 *points*]
   Answer: No. BF

3. (a) Briefly distinguish between the terms *well-formed* and *valid* when applied to an XML document. [3 *points*]

Answer: *Valid* describes conformance to a DTD; *well-formed* does not.

(b) Consider the following DTD:

```
<!ELEMENT db (r,s)>
<!ELEMENT r (rt*)>
<!ELEMENT s (st*)>
<!ELEMENT rt (a,b,c)>
<!ELEMENT st (d,e,f)>
...
```

All undefined tags are `#PCDATA`.

i. Write down an appropriate SQL DDL to store XML data conforming to this DTD. Assume `CHAR(10)` for `#PCDATA`. [4 *points*]
Answer:

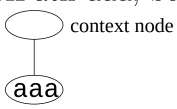
```
CREATE TABLE r ( a CHAR(10), b CHAR(10), c CHAR(10) )
CREATE TABLE s ( d CHAR(10), e CHAR(10), f CHAR(10) )
```

ii. What property or properties of the XML data would your relational implementation fail to capture? [3 *points*]
Answer: Order (of the `rt` elements within `r` and the `st` elements within `s`.)

(c) The following statements, of the form $p_1 \subset p_2$, assert that, for *any* XML document and *any* context node, the set of nodes identified by XPath expression p_1 is included of the set of nodes identified by p_2. In each case state whether the inclusion is true or not. If it is false, and only if it is false, give an example of an XML document and context node (you may draw a tree diagram) for which p_1 identifies a node that p_2 does not.

i. `aaa` $\subseteq$ `(aaa|bbb)` [3 *points*]
Answer: True.

ii. `aaa` $\subseteq$ `aaa/bbb` [3 *points*]
Answer: False. In the following figure the node labeled `aaa` is returned by the XPath `aaa`, but not by `(aaa/bbb)`.



iii. `aaa/bbb` $\subseteq$ `aaa//bbb` [3 *points*]
Answer: True

iv. `aaa` $\subseteq$ `//aaa` [3 *points*]
Answer: True

v. `.//aaa` $\subseteq$ `aaa//aaa` [3 *points*]
Answer: False. The context node is returned by `.//aaa` but not by `aaa//aaa`

(d) Given that the file `xml.xml` contains the xml document

```
 <a>
   <b> <c> C1 </c> <c> C2 </c> <d> D1 </d> </b>
   <b> <c> C3 </c> <d> D2 </d> </b>
</a>
```

What does the following XQuery query produce?

```
<answer>{
   for $x in document("xml.xml")
     for $y in $x/b/c
     return <b> {$y} <e> {$x/b/d} </e> </b>
}</answer>
```

[5 *points*]

Answer:

```
<answer>
  <b>
    <c> C1 </c>
    <e> <d> D1 </d> <d> D2 </d> </e>
  </b>
  <b>
    <c> C2 </c>
    <e> <d> D1 </d> <d> D2 </d>  </e>
  </b>
  <b>
    <c> C3 </c>
    <e> <d> D1 </d> <d> D2 </d>  </e>
  </b>
</answer>
```