# Information Organization Exercises Part 2 (Information Retrieval)

## S-1

### Description



### Solution

DAC: A means of restricting access to objects based on the identity of subjects and/or groups to which they belong. The controls are discretionary in the sense that a subject with certain access permission is capable of passing that permission (perhaps indirectly) on to any other subject.

MAC: Subjects and objects each have a set of security attributes. Whenever a subject attempts to access an object, an authorization rule enforced by the operating system kernel examines these security attributes and decides whether the access can take place. Any operation by any subject on any object will be tested against the set of authorization rules (aka policy) to determine if the operation is allowed.

1. MAC provides access based on levels while DAC provides access based on identity.
2. DAC is more labor intensive than MAC.
3. DAC is more flexible than MAC.
4. MAC access can only be changed by admins while DAC access can be provided by other users.

## S-2

### Description



### Solution

A Trojan horse attack can be designed to accomplish any number of goals, but typically the intent is either pecuniary gain or spreading mayhem. For example, upon bringing the infected file onto your hard drive, the Trojan horse program might locate and send your bank information to the developer. The only limit to what a Trojan horse attack can accomplish is dependent on the limits of the developer's imagination and talent.

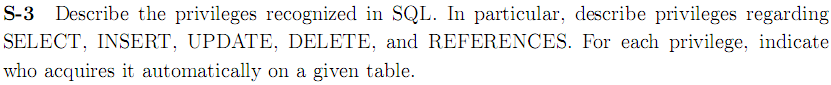
Example:

In DAC, there are a high level user A and a low level user B in the system, and a table T which only A can read. Suppose B is malicious and give a Trojan horse program to A which on the surface does some useful work. Now A runs the program, but without the notice of A, the program reads T and writes the content to another table T’ which B can read. Thus the information in T is leaked to unauthorized user B.

In MAC, if T’ has high security level, then B cannot read it; if it has low security level, the Trojan horse program, which has the same high security level as A, cannot write it.

## S-3

### Description



### Solution

SELECT: Can read all columns (including those added later via ALTER TABLE command).

INSERT (col-name): Can insert tuples with non-null or non-default values in this column. INSERT means same right with respect to all columns.

UPDATE: Can change the data of one or more records in a table. Either all the rows can be updated, or a subset may be chosen using a condition.

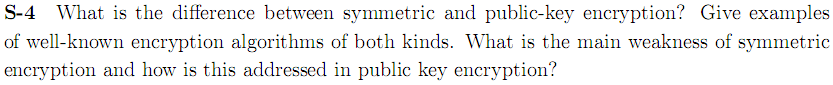
DELETE: Can delete tuples.

REFERENCES (col-name): Can define foreign keys (in other tables) that refer to this column.

Owner (Creator or Administrator) can automatically get privilege on a given table. User who was gave privilege through GRANT OPTION by a privileged user.

## S-4

### Description



### Solution

**Differences**: Symmetric encryption is a single shared, private key between communicating nodes. There is only 1 key involved.

Public Key encryption requires a pair of keys; a public and a private key for exchanging data in a secure manner.

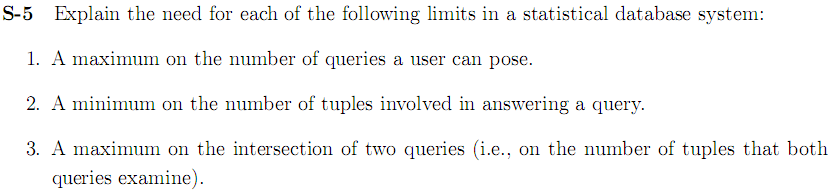
**Examples**: Symmetric encryption: DES, AES. Public Key encryption: RSA.

**Weakness**: Encryption key = decryption key; all authorized users know decryption key. Sharing the secret key in the beginning is a problem in symmetric key encryption. It has to be exchanged in a way that ensures it remains secret.

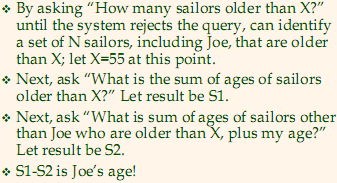
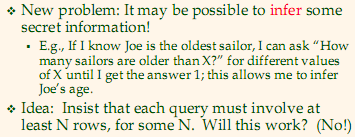
Public key encryption makes use of two keys: a public key and a private key. The public key is made publicly available and is used to encrypt messages by anyone who wishes to send a message to the person that the key belongs to. The private key is kept secret and is used to decrypt received messages.

## S-5

### Description



### Solution



## T-1

### Description



### Solution

**Atomicity (原子性)** requires that each transaction is "all or nothing": If one part of the transaction fails, the entire transaction fails, and the database state is left unchanged.

The **consistency (一致性)** property ensures that any transaction will bring the database from one valid state to another. Any data written to the database must be valid according to all defined rules, including but not limited to constraints, cascades, triggers, and any combination thereof.

The **isolation (隔离性)** property ensures that the concurrent execution of transactions results in a system state that could have been obtained if transactions are executed serially.

**Durability (耐久性)** means that once a transaction has been committed, it will remain so, even in the event of power loss, crashes, or errors. In a relational database, for instance, once a group of SQL statements execute, the results need to be stored permanently.

**Schedule** (or history) of a system is an abstract model to describe execution of transactions running in the system. Often it is a list of operations (actions) ordered by time, performed by a set of transactions that are executed together in the system.

A **dirty read** occurs when a transaction is allowed to read data from a row that has been modified by another running transaction and not yet committed.

A **non-repeatable read** occurs, when during the course of a transaction, a row is retrieved twice and the values within the row differ between reads.

**Serializable schedule**: A schedule that is equivalent to some serial execution of the transactions.

**Cascading abort**: A single transaction abort leads to a series of transaction rollback. Strategy to prevent cascading aborts is to disallow a transaction from reading uncommitted changes from another transaction in the same schedule.

## T-2

### Description



### Solution

Strict Two-phase Locking (Strict 2PL) Protocol:

Each Xact must obtain a S (shared) lock on object before reading, and an X (exclusive) lock on object before writing.

All locks held by a transaction are released when the transaction completes.

If an Xact holds an X lock on an object, no other Xact can get a lock (S or X) on that object.

## T-3

### Description



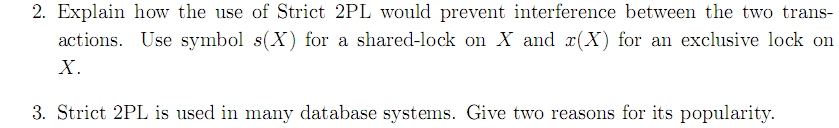
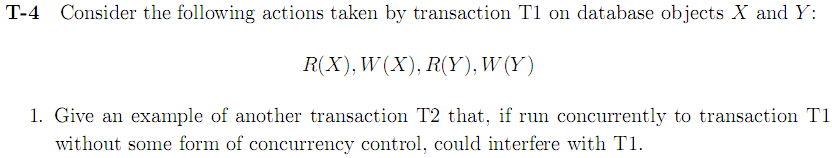
### Solution

A **phantom read** occurs when, in the course of a transaction, two identical queries are executed, and the collection of rows returned by the second query is different from the first.

Yes it can.

## T-4

### Description

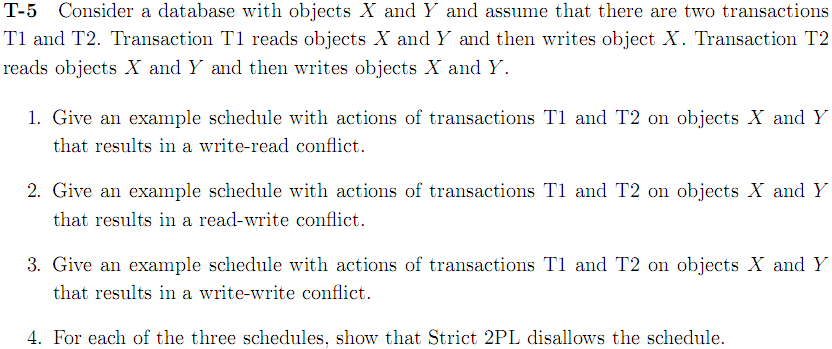


### Solution

1. If the transaction T2 performed W(Y) before T1 performed R(Y), and then T2 aborted, the value read by T1 would be invalid and the abort would be cascaded to T1 (i.e. T1 would also have to abort.).
2. Strict 2PL would require T2 to obtain x(Y) before writing to it. This lock would have to be held until T2 committed or aborted; this would block T1 from reading Y until T2 was finished, but there would be no interference.
3. Strict 2PL is popular for many reasons. One reason is that it ensures only ‘safe’ interleaving of transactions so that transactions are recoverable, avoid cascading aborts, etc. Another reason is that strict 2PL is very simple and easy to implement. The lock manager only needs to provide a lookup for exclusive locks and an atomic locking mechanism (such as with a semaphore).

## T-5

### Description



### Solution

1. T1: R(x) R(Y) W(X) Abort

T2: R(X) R(Y) W(X) W(Y)

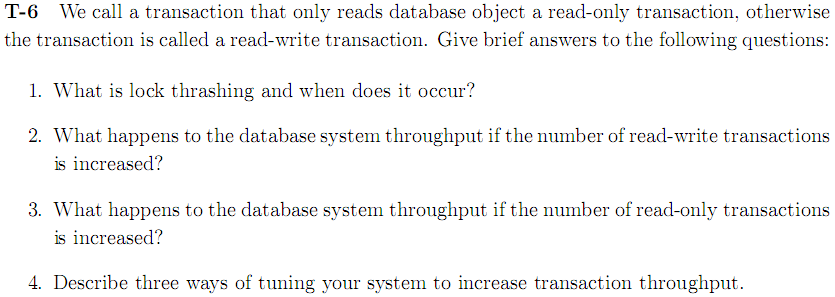
1. 有问题
2. T1: R(x) R(Y) W(X) C

T2: R(X) R(Y) W(X) W(Y) C

1. 用优先图判断环

## T-6

### Description

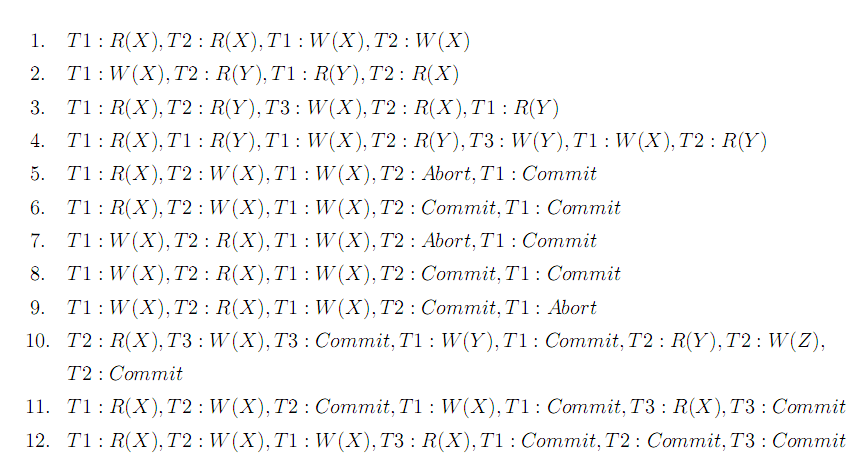
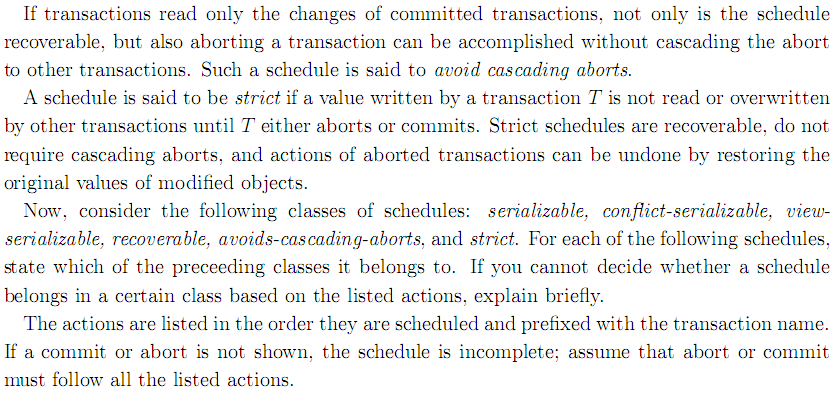
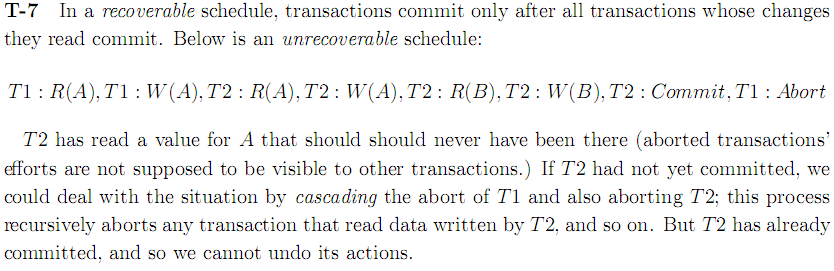


### Solution

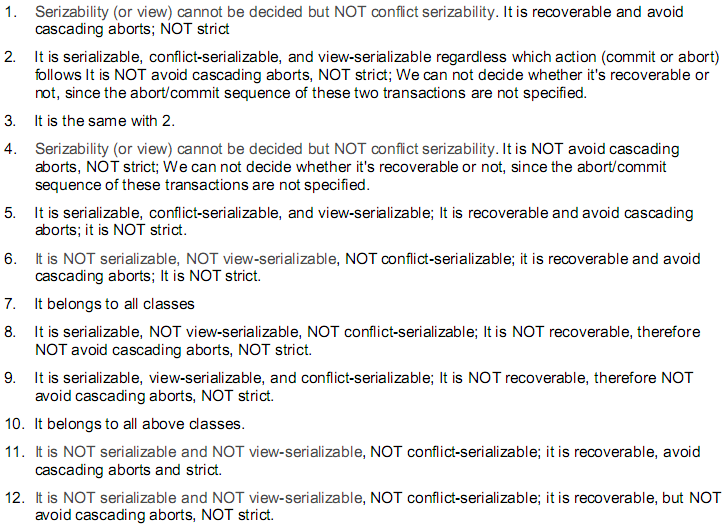
1. Lock Thrashing is the point where system performance(throughput) decreases with increasing load (adding more active transactions). It happens due to the contention of locks. Transactions waste time on lock waits.
2. At the beginning, with the increase of transactions, throughput will increase, but the increase will stop at the thrashing point, after the point the throughput will drop with the increasing number of transactions.
3. For read-only transactions, there is no lock waiting. The throughput will increase with the increasing number of concurrent transactions.
4. Three ways of tuning your system to increase transaction throughput:
5. By locking the smallest sized objects possible (reducing the likelihood that two transactions need the same lock).
6. By reducing the time that transaction hold locks (so that other transactions are blocked for a shorter time).
7. By reducing hot spots. A hot spot is a database object that is frequently accessed and modified, and causes a lot of blocking delays. Hot spots can significantly affect performance.

## T-7

### Description

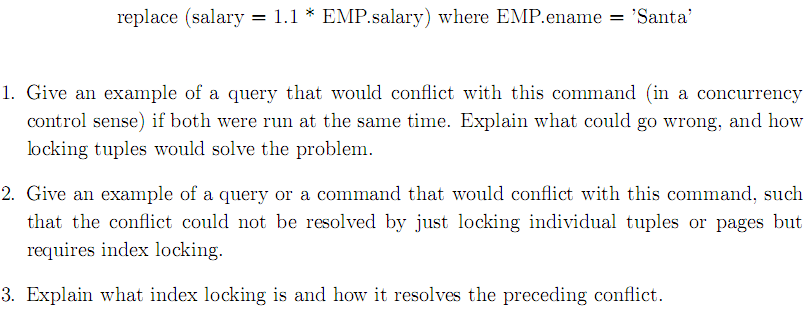
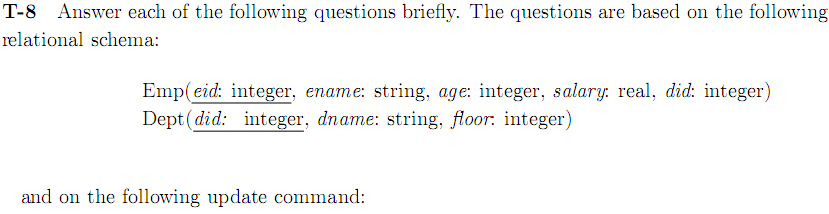


### Solution



## T-8

### Description



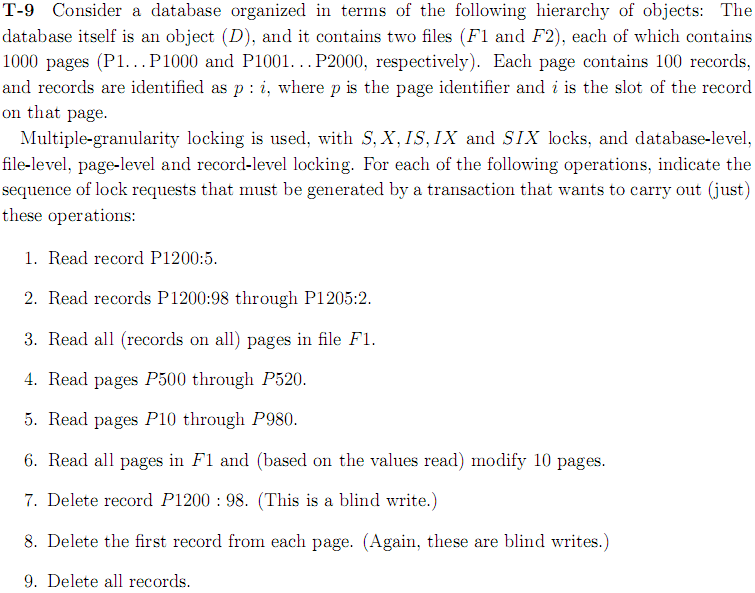
### Solution

1. select salary from EMP where EMP.ename=’Santa’
2. insert into EMP values(2, ‘Santa’, 18, 10000, 2)
3. Index locking is a technique used to maintain index integrity. A portion of an index is locked during a database transaction when this portion is being accesses by the transaction as a result of attempt to access related user data.

When a portion of index is locked by a transaction, other transactions may be blocked from accessing this index portion (blocked from modifying, and even from reading it, depending on lock type and needed operation)

## T-9

### Description



### Solution

1. D: IS F2: IS P1200: IS P1200:5: S
2. P1200: IS P1200:98: S
3. D: IS F1: S
4. P500: S
5. P10: S
6. D: SIX F1: SIX 10pages: X
7. D: IX F2: IX P1200: IX P1200:98: X
8. D: IX F1: IX F2: IX all pages: IX P1:1, P2:1, …, P2000:1: X
9. D: IX F1: IX F2: IX P1, P2, …, P2000: X

## X-1

### Description



### Solution

Hypertext Markup Language (HTML) is the main markup language for displaying web pages and other information that can be displayed in a web browser.

HTML does not provide meaning to the data contained in HTML tags. HTML cannot accommodate highly refined searches across data.

XML is extensible; it has limitless ability to define new languages or data sets. You can include your data and a description of what the data represents. This is useful for defining your own language or protocol.

## X-2

### Description



### Solution

Hypertext Markup Language (**HTML**) is the main markup language for displaying web pages and other information that can be displayed in a web browser.

In computing, a uniform resource locator (**URL**) (originally called universal resource locator) is a specific character string that constitutes a reference to an Internet resource.

Extensible Markup Language (**XML**) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable.

**XPath**, the XML Path Language, is a query language for selecting nodes from an XML document. In addition, XPath may be used to compute values (e.g., strings, numbers, or Boolean values) from the content of an XML document.

A Document Type Definition (**DTD**) is a set of markup declarations that define a document type for an SGML-family markup language (SGML, XML, HTML).

A **cookie** is usually a small piece of data sent from a website and stored in a user's web browser while a user is browsing a website. When the user browses the same website in the future, the data stored in the cookie can be retrieved by the website to notify the website of the user's previous activity. Cookies were designed to be a reliable mechanism for websites to remember the state of the website or activity the user had taken in the past. This can include clicking particular buttons, logging in, or a record of which pages were visited by the user even months or years ago.

The Hypertext Transfer Protocol (**HTTP**) is an application protocol for distributed, collaborative, hypermedia information systems. HTTP is the foundation of data communication for the World Wide Web.

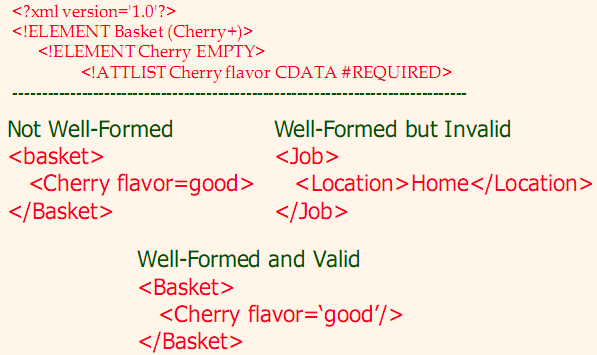
Hypertext Transfer Protocol Secure (**HTTPS**) is a widely used communications protocol for secure communication over a computer network, with especially wide deployment on the Internet. Technically, it is not a protocol in itself; rather, it is the result of simply layering the Hypertext Transfer Protocol (HTTP) on top of the SSL/TLS protocol, thus adding the security capabilities of SSL/TLS to standard HTTP communications.

## X-3

### Description

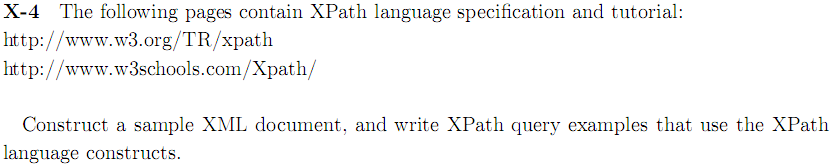


### Solution



## X-4

### Description

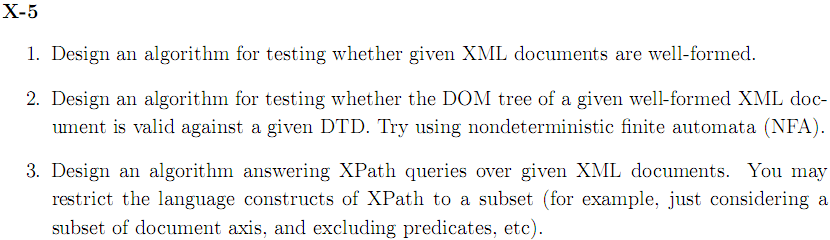


### Solution

So…easy…

## X-5

### Description

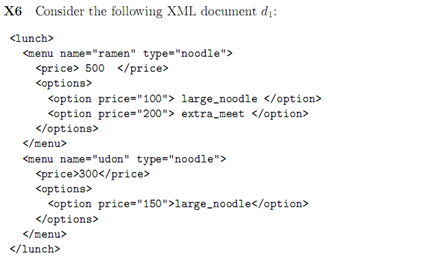


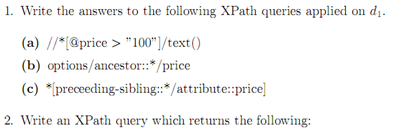
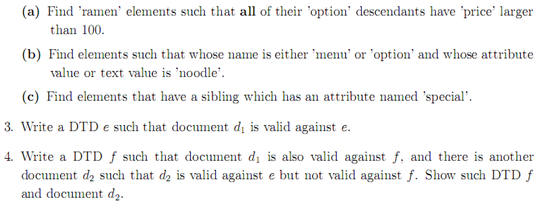
### Solution

So…hard…

## X-6

### Description



### Solution

1. (a). large\_noodle extra\_meet

(b). <price>500</price> <price>300</price>

(c). <option price=”200”>extra\_meet</option>

1. \*[@name=”ramen”] [option/@price>”100”]

\* [name() = menu|option] [@\*=”noodle” | text()=”noodle”]

\* [preceding-sibling::\* | following-sibling::\* /attribute::special]

1. <?xml version=’1.0’ ?>

<!ELEMENT lunch(menu\*)>

<!ELEMENT menu(price\*, options\*)>

<!ATTLIST menu name CDATA #REQUIRED>

<!ATTLIST menu type CDATA #REQUIRED>

<!ELEMENT price EMPTY>

<!ELEMENT options(option\*)>

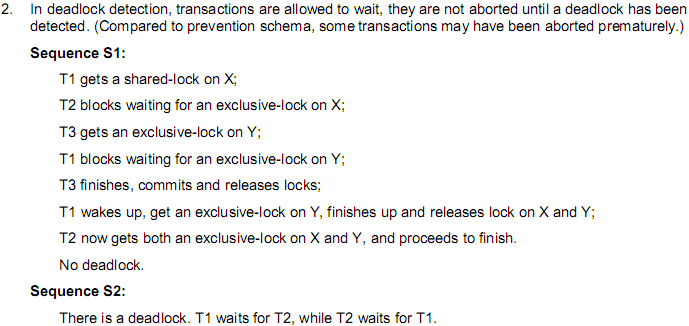
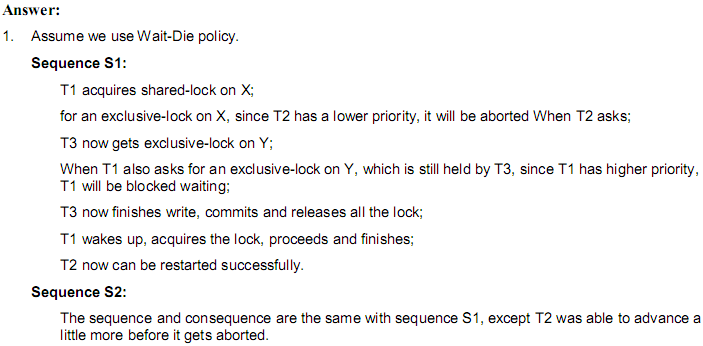
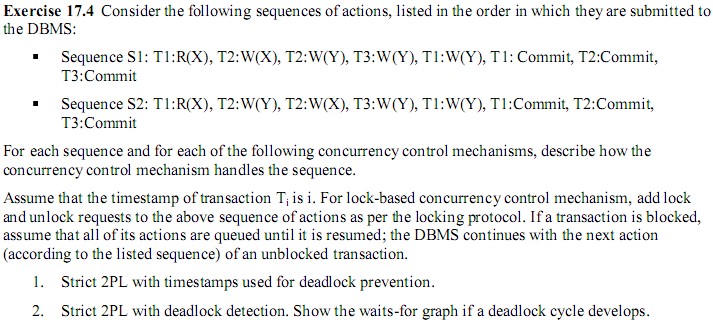
<!ELEMENT option EMPTY>

<!ATTLIST option price CDATA #REQUIRED>

1. f : 将e中的\*改成+

d2: <lunch></lunch>

## 附加题1



## 附加题2

