Introduction to Transaction Processing (1)

- **Single-User System:**
  - At most one user at a time can use the system.
- **Multiuser System:**
  - Many users can access the system concurrently.
- **Concurrency**
  - **Interleaved processing:**
    - Concurrent execution of processes is interleaved in a single CPU
  - **Parallel processing:**
    - Processes are concurrently executed in multiple CPUs.

- **نظام المستخدم المفرد:**
  - على الأكثر مستخدم واحد في وقت واحد يمكن استخدام النظام.
- **نظام متعدد المستخدمين:**
  - العديد من المستخدمين يمكن الوصول إلى النظام في وقت واحد.
- **التزامن**
  - **معالجة معشق:**
    - تعارض تنفيذ العمليات المتزامنة في وحدة معالجة مركزية واحدة
  - **المعالجة المتوازية:**
    - يتم تنفيذ العمليات في وقت واحد في وحدات المعالجة المركزية متعددة.

Introduction to Transaction Processing (2)

- A **Transaction**:
  - Logical unit of database processing that includes one or more access operations (read -retrieval, write - insert or update, delete).
- A transaction (set of operations) may be stand-alone specified in a high level language like SQL submitted interactively, or may be embedded within a program.
- **Transaction boundaries**:
  - **Begin and End transaction.**
- An **application program** may contain several transactions separated by the Begin and End transaction boundaries.

- **A عملية :**
  - وحدة منطقية من معالجة قاعدة البيانات التي تتضمن واحد أو أكثر من عمليات الوصول (قراءةretrieval- ، والكتابة - إدراج أو تحديث، حذف).
- الصفقة (مجموعة من العمليات) قد تكون قائمة بذاتها محددة بلغة عالية المستوى مثل SQL المقدمة بشكل تفاعلي، أو قد تكون جزءا لا يتجزأ من البرنامج .
- **حدود عملية :**
  - **بدء ونهاية الصفقة .**
- قد تحتوي على **برنامج تطبيق** العديد من الصفقات التي فرقتها الحدود بيغن والمعاملة النهاية .

Introduction to Transaction Processing (3)

SIMPLE MODEL OF A DATABASE (for purposes of discussing transactions):

- **A database** is a collection of named data items
- **Granularity** of data - a field, a record, or a whole disk block (Concepts are independent of granularity)
- Basic operations are **read** and **write**
  - **read item(X)**: Reads a database item named X into a program variable. To simplify our notation, we assume that the program variable is also named X.
  - **write item(X)**: Writes the value of program variable X into the database item named X.

نموذج بسيط من قاعدة البيانات (لأغراض مناقشة المعاملات) :

- **قاعدة بيانات** عبارة عن مجموعة من عناصر البيانات اسمه
- **تجزئة** البيانات - حقل، وهو رقم قياسي، أو كتلة القرص كله (مفاهيم مستقلة من التفاصيل)
- تتم **قراءة** العمليات الأساسية **والكتابة**

  - **read_item (X)** يقرأ عنصر قاعدة البيانات المسمى X إلى متغير برنامج .لتبسيط تدويننا، نفترض أن متغير البرنامج يسمى أيضا X.
  - **write_item (X)** يكتب قيمة المتغير برنامج X في البند قاعدة البيانات المسمى X.

Introduction to Transaction Processing (4)

READ AND WRITE OPERATIONS:

- Basic unit of data transfer from the disk to the computer main memory is one block. In general, a data item (what is read or written) will be the field of some record in the database, although it may be a larger unit such as a record or even a whole block.
- **read_item(X)** command includes the following steps:
  - Find the address of the disk block that contains item X.
  - Copy that disk block into a buffer in main memory (if that disk block is not already in some main memory buffer).
  - Copy item X from the buffer to the program variable named X.

- قراءة وكتابة العمليات :
- الوحدة الأساسية لنقل البيانات من القرص إلى الذاكرة الرئيسية للكمبيوتر كتلة واحدة .وبصفة عامة، فإن بند البيانات (ما يقرأ أو يكتب) سيكون مجال بعض السجلات في قاعدة البيانات، على الرغم من أنه قد يكون وحدة أكبر مثل سجل أو حتى كتلة كاملة .
- يتضمن الأمر **read_item (X)** الخطوات التالية :
  - ابحث عن عنوان كتلة القرص الذي يحتوي على العنصر X.
  - نسخ كتلة القرص إلى مخزن مؤقت في الذاكرة الرئيسية (إذا كان هذا القرص غير موجود بالفعل في بعض المخزن المؤقت الذاكرة الرئيسية) .
  - نسخ العنصر X من المخزن المؤقت إلى متغير البرنامج المسمى X.

Introduction to Transaction Processing (5)

READ AND WRITE OPERATIONS (cont.):

- **write_item(X)** command includes the following steps:
  - Find the address of the disk block that contains item X.
  - Copy that disk block into a buffer in main memory (if that disk block is not already in some main memory buffer).
  - Copy item X from the program variable named X into its correct location in the buffer.
  - Store the updated block from the buffer back to disk (either immediately or at some later point in time).

قراءة وكتابة العمليات (تابع) :

- ويشمل **write_item (X)** الأمر الخطوات التالية :
  - ابحث عن عنوان كتلة القرص الذي يحتوي على العنصر X.
  - نسخ كتلة القرص إلى مخزن مؤقت في الذاكرة الرئيسية (إذا كان هذا القرص غير موجود بالفعل في بعض المخزن المؤقت الذاكرة الرئيسية) .
  - نسخ العنصر X من متغير البرنامج المسمى X في موقعه الصحيح في المخزن المؤقت .
  - تخزين كتلة محدثة من المخزن المؤقت مرة أخرى إلى القرص (إما على الفور أو في بعض نقطة لاحقة في الوقت المناسب) .

Two Sample Transactions

(a)

T_1

read_item(X);
$X := X - N$;
write_item(X);
read_item(Y);
$Y := Y + N$;
write_item(Y);

(b)

T_2

read_item(X);
$X := X + M$;
write_item(X);

Figure 21.2
Two sample transactions. (a) Transaction T_1. (b) Transaction T_2.

Introduction to Transaction Processing (6)

Why Concurrency Control is needed:

- **The Lost Update Problem**
  - This occurs when two transactions that access the same database items have their operations interleaved in a way that makes the value of some database item incorrect.
- **The Temporary Update (or Dirty Read) Problem**
  - This occurs when one transaction updates a database item and then the transaction fails for some reason (see Section 21.1.4).
  - The updated item is accessed by another transaction before it is changed back to its original value.
- **The Incorrect Summary Problem**
  - If one transaction is calculating an aggregate summary function on a number of records while other transactions are updating some of these records, the aggregate function may calculate some values before they are updated and others after they are updated.

لماذا يلزم التحكم في التزامن :

- **مشكلة التحديث المفقود**
  - يحدث هذا عندما المعاملات اثنين التي الوصول إلى نفس عناصر قاعدة البيانات عملياتها معشق بطريقة تجعل قيمة بعض عنصر قاعدة البيانات غير صحيحة .
- **مشكلة التحديث المؤقت (أو القراءة القذرة)**
  - يحدث هذا عندما تقوم معاملة واحدة بتحديث عنصر قاعدة بيانات ثم تفشل المعاملة لسبب ما (انظر القسم 21.1.4) .
  - يتم الوصول إلى العنصر المحدث بواسطة معاملة أخرى قبل أن يتم تغييره مرة أخرى إلى قيمته الأصلية .
- **مشكلة الملخص غير صحيحة**
  - إذا كانت إحدى المعاملات تقوم بحساب دالة ملخص تجميعي على عدد من السجلات بينما تقوم معاملات أخرى بتحديث بعض هذه السجلات، فإن الدالة التجميعية قد تحسب بعض القيم قبل تحديثها وبعضها الآخر بعد تحديثها .

Concurrent execution is uncontrolled: (a) The lost update problem

التنفيذ المتزامن غير المنضبط: (أ) مشكلة التحديث المفقودة

(أ) مشكلة التحديث المفقودة.

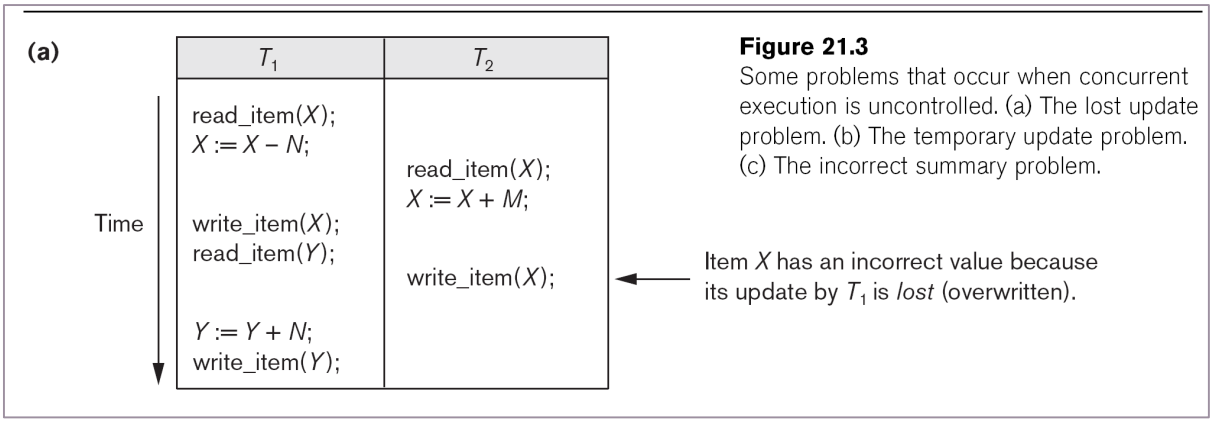
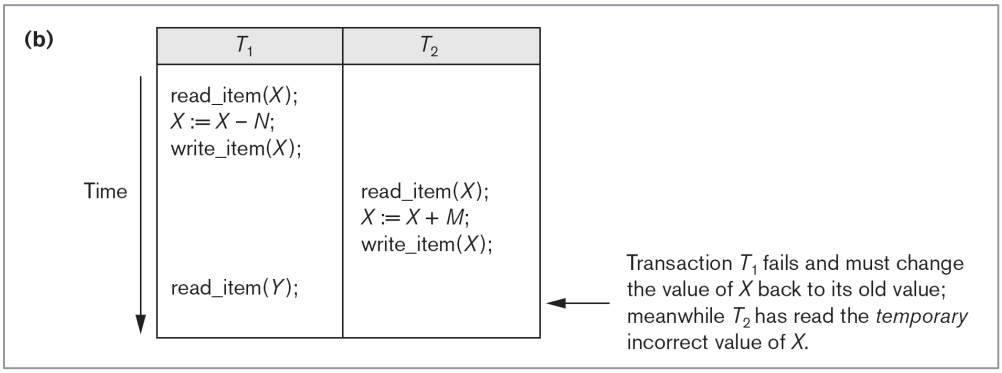


Figure 21.3
Some problems that occur when concurrent execution is uncontrolled. (a) The lost update problem. (b) The temporary update problem. (c) The incorrect summary problem.

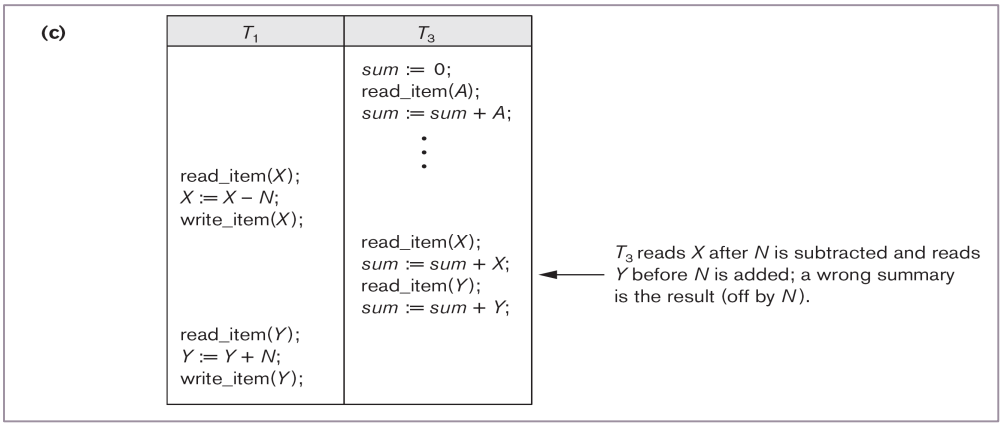
Concurrent execution is uncontrolled:
(b) The temporary update problem.

التنفيذ المتزامن غير منضبط:
(ب) مشكلة التحديث المؤقت.



Concurrent execution is uncontrolled:
The incorrect summary problem

التنفيذ المتزامن هو غير المنضبط:
ج) مشكلة ملخص غير صحيحة.



Introduction to Transaction Processing (12)

Why **recovery** is needed:
(What causes a Transaction to fail)

1. A computer failure (system crash):

A hardware or software error occurs in the computer system during transaction execution. If the hardware crashes, the contents of the computer's internal memory may be lost.

2. A transaction or system error:

Some operation in the transaction may cause it to fail, such as integer overflow or division by zero. Transaction failure may also occur because of erroneous parameter values or because of a logical programming error. In addition, the user may interrupt the transaction during its execution.

لماذا هناك حاجة الانتعاش او الاسترداد :
ما الذي يؤدي إلى فشل المعاملة

1. **فشل الكمبيوتر (تحطم النظام):**
يحدث خطأ في الأجهزة أو البرامج في نظام الكمبيوتر أثناء تنفيذ المعاملة .إذا تعطل الجهاز، قد تفقد محتويات الذاكرة الداخلية للكمبيوتر .
2. **خطأ في المعاملة أو النظام :**
قد تؤدي بعض العمليات في المعاملة إلى الفشل، مثل تجاوز عدد صحيح أو تقسيم بمقدار صفر .قد يحدث فشل المعاملة أيضا بسبب قيم معلمة خطأ أو بسبب خطأ برمجة منطقي .بالإضافة إلى ذلك، قد يقوم المستخدم بقطع المعاملة أثناء تنفيذها.

Introduction to Transaction Processing (13)

Why **recovery** is needed (cont.):

(What causes a Transaction to fail)

3. Local errors or exception conditions detected by the transaction:

Certain conditions necessitate cancellation of the transaction. For example, data for the transaction may not be found. A condition, such as insufficient account balance in a banking database, may cause a transaction, such as a fund withdrawal from that account, to be canceled.

A programmed abort in the transaction causes it to fail.

4. Concurrency control enforcement:

The concurrency control method may decide to abort the transaction, to be restarted later, because it violates serializability or because several transactions are in a state of deadlock (see Chapter 22).

لماذا هناك حاجة **الاسترداد** (تابع):
ما الذي يؤدي إلى فشل المعاملة

3. الأخطاء المحلية أو شروط الاستثناء التي تم الكشف عنها بواسطة المعاملة :

تتطلب بعض الشروط إلغاء المعاملة .على سبيل المثال، قد لا يتم العثور على بيانات المعاملة .وهناك شرط، مثل عدم كفاية رصيد الحساب في قاعدة بيانات مصرفية، قد يؤدي إلى إلغاء الصفقة، مثل سحب الأموال من ذلك الحساب يؤدي الإجهاض المبرمج في المعاملة إلى الفشل .

4. تطبيق مراقبة التزامن :

قد تقرر طريقة التحكم التزامن لإجهاض المعاملة، لإعادة تشغيل لاحقا، لأنه ينتهك تسلسل أو لأن العديد من المعاملات في حالة من الجمود (انظر الفصل 22)

Introduction to Transaction Processing (14)

Why **recovery** is needed (cont.):

(What causes a Transaction to fail)

5. Disk failure:

Some disk blocks may lose their data because of a read or write malfunction or because of a disk read/write head crash. This may happen during a read or a write operation of the transaction.

6. Physical problems and catastrophes:

This refers to an endless list of problems that includes power or air-conditioning failure, fire, theft, sabotage, overwriting disks or tapes by mistake, and mounting of a wrong tape by the operator.

لماذا هناك حاجة (**الاسترداد**)تابع:
(ما الذي يؤدي إلى فشل المعاملة)

5. فشل القرص :

قد تفقد بعض كتل القرص البيانات الخاصة بهم بسبب قراءة أو كتابة عطل أو بسبب القرص القرص / الكتابة رئيس تحطم الطائرة .قد يحدث هذا أثناء قراءة أو عملية كتابة المعاملة .

6. المشاكل والكوارث المادية :

هذا يشير إلى قائمة لا نهاية لها من المشاكل التي تشمل السلطة أو فشل تكييف الهواء، والحريق، والسرقة، والتخريب، الكتابة فوق الأقراص أو الأشرطة عن طريق الخطأ، وتصاعد الشريط الخطأ من قبل المشغل

2 Transaction and System Concepts (1)

- A **transaction** is an atomic unit of work that is either completed in its entirety or not done at all.
  - For recovery purposes, the system needs to keep track of when the transaction starts, terminates, and commits or aborts.
- **Transaction states:**
  - Active state
  - Partially committed state
  - Committed state
  - Failed state
  - Terminated State

- **معاملة** هي وحدة ذرية من العمل الذي هو إما أنجزت في مجملها أو لم تفعل على الإطلاق .
  - ولأغراض الاسترداد، يحتاج النظام إلى تتبع متى تبدأ المعاملة أو ينتهي أو يرتكب أو يضر .
- **الدول الصفقة :**
  - حالة نشطة
  - حاله ملتزمة جزئيا
  - حاله ارتكبت
  - حاله منتهيه
  - الدولة المنتهية

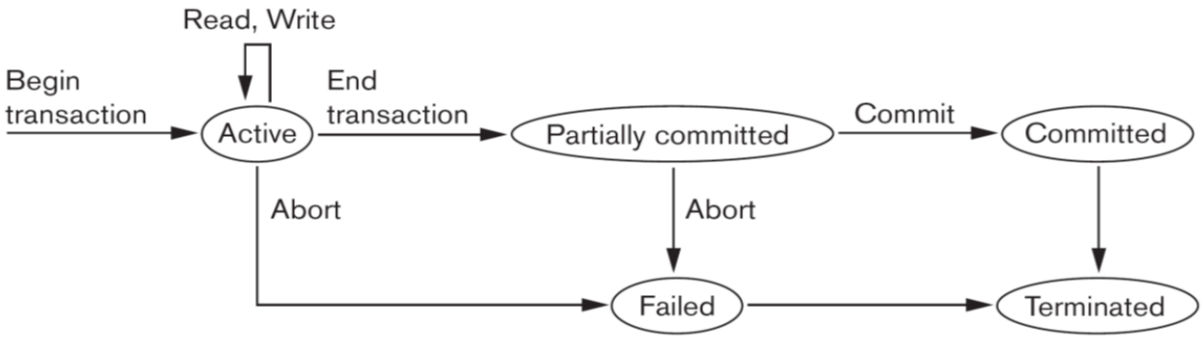
Transaction and System Concepts (4)

- Recovery techniques use the following operators:
  - **undo**: Similar to rollback except that it applies to a single operation rather than to a whole transaction.
  - **redo**: This specifies that certain *transaction operations* must be *redone* to ensure that all the operations of a committed transaction have been applied successfully to the database.

- تستخدم تقنيات الاسترداد المشغلين التاليين :
  - **التراجع** :على غرار التراجع إلا أنه ينطبق على عملية واحدة بدلا من صفقة كاملة .
  - **إعادة** :يحدد هذا أن *عمليات معاملة* معينة ويجب *إعادة بنائه* لضمان أن جميع عمليات صفقة ملتزمة طبقت بنجاح إلى قاعدة البيانات .

State Transition Diagram Illustrating the States for Transaction Execution

Figure 21.4
State transition diagram illustrating the states for transaction execution.



Transaction and System Concepts (10)

Commit Point of a Transaction:

- **Definition a Commit Point:**
  - A transaction T reaches its **commit point** when all its operations that access the database have been executed successfully *and* the effect of all the transaction operations on the database has been recorded in the log.
  - Beyond the commit point, the transaction is said to be committed, and its effect is assumed to be permanently recorded in the database.
  - The transaction then writes an entry [commit,T] into the log.
- **Roll Back of transactions:**
  - Needed for transactions that have a [start_transaction,T] entry into the log but no commit entry [commit,T] into the log.

- الالتزام نقطة من الصفقة :
- **تعريف نقطة الالتزام :**
  - وقد سجلت معاملة T يصل لها **نقطة الالتزام** عند كل عملياتها الوصول إلى قاعدة بيانات تم تنفيذها بنجاح *وتأثير* كل تلك العمليات معاملة في قاعدة البيانات في السجل .
  - وبعيدا عن نقطة الالتزام، يقال إن المعاملة ملتزمة، ومن المفترض أن يتم تسجيل أثرها بشكل دائم في قاعدة البيانات .
  - ثم يكتب العملية إدخال] كوميت، [Tفي السجل .
- **عودة المعاملات :**
  - مطلوب للمعاملات التي تحتوي على إدخال[start_transaction ، [Tفي السجل ولكن لا يلزم إدخال] كوميت، [Tفي السجل

Transaction and System Concepts (11)

Commit Point of a Transaction (cont):

- **Redoing transactions:**
  - Transactions that have written their commit entry in the log must also have recorded all their write operations in the log; otherwise they would not be committed, so their effect on the database can be redone from the log entries. (Notice that the log file must be kept on disk.
  - At the time of a system crash, only the log entries that have been written back to disk are considered in the recovery process because the contents of main memory may be lost.)
- **Force writing a log:**
  - Before a transaction reaches its commit point, any portion of the log that has not been written to the disk yet must now be written to the disk.
  - This process is called force-writing the log file before committing a transaction.

- الالتزام بنقطة المعاملة (تابع) :
- **إعادة المعاملات :**
  - يجب أن تكون المعاملات التي كتبت إدخال الالتزام في السجل أيضا قد سجلت جميع عمليات الكتابة في السجل؛ وإلا فإنها لن تكون ملتزمة، لذلك تأثيرها على قاعدة البيانات يمكن إعادة تعيين من إدخالات السجل) .لاحظ أنه يجب الاحتفاظ بملف السجل على القرص .
  - في وقت تعطل النظام فقط إدخالات السجل التي تم كتابتها مرة أخرى إلى القرص تعتبر في عملية الاسترداد لأنه قد يتم فقدان محتويات الذاكرة الرئيسية (.
- **فرض كتابة سجل :**
  - قبل أن تصل المعاملة إلى نقطة الالتزام الخاصة بها، يجب كتابة أي جزء من السجل الذي لم يتم كتابته إلى القرص حتى الآن إلى القرص .
  - وتسمى هذه العملية قوة كتابة ملف السجل قبل ارتكاب معاملة .

3 Desirable Properties of Transactions (1)

ACID properties:

- **Atomicity**: A transaction is an atomic unit of processing; it is either performed in its entirety or not performed at all.
- **Consistency preservation**: A correct execution of the transaction must take the database from one consistent state to another.
- **Isolation**: A transaction should not make its updates visible to other transactions until it is committed; this property, when enforced strictly, solves the temporary update problem and makes cascading rollbacks of transactions unnecessary (see Chapter 21).
- **Durability or permanency**: Once a transaction changes the database and the changes are committed, these changes must never be lost because of subsequent failure.

خصائص ACID:

- **Atomicity:** معاملة هي وحدة ذرية من تجهيز؛ فإنه إما أن يؤدي في مجملها أو لا يؤدي على الإطلاق .
- **الحفاظ على الاتساق A :**التنفيذ الصحيح لهذه الصفقة يجب أن قاعدة البيانات من حالة متناسقة واحد إلى آخر .
- **العزلة :**يجب معاملة لا تجعل التحديثات الخاصة به مرئية للمعاملات أخرى حتى تلتزم به؛ فإن هذه الخاصية، عند فرضها بدقة، تحل مشكلة التحديث المؤقت وتؤدي إلى تعويضات متعاقبة من المعاملات غير ضرورية (انظر الفصل 21) .
- **المتانة أو ديمومة :**عندما تتغير معاملة قاعدة البيانات وملتزمون التغييرات، يجب ألا تضيع هذه التغييرات بسبب الفشل اللاحق.

4 Characterizing Schedules Based on Recoverability (1)

- **Transaction schedule or history**:
  - When transactions are executing concurrently in an interleaved fashion, the order of execution of operations from the various transactions forms what is known as a transaction schedule (or history).
- A **schedule** (or **history**) S of n transactions T1, T2, …, Tn:
  - It is an ordering of the operations of the transactions subject to the constraint that, for each transaction Ti that participates in S, the operations of T1 in S must appear in the same order in which they occur in T1.
  - Note, however, that operations from other transactions Tj can be interleaved with the operations of Ti in S.

- **جدول عملية أو تاريخ :**
  - عندما يتم تنفيذ المعاملات بشكل متزامن بطريقة مشوشة، يشكل ترتيب تنفيذ العمليات من المعاملات المختلفة ما يعرف بجدول المعاملات (أو التاريخ) .
- **جدول)أو التاريخ** S of n transactions T1, T2, …, Tn:
  - وهو ترتيب لعمليات المعاملات تخضع للقيود التي، لكل تي المعاملة التي تشارك فيS ، يجب أن تظهر عمليات T1 في S في نفس الترتيب الذي تحدث في T1.
  - لاحظ، مع ذلك، أن العمليات من المعاملات الأخرى تي يمكن أن تكون معشق مع عمليات تي في S.

5 Characterizing Schedules Based on Serializability (1)

- Serial schedule:
  - A schedule S is serial if, for every transaction T participating in the schedule, all the operations of T are executed consecutively in the schedule.
    - Otherwise, the schedule is called nonserial schedule.
- Serializable schedule:
  - A schedule S is serializable if it is equivalent to some serial schedule of the same n transactions.

- الجدول الزمني التسلسلي:
  - جدول S هو المسلسل إذا، لكل المعاملات T المشاركة في الجدول الزمني، يتم تنفيذ جميع عمليات T على التوالي في الجدول الزمني.
    - خلاف ذلك، ويسمى الجدول الزمني الجدول الزمني غير مجهري.
- جدول التسلسل:
  - جدول S هو تسلسل إذا كان يعادل بعض الجدول الزمني التسلسلي من نفس المعاملات ن.

Characterizing Schedules Based on Serializability (2)

- Result equivalent:
  - Two schedules are called result equivalent if they produce the same final state of the database.
- Conflict equivalent:
  - Two schedules are said to be conflict equivalent if the order of any two conflicting operations is the same in both schedules.
- Conflict serializable:
  - A schedule S is said to be conflict serializable if it is conflict equivalent to some serial schedule S'.

- النتيجة المكافئة:
  - ويطلق على جدولين ما يعادل النتيجة إذا كانت تنتج نفس الحالة النهائية لقاعدة البيانات.
- تعادل النزاع:
  - ويقال إن جدولين متكافئين إذا كان ترتيب أي عمليتين متعارضتين هو نفسه في كلا الجدولين.
- تسلسل النزاع:
  - ويقال جدول S أن يكون الصراع تسلسل إذا كان الصراع يعادل بعض الجدول الزمني المسلسل S '.

Characterizing Schedules Based on Serializability (3)

- Being serializable is not the same as being serial
- Being serializable implies that the schedule is a correct schedule.
  - It will leave the database in a consistent state.
  - The interleaving is appropriate and will result in a state as if the transactions were serially executed, yet will achieve efficiency due to concurrent execution.

- يجري تسلسل ليست هي نفسها بأنها المسلسل
- يجري تسلسل يعني أن الجدول الزمني هو الجدول الزمني الصحيح.
  - فإنه سيتم ترك قاعدة البيانات في حالة متسقة.
  - interleaving هو المناسب وسوف يؤدي إلى حالة كما لو كانت المعاملات نفذت بشكل متسلسل، ولكن تحقيق الكفاءة بسبب التنفيذ المتزامن.

Characterizing Schedules Based on Serializability (4)

- Serializability is hard to check.
  - Interleaving of operations occurs in an operating system through some scheduler
  - Difficult to determine beforehand how the operations in a schedule will be interleaved.

- من الصعب التحقق من التسلسل.
  - يحدث تشذير العمليات في نظام تشغيل من خلال بعض المجدول
  - من الصعب تحديد كيفية تشابك العمليات في جدول زمني مسبق.

Characterizing Schedules Based on Serializability (11)

Testing for conflict serializability: Algorithm 21.1:

- Looks at only read_Item (X) and write_Item (X) operations
- Constructs a precedence graph (serialization graph) - a graph with directed edges
- An edge is created from Ti to Tj if one of the operations in Ti appears before a conflicting operation in Tj
- The schedule is serializable if and only if the precedence graph has no cycles.

- **اختبار تسلسل الصراع: خوارزمية 21.1:**
  - يبدو فقط في القراءة_Item (X) و write_Item (X) العمليات
  - يبني الرسم البياني الأسبقية (الرسم البياني التسلسل) - رسم بياني مع حواف موجهة
  - يتم إنشاء حافة من تي إلى تي إذا ظهرت إحدى العمليات في تي قبل عملية متضاربة في تي
  - الجدول الزمني هو تسلسل إذا وفقط إذا كان الرسم البياني الأسبقية ليس لديه دورات.

Constructing the Precedence Graphs إنشاء الرسوم البيانية الأسبقية

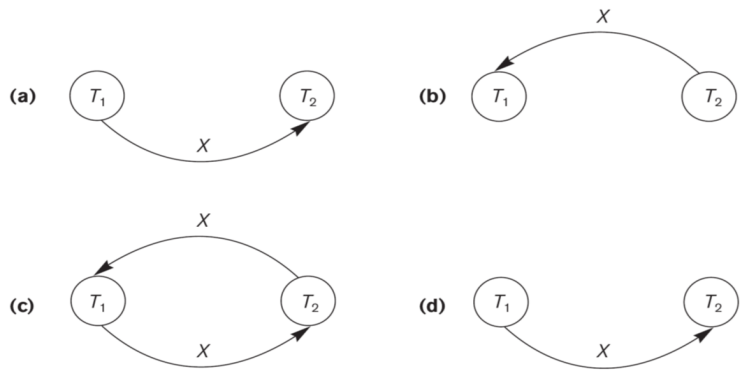


Figure 21.7
Constructing the precedence graphs for schedules A to D from Figure 21.5 to test for conflict serializability. (a) Precedence graph for serial schedule A. (b) Precedence graph for serial schedule B. (c) Precedence graph for schedule C (not serializable). (d) Precedence graph for schedule D (serializable, equivalent to schedule A).

Another Example of Serializability Testing

مثال آخر لاختبار التسلسل

(a)

Transaction T_1	Transaction T_2	Transaction T_3
read_item(X);	read_item(Z);	read_item(Y);
write_item(X);	read_item(Y);	read_item(Z);
read_item(Y);	write_item(Y);	write_item(Y);
write_item(Y);	read_item(X);	write_item(Z);
	write_item(X);	

(b)

Time	Transaction T_1	Transaction T_2	Transaction T_3
		read_item(Z);	
		read_item(Y);	
		write_item(Y);	
			read_item(Y);
			read_item(Z);
	read_item(X);		
	write_item(X);		
			write_item(Y);
			write_item(Z);
		read_item(X);	
	read_item(Y);		
	write_item(Y);		
		write_item(X);	

Schedule E

(c)

Time	Transaction T_1	Transaction T_2	Transaction T_3
			read_item(Y);
			read_item(Z);
	read_item(X);		
	write_item(X);		
			write_item(Y);
			write_item(Z);
		read_item(Z);	
	read_item(Y);		
	write_item(Y);		
		read_item(Y);	
		write_item(Y);	
		read_item(X);	
		write_item(X);	

Schedule F

Figure 21.8
Another example of serializability testing. (a) The read and write operations of three transactions T_1, T_2, and T_3. (b) Schedule E. (c) Schedule F.

Summary

- Transaction and System Concepts
- Desirable Properties of Transactions
- Characterizing Schedules based on Recoverability
- Characterizing Schedules based on Serializability
- Transaction Support in SQL

ملخص

- المعاملات ومفاهيم النظام
- خصائص مرغوبة من المعاملات
- توصيف الجداول على أساس الاسترداد
- وصف جداول استنادا إلى تسلسل
- دعم المعاملات في SQL