Chapter 1

Basic Definitions:

Database:

A collection of related data.

((مجموعة من البيانات ذات الصلة))

Data:

Known facts that can be recorded and have an implicit meaning.

((الحقائق المعروفة التي يمكن تسجيلها ولها معنى ضمني))

Mini world:

Some part of the real world about which data is stored in a database. For example, student grades and transcripts at a university.

((جزء من العالم الحقيقي يتعلق بالبيانات المخزنة في قاعدة بيانات. على سبيل المثال، درجات الطالب ونصوصه في إحدى الجامعات))

Database Management System (DBMS):

A software package/ system to facilitate the creation and maintenance of a computerized database.

((حزمة / نظام برمجيات لتسهيل إنشاء وصيانة قاعدة بيانات محوسبة))

Database System:

The DBMS software together with the data itself. Sometimes, the applications are also included.

((برنامج DBMS مع البيانات نفسها. في بعض الأحيان، يتم تضمين التطبيقات أيضًا))

The four components of a *database system* are:

- Users.
- Database Application.
- Database Management System (DBMS).

- Database.

Users:

A user of a database system will:

- Use a database application to track things.
- Use forms to enter, read, delete, and query data.
- Produce reports.

The Database:

A database is a self-describing collection of related records.

((قاعدة البيانات هي مجموعة ذاتية الوصف من السجلات ذات الصلة))

Self-describing:

The database itself contains the definition of its structure.

((تحتوي قاعدة البيانات نفسها على تعريف هيكلها))

Metadata is data describing the structure of the database data.

((البيانات الوصفية هي بيانات تصف هيكل بيانات قاعدة البيانات))

Tables within a relational database are related to each other.

((ترتبط الجداول الموجودة في قاعدة البيانات العلائقية ببعضها البعض))

Database Management System (DBMS):

A database management system (DBMS) serves as an intermediary between database applications and the database.

((يعمل نظام إدارة قواعد البيانات (DBMS) كوسيط بين تطبيقات قاعدة البيانات وقاعدة البيانات))

The DBMS manages and controls database activities.

((يقوم نظام إدارة قواعد البيانات (DBMS) بإدارة أنشطة قاعدة البيانات والتحكم فيها))

The DBMS creates, processes, and administers the databases it controls.

((يقوم DBMS بإنشاء ومعالجة وإدارة قواعد البيانات التي يتحكم فيها))

Functions of a DBMS:

Create databases.

Create tables.

Create supporting structures.

Read database data.

Modify database data (insert, update, delete).

Maintain database structures.

Enforce rules.

Control concurrency Provide security.

Perform backup and recovery.

Referential Integrity Constraints:

The DBMS will enforce many constraints.

Referential integrity constraints ensure that the values of a column in one table are valid based on the values in another table.

If a 5 was entered as a CustomerID in the PROJECT table, a customer having a CustomerID value of 5 must exist in the CUSTOMER table.

Self-Describing Nature of a Database System:

Database system contains complete definition of structure and constraints.

((يحتوي نظام قواعد البيانات على تعريف كامل للهيكل والقيود))

Meta-data:

Describes structure of the database.

((يصف هيكل لقاعدة البيانات))

Database catalog used by:

DBMS software.

Database users who need information about database structure.

((مستخدمو قاعدة البيانات الذين يحتاجون إلى معلومات حول بنية قاعدة البيانات))

Insulation Between Programs and Data:

Program-data independence:

Structure of data files is stored in DBMS catalog separately from access programs.

((يتم تخزين بنية ملفات البيانات في ملف DBMS بشكل منفصل عن برامج الوصول))

Program-operation independence.

((استقلالية تشغيل البرنامج))

Operations, specified in two parts:

Interface includes operation name and data types of its arguments.

((تتضمن الواجهة اسم العملية وأنواع البيانات لوسائطها))

Implementation can be changed without affecting the interface.

((يمكن تغيير التنفيذ دون التأثير على الواجهة))

Database Applications:

A database application is a set of one or more computer programs that serves as an intermediary between the user and the DBMS.

((تطبيق قاعدة البيانات هو مجموعة من واحد أو أكثر من برامج الكمبيوتر التي تعمل كوسيط بين المستخدم ونظام إدارة قواعد البيانات))

Functions of Database Applications:

- Create and process forms.
- Process user queries.
- Create and process reports.
- Execute application logic.
- Control database applications.

Desktop Database Systems:

Desktop database systems typically:

Have one application.

Have only a few tables.

Are simple in design.

Involve only one computer.

Support one user at a time.

((من الخصائص لهذا النوع انه يوجد لديك تطبيق واحد وأيضا عدد قليل من الجداول يدعم جهاز واحد وبرنامج واحد وهو بسيط في التصميم))

Commercial DBMS Products:

Example Desktop DBMS Products:

1. Microsoft Access.

Example Organizational DBMS Products:

1. Oracle's Oracle.
2. Microsoft's SQL Server.
3. IBM's DB2.

Chapter 2

Three-Schema Architecture:

Internal Schema:

Describes physical storage structure of the database.

((يصف هيكل التخزين المادي لقاعدة البيانات))

has an internal schema, which describes the physical storage structure of the database. The internal schema uses a physical data model and describes the complete details of data storage and access paths for the database.

((يحتوي على مخطط داخلي يصف بنية التخزين الفعلية لقاعدة البيانات. يستخدم المخطط الداخلي نموذج بيانات مادي ويصف التفاصيل الكاملة لتخزين البيانات ومسارات الوصول لقاعدة البيانات))

Conceptual Schema:

Describes structure of the whole database for a community of users.

((يصف هيكل قاعدة البيانات بأكملها لمجتمع من المستخدمين))

has a conceptual schema, which describes the structure of the whole database for a community of users. The conceptual schema hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints. This is usually specified using a DBMS such as Oracle, MySQL, … etc.

((يحتوي على مخطط مفاهيمي يصف هيكل قاعدة البيانات بأكملها لمجتمع من المستخدمين. يخفي المخطط المفاهيمي تفاصيل هياكل التخزين المادية ويركز على وصف الكيانات وأنواع البيانات، والعلاقات، وعمليات المستخدم، والقيود. يتم تحديد ذلك عادةً باستخدام DBMS مثل Oracle وMySQL ... إلخ))

Usually, a representational data model is used to describe the conceptual schema when a database system is implemented. The representational data model is a high-level data model.

((عادة، يتم استخدام نموذج البيانات التمثيلية لوصف المخطط المفاهيمي عند تنفيذ نظام قاعدة البيانات. نموذج البيانات التمثيلية هو نموذج بيانات عالي المستوى))

External or view Schema:

Describes part of the database that a particular user group is interested in.

((يصف جزءًا من قاعدة البيانات تهتم به مجموعة مستخدمين معينة))

It's also called view level which includes a few external schemas or user views. Each external schema describes the part of the database that a particular user group is interested in and hides the rest of the database from that user group.

((يطلق عليه مستوى العرض والذي يتضمن أيضًا عددًا من المخططات الخارجية أو طرق عرض المستخدم. يصف كل مخطط خارجي جزء قاعدة البيانات الذي تهتم به مجموعة مستخدمين معينة ويخفي بقية قاعدة البيانات من مجموعة المستخدمين هذه))

As in the previous level, each external schema is typically implemented using a conceptual data model, possibly based on an external schema design in a high-level data model.

((كما في المستوى السابق، يتم تنفيذ كل مخطط خارجي عادةً باستخدام نموذج بيانات مفاهيمي، ربما يعتمد على تصميم مخطط خارجي في نموذج بيانات عالي المستوى))

that the three schemas are only descriptions of data; the stored data that exists is at the physical level only. In a DBMS based on the three-schema architecture, each user group refers to its own external schema.

((أن المخططات الثلاثة ليست سوى أوصاف البيانات؛ البيانات المخزنة الموجودة بالفعل هي على المستوى المادي فقط. في DBMS استنادًا إلى بنية المخططات الثلاثة، تشير كل مجموعة مستخدمين إلى مخططها الخارجي الخاص))

Hence, the DBMS must transform a request specified on an external schema into a request against the conceptual schema, and then into a request on the internal schema for processing over the stored database. If the request is a database retrieval, the data extracted from the stored database must be reformatted to match the user's external view.

((ومن ثم، يجب أن يقوم DBMS بتحويل طلب محدد في مخطط خارجي إلى طلب مقابل المخطط المفاهيمي، ثم إلى طلب على المخطط الداخلي للمعالجة عبر قاعدة البيانات المخزنة. إذا كان الطلب عبارة عن استرداد لقاعدة بيانات، فيجب إعادة تنسيق البيانات المستخرجة من قاعدة البيانات المخزنة لتتناسب مع العرض الخارجي للمستخدم))

Mapping:

The processes of transforming requests and results between levels are called mappings.

((هي عملية الربط بين النتائج والمستويات))

Chapter 3

Transaction Processing:

Single-User System:

At most one user at a time can use the system.

((يمكن لمستخدم واحد على الأكثر استخدام النظام))

Multiuser System:

Many users can access the system concurrently.

((يمكن للعديد من المستخدمين الوصول إلى النظام في نفس الوقت))

Concurrency

Interleaved processing:

Concurrent execution of processes is interleaved in a single CPU.

((التنفيذ المتزامن للعمليات معشق في وحدة معالجة مركزية واحدة))

Parallel processing:

Processes are concurrently executed in multiple CPUs.

((يتم تنفيذ العمليات بشكل متزامن في وحدات معالجة مركزية متعددة))

A Transaction:

Logical unit of database processing that includes one or more access operations (read -retrieval, write - insert or update, delete).

((وحدة منطقية لمعالجة قاعدة البيانات تتضمن عملية وصول واحدة أو أكثر (قراءة - استرجاع، كتابة - إدراج أو تحديث، حذف)))

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.

((قد تكون المعاملة (مجموعة العمليات) قائمة بذاتها محددة بلغة عالية المستوى مثل SQL المقدمة بشكل تفاعل، أو قد تكون مضمنة في برنامج))

Transaction boundaries:

Beginning and End transaction.

An application program may contain several transactions separated by the Begin and End transaction boundaries.

((قد يحتوي برنامج التطبيق على عدة معاملات مفصولة بحدود بدء المعاملة والانتهاء))

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

A database is a collection of named data items.

((قاعدة البيانات هي مجموعة من عناصر البيانات المسماة))

Granularity of data – a field, a record, or a whole disk block (Concepts are independent of granularity).

((دقة البيانات - حقل أو سجل أو كتلة قرص كاملة (المفاهيم مستقلة عن التفاصيل)))

Basic operations are reading and writing.

((تتم قراءة وكتابة العمليات الأساسية))

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.

((read_item (X): يقرأ عنصر قاعدة بيانات يسمى X في متغير برنامج. لتبسيط الترميز الخاص بنا، نفترض أن متغير البرنامج يسمى أيضًا X))

write_item(X): Writes the value of program variable X into the database item named X.

((write_item (X): يكتب قيمة متغير البرنامج X في عنصر قاعدة البيانات المسمى X))

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.

((الوحدة الأساسية لنقل البيانات من القرص إلى الذاكرة الرئيسية للحاسوب هي كتلة واحدة. بشكل عام، سيكون عنصر البيانات (ما يقرأ أو يكتب) مجالًا لبعض السجلات في قاعدة البيانات، على الرغم من أنه قد يكون وحدة أكبر مثل السجل أو حتى كتلة كاملة))

read_item(X) command includes the following steps:

Find the address of the disk block that contains item X.

((يتضمن الأمر read_item (X) الخطوات التالية:

ابحث عن عنوان كتلة القرص التي تحتوي على العنصر X))

Copy that disk block into a buffer in main memory (if that disk block is not already in some main memory buffer).

((انسخ كتلة القرص هذه إلى مخزن مؤقت في الذاكرة الرئيسية (إذا لم تكن كتلة القرص هذه موجودة بالفعل في بعض مخازن الذاكرة الرئيسية)))

Copy item X from the buffer to the program variable named X.

((انسخ العنصر X من المخزن المؤقت إلى متغير البرنامج المسمى X))

write_item(X) command includes the following steps:

Find the address of the disk block that contains item X.

((ابحث عن عنوان كتلة القرص التي تحتوي على العنصر X))

Copy that disk block into a buffer in main memory (if that disk block is not already in some main memory buffer).

((انسخ كتلة القرص هذه إلى مخزن مؤقت في الذاكرة الرئيسية (إذا لم تكن كتلة القرص هذه موجودة بالفعل في بعض مخازن الذاكرة الرئيسية)))

Copy item X from the program variable named X into its correct location in the buffer.

((انسخ العنصر X من متغير البرنامج المسمى X إلى موقعه الصحيح في المخزن المؤقت))

Store the updated block from the buffer back to disk (either immediately or at some later point in time).

((قم بتخزين الكتلة المحدثة من المخزن المؤقت إلى القرص (إما على الفور أو في وقت لاحق في وقت لاحق)))

Why Concurrency:

When a transaction is performing an I/O operation, the CPU is idle. So, why not assign another transaction to CPU (Higher Efficiency).

((عندما تقوم إحدى المعاملات بتنفيذ عملية إدخال / إخراج، تكون وحدة المعالجة المركزية في وضع الخمول. لذا، لماذا لا يتم تخصيص معاملة أخرى لوحدة المعالجة المركزية (كفاءة أعلى)))

Suppose that one transaction needs a long time to conclude, then it is better to postpone it for a while and let other transactions work (Higher Productivity).

((لنفترض أن معاملة واحدة تحتاج إلى وقت طويل لإتمامها، فمن الأفضل تأجيلها لفترة والسماح للمعاملات الأخرى بالعمل (إنتاجية أعلى)))

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.

((يحدث هذا عندما تكون هناك عمليتان تصلان إلى عناصر قاعدة البيانات نفسها معشق عملياتهما بطريقة تجعل قيمة بعض عناصر قاعدة البيانات غير صحيحة))

The Temporary Update (or Dirty Read) Problem:

This occurs when one transaction updates a database item and then the transaction fails for some reason.

The updated item is accessed by another transaction before it is changed back to its original value.

((يحدث هذا عندما تقوم إحدى المعاملات بتحديث عنصر قاعدة البيانات ثم تفشل المعاملة لسبب ما.

يتم الوصول إلى العنصر الذي تم تحديثه من خلال معاملة أخرى قبل أن يتم تغييره مرة أخرى إلى قيمته الأصلية))

The Incorrect Summary Problem:

If one transaction is calculating an aggregate summary function on several 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.

((إذا كانت إحدى المعاملات تحسب دالة تلخيصيه مجمعة على عدد من السجلات بينما تقوم حركات أخرى بتحديث بعض هذه السجلات، فقد تحسب الدالة التجميعية بعض القيم قبل تحديثها والبعض الآخر بعد تحديثها))

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.

((يحدث خطأ في الأجهزة أو البرامج في نظام الحاسوب أثناء تنفيذ المعاملة. في حالة تعطل الجهاز، فقد يتم فقد محتويات الذاكرة الداخلية للحاسوب))

2- A transaction or system error:

Some operations 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.

((قد تؤدي بعض العمليات في المعاملة إلى فشلها، مثل تجاوز عدد صحيح أو القسمة على صفر. قد يحدث فشل المعاملة أيضًا بسبب قيم معلمات خاطئة أو بسبب خطأ برمجة منطقي. بالإضافة إلى ذلك، يجوز للمستخدم مقاطعة المعاملة أثناء تنفيذها))

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.

((تتطلب بعض الشروط إلغاء الصفقة. على سبيل المثال، قد لا يتم العثور على بيانات المعاملة. قد يتسبب شرط، مثل رصيد الحساب غير الكافي في قاعدة بيانات مصرفية، في إلغاء معاملة، مثل سحب الأموال من هذا الحساب.

يؤدي الإجهاض المبرمج في المعاملة إلى فشلها))

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.

((قد تقرر طريقة التحكم في التزامن إجهاض المعاملة، أو إعادة تشغيلها لاحقًا، لأنها تنتهك قابلية التسلسل أو لأن العديد من المعاملات في حالة توقف تام))

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.

((قد تفقد بعض كتل الأقراص بياناتها بسبب عطل في القراءة أو الكتابة أو بسبب تعطل رأس قراءة / كتابة القرص. قد يحدث هذا أثناء عملية قراءة أو كتابة للمعاملة))

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.

((يشير هذا إلى قائمة لا حصر لها من المشاكل التي تشمل فشل الطاقة أو تكييف الهواء، والحريق، والسرقة، والتخريب، والكتابة فوق الأقراص أو الأشرطة عن طريق الخطأ، وتركيب شريط خاطئ من قبل المشغل))

Transaction and System Concepts:

A transaction is an atomic unit of work that is either completed in its entirety or not done at all.

((transaction هي وحدة ذرية من العمل التي إما أن تكتمل بالكامل أو لم يتم إجراؤها على الإطلاق))

For recovery purposes, the system needs to keep track of when the transaction starts, terminates, and commits or aborts.

((لأغراض الاسترداد، يحتاج النظام إلى تتبع وقت بدء المعاملة وانتهائها وارتكابها أو فشلها))

Transaction states:

1. Active state.
2. Partially committed state.
3. Committed state.
4. Failed state.
5. Terminated State.

Recovery manager keeps track of the following operations:

begin_transaction: This marks the beginning of transaction execution.

((begin_transaction: يمثل هذا بداية تنفيذ المعاملة))

read or write: These specify read or write operations on the database items that are executed as part of a transaction.

((القراءة أو الكتابة: تحدد عمليات القراءة أو الكتابة على عناصر قاعدة البيانات التي يتم تنفيذها كجزء من المعاملة))

end_transaction: This specifies that read and write transaction operations have ended and marks the end limit of transaction execution.

((end_transaction: هذا يحدد انتهاء عمليات قراءة وكتابة المعاملات ويمثل حدًا نهائيًا لتنفيذ المعاملة))

At this point it may be necessary to check whether the changes introduced by the transaction can be permanently applied to the database or whether the transaction must be aborted because it violates concurrency control or for some other reason.

((في هذه المرحلة، قد يكون من الضروري التحقق مما إذا كان يمكن تطبيق التغييرات التي أدخلتها المعاملة بشكل دائم على قاعدة البيانات أو ما إذا كان يجب إلغاء المعاملة لأنها تنتهك التحكم في التزامن أو لسبب آخر))

commit_transaction: This signals a successful end of the transaction so that any changes (updates) executed by the transaction can be safely committed to the database and will not be undone.

((يشير هذا إلى نهاية ناجحة للمعاملة بحيث يمكن الالتزام بأمان بأي تغييرات (تحديثات) تنفذها المعاملة بقاعدة البيانات ولن يتم التراجع عنها))

rollback (or abort): This signals that the transaction has ended unsuccessfully, so that any changes or effects that the transaction may have applied to the database must be undone.

((يشير هذا إلى أن المعاملة قد انتهت دون نجاح، لذلك يجب التراجع عن أي تغييرات أو تأثيرات قد تكون قد طبقتها المعاملة على قاعدة البيانات))

undo: Like rollback except that it applies to a single operation rather than to a whole transaction.

((يشبه التراجع باستثناء أنه ينطبق على عملية واحدة بدلاً من المعاملة بأكملها))

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.

((يحدد هذا أنه يجب إعادة إجراء عمليات معاملة معينة لضمان تطبيق جميع عمليات المعاملة الملتزم بها بنجاح على قاعدة البيانات))

The System Log

Log or Journal: The log keeps track of all transaction operations that affect the values of database items.

((يتتبع السجل جميع عمليات المعاملات التي تؤثر على قيم عناصر قاعدة البيانات))

This information may be needed to permit recovery from transaction failures.

((قد تكون هناك حاجة إلى هذه المعلومات للسماح بالاسترداد من فشل المعاملات))

The log is kept on disk, so it is not affected by any type of failure except for disk or catastrophic failure.

((يتم الاحتفاظ بالسجل على القرص، لذلك لا يتأثر بأي نوع من الفشل باستثناء القرص أو فشل ذريع))

In addition, the log is periodically backed up to archival storage (tape) to guard against such catastrophic failures.

((بالإضافة إلى ذلك، يتم نسخ السجل احتياطيًا بشكل دوري إلى التخزين الأرشيفي (الشريط) للحماية من مثل هذه الإخفاقات الكارثية))

Types of log record:

T in the following discussion refers to a unique transaction-id that is generated automatically by the system and is used to identify each transaction:

((يشير حرف T الى معرف يتم إنشاؤه من قبل النظام بشكل تلقائي))

[start_transaction,T]: Records that transaction T has started execution.

((يسجل أن المعاملة T بدأ تنفيذها))

[write_item,T,X,old_value,new_value]: Records that transaction T has changed the value of database item X from old_value to new_value.

((سجلات تلك المعاملة T غيرت قيمة عنصر قاعدة البيانات X من old_value إلى new_value))

[read_item,T,X]: Records that transaction T has read the value of database item X.

((يسجل أن المعاملة T قد قرأت قيمة عنصر قاعدة البيانات X))

[commit,T]: Records that transaction T has completed successfully, and affirms that its effect can be committed (recorded permanently) to the database.

((تعني ان العملية قد تم تنفيذها بنجاح ويسجل تأثيرها في قاعدة البيانات))

[abort,T]: Records that transaction T has been aborted.

((تعني ان العملية لم تتم وسيتم احباطها وإعادة نفيذها rollback))

Protocols for recovery that avoid cascading rollbacks do not require that read operations be written to the system log, whereas other protocols require these entries for recovery.

((لا تتطلب بروتوكولات الاسترداد التي تتجنب التراجع المتتالي كتابة عمليات القراءة في سجل النظام، بينما تتطلب البروتوكولات الأخرى هذه الإدخالات للاسترداد))

Strict protocols require simpler write entries that do not include new_value.

((تتطلب البروتوكولات الصارمة إدخالات كتابة أبسط لا تتضمن new_value))

If the system crashes, we can recover to a consistent database state by examining the log and using one of the techniques described in.

((في حالة تعطل النظام، يمكننا استعادة حالة قاعدة بيانات ثابتة عن طريق فحص السجل واستخدام إحدى التقنيات الموضحة في))

Because the log contains a record of every write operation that changes the value of some database item, it is possible to undo the effect of these write operations of a transaction T by tracing backward through the log and resetting all items changed by a write operation of T to their old_values.

((نظرًا لأن السجل يحتوي على سجل لكل عملية كتابة تقوم بتغيير قيمة بعض عناصر قاعدة البيانات، فمن الممكن التراجع عن تأثير عمليات الكتابة هذه للمعاملة T من خلال التتبع للخلف من خلال السجل وإعادة تعيين جميع العناصر التي تم تغييرها بواسطة عملية كتابة لـ T لقيمهم القديمة))

We can also redo the effect of the write operations of a transaction T by tracing forward through the log and setting all items changed by a write operation of T (that did not get done permanently) to their new_values.

((يمكننا أيضًا إعادة تأثير عمليات الكتابة للمعاملة T من خلال التتبع للأمام من خلال السجل وتعيين جميع العناصر التي تم تغييرها بواسطة عملية الكتابة T (التي لم يتم تنفيذها بشكل دائم) إلى قيمها الجديدة))

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.

((تصل المعاملة T إلى نقطة الالتزام الخاصة بها عندما يتم تنفيذ جميع عملياتها التي تصل إلى قاعدة البيانات بنجاح ويتم تسجيل تأثير جميع عمليات المعاملات على قاعدة البيانات في log file))

Beyond the commit point, the transaction is said to be committed, and its effect is assumed to be permanently recorded in the database.

((بعد نقطة الالتزام، يُقال إن المعاملة ملتزمة، ويفترض أن يتم تسجيل تأثيرها بشكل دائم في قاعدة البيانات))

The transaction then writes an entry [commit,T] into the log.

((تقوم المعاملة بعد ذلك بكتابة إدخال [الالتزام، T] في السجل))

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.

((مطلوب للمعاملات التي تحتوي على إدخال [start_transaction، T] في السجل، ولكن لا يوجد إدخال التزام [الالتزام، T] في السجل))

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.

((يجب ان يكون العمليات جميعها مسجلة بالسجل، خلاف ذلك يمكن إعادة تأثيرها على قاعدة البيانات من خلال السجلات))

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.)

((في وقت تعطل النظام، يتم اعتبار إدخالات السجل التي تمت كتابتها مرة أخرى على القرص فقط في عملية الاسترداد لأنه قد يتم فقد محتويات الذاكرة الرئيسية))

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.

((قبل أن تصل المعاملة إلى نقطة الالتزام الخاصة بها، يجب الآن كتابة أي جزء من السجل لم تتم كتابته على القرص بعد على القرص))

This process is called force-writing the log file before committing a transaction.

((تسمى هذه العملية بالقوة كتابة ملف السجل قبل تنفيذ أي معاملة))

ACID properties:

Atomicity: A transaction is an atomic unit of processing; it is either performed in its entirety or not performed at all.

((الصفقة هي وحدة ذرية للمعالجة؛ يتم إجراؤها إما بالكامل أو لا يتم إجراؤها على الإطلاق))

(Transaction Recovery Subsystem)

Consistency preservation: A correct execution of the transaction must take the database from one consistent state to another.

((يجب أن يأخذ التنفيذ الصحيح للمعاملة قاعدة البيانات من حالة متسقة إلى أخرى))

Consistency Requirement responsible Developers.

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.

((يجب ألا تجعل المعاملة تحديثاتها مرئية للمعاملات الأخرى حتى يتم الالتزام بها؛ هذه الخاصية، عند فرضها بصرامة، تحل مشكلة التحديث المؤقتة وتجعل التراجع المتتالي للمعاملات غير ضروري))

(Concurrency control subsystem)

Durability or permanency: Once a transaction changes the database and the changes are committed, these changes must never be lost because of subsequent failure.

((بمجرد تغيير المعاملة لقاعدة البيانات وتنفيذ التغييرات، يجب ألا تُفقد هذه التغييرات أبدًا بسبب الفشل اللاحق))

(Recovery subsystem)

Characterizing Schedules based on Recoverability:

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).

((عندما يتم تنفيذ المعاملات بشكل متزامن بطريقة متداخلة ، فإن ترتيب تنفيذ العمليات من المعاملات المختلفة يشكل ما يعرف بجدول المعاملات (أو التاريخ)))

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.

r1(X) means Transaction 1 is reading the value of X from DB.

w1(X) means Transaction 1 is writing the value of X to DB.

c1 means Transaction 1 commits.

a1 means Transaction 1 aborts.

Recoverable schedule:

One where no transaction needs to be rolled back.

((t1لا تحتاج الى عملية rollback))

A schedule S is recoverable if no transaction T in S commits until all transactions T’ that have written an item that T reads have committed.

((يكون الجدول S قابلاً للاسترداد إذا لم يتم تنفيذ أي معاملة T في S حتى تلتزم جميع المعاملات T التي كتبت عنصرًا تقرأه T))

Cascadeless schedule:

One where every transaction reads only the items that are written by committed transactions.

((واحد حيث تقرأ كل معاملة فقط العناصر التي تمت كتابتها من خلال المعاملات الملتزمة))

Non-Recoverable Schedule:

Condition 1 not satisfied:

T1 aborted and therefore it will rollback.

Condition 2: not satisfied.

T2 read X after T1 has written it to Disk.

But, T2 has committed before T1, and T1 has aborted which means that the value of x read by T1 is false.

((في النقطة الأولى لم يتم تنفيذ T1 لذا تمت عملية rollback لها))

((في النقطة الثانية تم أيضا عدم تنفيذ T2 بسبب فشل T1 حتى وأنها قد تمت العملية بنجاح لكن قيمة X لن تكون صحيحة ويتم عملية rollback))

Schedules requiring cascaded rollback:

A schedule in which uncommitted transactions that read an item from a failed transaction must be rolled back.

((جدول يجب فيه التراجع عن المعاملات غير الملتزم بها التي تقرأ عنصرًا من معاملة فاشلة))

Strict Schedules:

A schedule in which a transaction can neither read nor write an item X until the last transaction that wrote X has committed.

((جدول لا يمكن فيه للمعاملة قراءة عنصر X أو كتابته حتى يتم تنفيذ آخر معاملة كتبت X))

Conflicting Operations:

They belong to different transactions.

They access the same database item.

Their type of operations is.

Chapter 4

Database Concurrency Control:

Purpose of Concurrency Control:

-To enforce Isolation (through mutual exclusion) among conflicting transactions.

((لفرض العزلة (من خلال الاستبعاد المتبادل) بين المعاملات المتضاربة))

-To preserve database consistency through consistency preserving execution of transactions.

((للحفاظ على اتساق قاعدة البيانات من خلال الاتساق مع الحفاظ على تنفيذ المعاملات))

-To resolve read-write and write-write conflicts.

Locking is an operation which secures:

permission to Read.

permission to Write a data item for a transaction.

Unlocking is an operation which removes these permissions from the data item.

Lock and Unlock are atomic operations.

1- Binary Lock

Locked:

A lock for reading/writing data item.

Unlocked:

2- Shared/Exclusive Lock

Read_Locked: also called shared.

A lock for reading data items.

Write_Locked: also called exclusive.

A lock for reading/writing data items.

Unlocked.

في هذه النقطة يتم شرح عملية lock داخل قاعدة البيانات والتي لديها نوعين read\shared lock المختصة بعمليات قراءة البيانات فقط اما النوع الثاني write\exclusive والذي يستعمل عند تعديل البيانات من قاعدة البيانات الرئيسية))

Binary Locks are too restrictive:

Binary Locks disallow a transaction (T2) from reading a data item if another transaction (T1) is holding a lock on that item.

But, if T1 is only reading the data item, then there is no harm in letting T2 read the same item because read operations do not conflict with each other.

((لا يسمح لاي عملية بتعديل المتغير في حال كانت هناك عملية أخرى على نفس العنصر لكن يتم السماح بحجز نفس العنصر لأكثر من عملية إذا كان العملية عبارة عن عملية القراءة فقط لكلا العمليتين))

Two locks modes:

(a) shared (read).

(b) exclusive (write).

Shared mode: shared lock (X):

More than one transaction can apply share lock on X for reading its value, but no write lock can be applied on X by any other transaction.

((يمكن لأكثر من معاملة تطبيق قفل المشاركة على X لقراءة قيمتها، ولكن لا يمكن تطبيق قفل الكتابة على X بأي معاملة أخرى))

Exclusive mode: Write lock (X):

Only one write lock on X can exist at any time and no shared lock can be applied by any other transaction on X.

((يمكن أن يوجد قفل كتابة واحد فقط على X في أي وقت ولا يمكن تطبيق أي قفل مشترك بواسطة أي معاملة أخرى على X))

Conflict matrix:

	Read	Write
Read	Y	N
Write	N	N

((هذه المصفوفة تشرح مبدا عمل lockالتأثيرها على الconflict أي التقاء عملتين لا تلتقيان و يتبين ان العمليتين لا يتنفذان دون انتهاء الأخرى الا بحالة ان العمليتين هما عمليتين قراءة فقط دون تعديل أي شيء على العنصر))

Lock Manager:

Managing locks on data items.

Lock table:

Lock manager uses it to store the identify of transaction locking a data item, the data item, lock mode and pointer to the next data item locked. One simple way to implement a lock table is through a linked list.

((يستخدم من قبل مديرو قواعد البيانات وهو جدول يظهر الأمور المتعلقة بال lock مثل المؤشر ووضع القفل))

Database requires that all transactions should be well-formed. A transaction is well-formed if:

It must lock the data item before it reads or writes to it.

It must not lock an already locked data item and it must not try to unlock a free data item.

((بكل بساطة يجب عمل lockقبل البدء باي عملية سواء كانت قراءة ام تعديل لكن في حال التعديل لا يمكن لاي عمليه عمل lock على نفس العنصر))

Lock upgrade:

A transaction T1 can convert a read_lock(x) to write_lock(x) if it holds a read_lock on data item x and

No other transactions are holding a read_lock on x.

Otherwise, it waits until no transaction is holding a read_lock on x.

This happens when T1 wants to perform a write operation on x.

((تسمى العملية التي تقوم بتحويل العملية من read-lock الى write-lock ترقية او upgrade))

Lock downgrade:

A transaction T1 can convert a write_lock(x) to read_lock(x) if

It holds a write_lock on data item x. Remember that only one transaction can hold a write_lock on the same dat item.

This happens when the rest of operations on x needed for T1 are read operations.

((اما العملية العكسية أي تحويل من write-lock الى read-lock التراجع او downgrade))

Two-Phase Locking (2PL):

All locking operations in a transaction must precede the first unlock operation.

((يجب أن تسبق جميع عمليات القفل في المعاملة عملية إلغاء القفل الأولى))

Two Phases:

(a) Locking (Growing)

(b) Unlocking (Shrinking).

Locking (Growing) Phase:

A transaction applies locks (read or write) on desired data items one at a time.

This phase happens first.

((تطبق المعاملة الأقفال (القراءة أو الكتابة) على عناصر البيانات المطلوبة واحدًا تلو الآخر))

Unlocking (Shrinking) Phase:

A transaction unlocks its locked data items one at a time.

This phase happens second.

((تفتح المعاملة عناصر البيانات المقفلة الخاصة بها واحدة تلو الأخرى))

Requirement:

For a transaction these two phases must be mutually exclusively, that is, during locking phase unlocking phase must not start and during unlocking phase locking phase must not begin.

((بالنسبة للمعاملة، يجب أن تكون هاتان المرحلتان حصريًا، أي أثناء مرحلة القفل، يجب ألا تبدأ مرحلة فتح القفل ويجب ألا تبدأ مرحلة قفل طور الفتح))

Two-phase policy generates two locking algorithms.

(a) Basic

(b) Conservative

Conservative:

Prevents deadlock by locking all desired data items before transaction begins execution.

((يمنع deadlock عن طريق قفل جميع عناصر البيانات المطلوبة قبل بدء تنفيذ المعاملة))

Basic:

Transaction locks data items incrementally. This may cause a deadlock which is dealt with.

((تقوم المعاملة بتأمين عناصر البيانات بشكل متزايد. قد يتسبب هذا في طريق مسدود يتم التعامل معه))

Strict:

A stricter version of Basic algorithm where unlocking write_locks is performed after a transaction terminates (commits or aborts and rolled-back). This is the most used two-phase locking algorithm.

((بالمعنى عدم عمل الغاء لعملية القفل الا عند عمل commit او rollback للعملية))

It is important to know the difference between

1. Deadlock Prevention:
   A transaction locks all data items it refers to before it begins execution.
   ((تقوم المعاملة بتأمين جميع عناصر البيانات التي تشير إليها قبل أن تبدأ في التنفيذ))
   This way of locking prevents deadlock since a transaction never waits for a data item.
   ((تمنع طريقة القفل هذه حدوث طريق مسدود لأن المعاملة لا تنتظر أبدًا عنصر بيانات))
   The conservative two-phase locking uses this approach.
   ((يستخدم القفل المحافظ على مرحلتين هذا النهج))
2. Deadlock Detection and Resolution:
   In this approach, deadlocks are allowed to happen. The scheduler maintains a wait-for-graph for detecting cycle. If a cycle exists, then one transaction involved in the cycle is selected (victim) and rolled back.
   ((في هذا النهج، يُسمح بحدوث الجمود. يحافظ المجدول على انتظار رسم بياني للكشف عن الدورة. في حالة وجود دورة، يتم تحديد معاملة واحدة متضمنة في الدورة (الضحية) وإعادتها إلى الوراء))
   A wait-for-graph is created using the lock table. As soon as a transaction is blocked, it is added to the graph. When a cycle exists, then one transaction involved in the cycle is selected (victim) and rolled back.
   ((يتم إنشاء انتظار الرسم البياني باستخدام جدول القفل. بمجرد حظر المعاملة، يتم إضافتها إلى الرسم البياني. عندما توجد دورة، يتم تحديد معاملة واحدة متضمنة في الدورة (الضحية) والتراجع))
3. Deadlock Avoidance:

There are many variations of the two-phase locking algorithm. Some avoid deadlock by not letting the cycle complete.

((يتجنب البعض deadlock من خلال عدم ترك الدورة تكتمل))

That is as soon as the algorithm discovers that blocking a transaction is likely to create a cycle, it rolls back the transaction.

((هذا بمجرد أن تكتشف الخوارزمية أن حظر المعاملة من المحتمل أن يؤدي إلى إنشاء دورة، فإنها تتراجع عن المعاملة))

Wound-Wait and Wait-Die algorithms use timestamps to avoid deadlocks by rolling-back victims.

((تستخدم خوارزميات Wound-Wait وWait-Die طوابع زمنية لتجنب المآزق عن طريق عودة الضحية))

Time Stamps:

A monotonically increasing variable (integer) indicating the age of an operation or a transaction. A larger timestamp value indicates a more recent event or operation.

((متغير متزايد بشكل رتيب (عدد صحيح) يشير إلى عمر عملية أو معاملة. تشير قيمة الطابع الزمني الأكبر إلى حدث أو عملية أحدث))

Wait-Die:

Suppose T1 is not able to obtain a lock on X because X is locked by T2

If TS(T1) < TS(T2) (T1 is older than T2)

T1 is allowed to wait.

Otherwise (T1 is younger than T2)

T1 is aborted and rolled back and it will be started later with the same time stamp.

((عندما تكون العملية الأولى أصغر من العملية الثانية يتم ذهاب العملية الأولى الى وضع الانتظار))

((غير هذه الحالة يتم abort للعملية الأولى ثم rollback ويتم الخول مرة أخرى للنظام في نفس time stamp))

Wound-Wait:

Suppose T1 is not able to obtain a lock on X because X is locked by T2.

If TS(T1) < TS(T2) (T1 is older than T2)

T2 is aborted and rolled back, and it will be started later with the same time stamp.

Otherwise (T1 is younger than T2)

T1 is allowed to wait.

((هذه الحالة هي معاكسة لحالة Wait-Die حيث ان العملية الأولى أصغر من العملية الثانية يتم عمل rollback وفي الحالة الثانية أي أصغر او يساوي العملية الثانية فيتم ذهاب العملية الأولى الى عملية الانتظار))

Starvation:

Starvation occurs when a particular transaction consistently waits or restarts and never gets a chance to proceed further.

In a deadlock resolution it is possible that the same transaction may consistently be selected as victim and rolled-back.

This limitation is inherent in all priority-based scheduling mechanisms.

In Wound-Wait scheme a younger transaction may always be wounded (aborted) by a long-running older transaction which may create starvation.

((هنا وفي هذه الحالة يتم طلب الدخول الى احد العناصر و لكن يكون هذا العنصر مشغول فتذهب العملية الى وضع الانتظار ويتم إعادة تشغيلها و لا تتاح لها الفرصة لتبدا العملية وتسمى هذه المشكلة starvation))

in a Basic Time Stamp Algorithm, a Time Stamp is given to data items

Read_TS(X): The time stamp of the last transaction to read X.

Write_TS(X): The time stamp of the last transaction to write X.

((يقسم الtime stamp الى قسمين أحدهما للقراءة read-timestamp والاخر للتعديل ويسمى write-timestamp))

Timestamp based concurrency control algorithm:

Basic Timestamp Ordering

Transaction T issues a write_item(X) operation:

If read_TS(X) > TS(T) or if write_TS(X) > TS(T), then an younger transaction has already read the data item so abort and roll-back T and reject the operation.

If the condition in part (a) does not exist, then execute write_item(X) of T and set write_TS(X) to TS(T).

((هناك حالتين في عمليات time-stamp بالنسبة لعمليات التعديل فانه يتم التأكد من العملية مرتين مع time-stamp للعملية مرة مع timestamp للتعديل لنفس العنصر اما الأخرى فهي مقارنة مع time-stamp الخاص بالقراءة لنفس العنصر))

((تتم المقارنة بين time-stamp للعملية مع ال timestampالخاص لعمليتي القراءة و التعديل كما ذكرنا سابقا اذا كان time stamp للعملية اصغر فيتم ازاله العملية rejected و عمل rollback اما ال-time-stamp فيتم ادخال قيمة جديدة للعملية))

((هناك حاله واحدة وهو اما ان يكون time-stamp للعملية او القراءة ة الكتابة اما متساويين او ان العملية هي التي تكون أكبر فهنا يتم اكمال العملية بنجاح و يتم تغيير write-timestmp))

Transaction T issues a read_item(X) operation:

If write_TS(X) > TS(T), then an younger transaction has already written to the data item so abort and roll-back T and reject the operation.

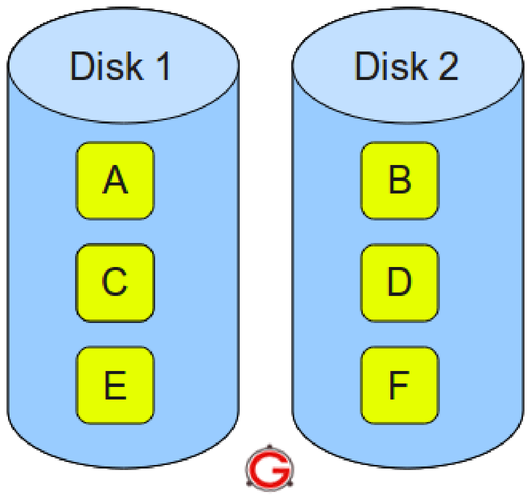
If write_TS(X) ≤ TS(T), then execute read_item(X) of T and set read_TS(X) to the larger of TS(T) and the current read_TS(X).

((حالة القراءة أسهل من حالة التعديل حيث يتم المقارنة مرة واحدة فقط مع بين transaction-timestamp وبين write-timestamp وهو بنفس خطوات التي ذكرناها سابقا حيث يتم انجاز العملية اذا كان الtime stamp للعملية اكبر او يساويtime-stamp من الخاص بالتعديلwrite-timestamp الى ان القيمة يتم وضعها في read-timestamp))

Chapter 5

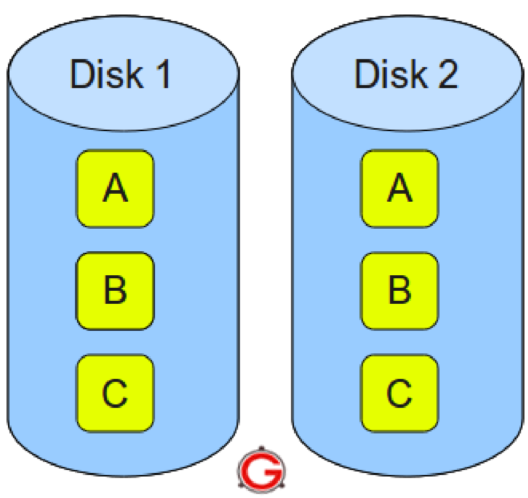
RAID :

RAID 0:



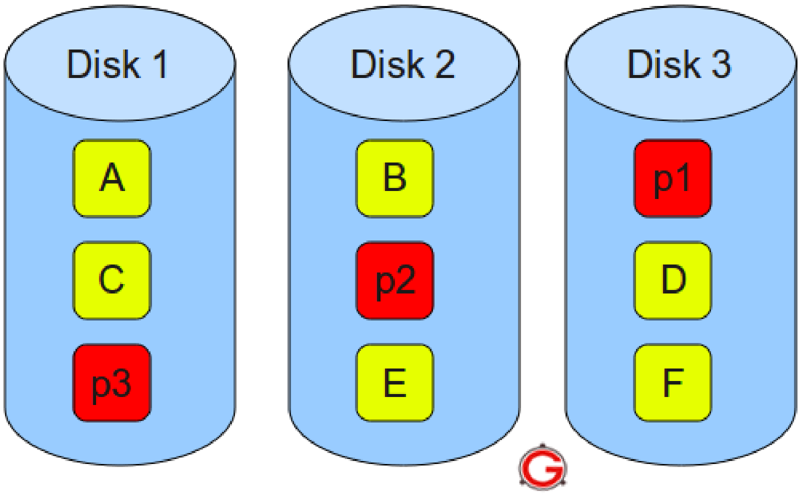
RAID 0 – Blocks Striped. No Mirror. No Parity.

RAID 1:



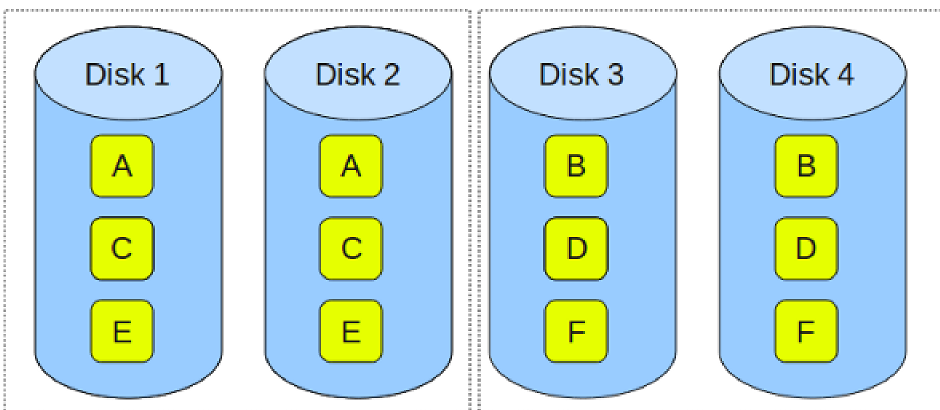
RAID 1 – Blocks Mirrored. No Stripe. No parity.

RAID 5:



RAID 5 – Blocks Striped. Distributed Parity.

RAID 10 (also known as RAID 1+0):



RAID 10 – Blocks Mirrored. (and Blocks Striped)

Failure Classification :

Transaction Failure :

Logical errors: transaction cannot be completed due to some internal error condition (bad input, data not found , overflow... etc.).

((لا يمكن إكمال المعاملة بسبب حالة خطأ داخلي (إدخال غير صحيح، بيانات غير موجودة، تجاوز، إلخ)))

System errors:

the database system must terminate an active transaction due to an error condition (e.g., deadlock).

((يجب أن ينهي نظام قاعدة البيانات معاملة نشطة بسبب حالة خطأ))

System crash:

A power failure or other hardware or software failure causes the system to crash and lose the content of volatile storage (in RAM) and bring transaction processing to a halt.

((يؤدي انقطاع التيار الكهربائي أو عطل آخر في الأجهزة أو البرامج إلى تعطل النظام وفقدان محتوى التخزين المتطاير (في ذاكرة الوصول العشوائي) وإيقاف معالجة المعاملات))

Fail-stop assumption:

non-volatile storage contents are assumed to not be corrupted by system crash.

((من المفترض ألا تتلف محتويات التخزين غير المتطايرة بسبب تعطل النظام))

Disk failure:

A head crash or similar disk failure destroys all or part of disk storage.

((يؤدي تعطل الرأس أو عطل مماثل في القرص إلى تدمير مساحة تخزين القرص بالكامل أو جزء منها))

Destruction is assumed to be detectable:

Disk drives use checksums to detect failures.

((تستخدم محركات الأقراص مجاميع اختبارية لاكتشاف الأعطال))

Recovery Algorithms:

Recovery algorithms are techniques to ensure database consistency and transaction atomicity and durability despite failures.

((خوارزميات الاسترداد هي تقنيات لضمان اتساق قاعدة البيانات وسلسلة المعاملات ومتانتها على الرغم من حالات الفشل))

Recovery algorithms have two parts:

1. Actions taken during normal transaction processing to ensure enough information exists to recover from failures.

((الإجراءات المتخذة أثناء معالجة المعاملات العادية لضمان وجود معلومات كافية للتعافي من حالات الفشل))

2. Actions taken after a failure to recover the database contents to a state that ensures atomicity, consistency, and durability.
   ((الإجراءات المتخذة بعد الفشل في استعادة محتويات قاعدة البيانات إلى حالة تضمن الذرية والاتساق والمتانة))

Transaction Log

For recovery from any type of failure data values prior to modification (BFIM - BeFore Image) and the new value after modification (AFIM – AFter Image) are required.

((للاسترداد من أي نوع من قيم بيانات الفشل قبل التعديل (BFIM - BeFore Image) ، ويلزم استخدام القيمة الجديدة بعد التعديل (AFIM - AFter Image)))

These values and other information is stored in a sequential file called Transaction log. A sample log is given below.

((يتم تخزين هذه القيم والمعلومات الأخرى في ملف تسلسلي يسمى سجل المعاملات))

Storage Structure:

To understand how to ensure the Atomicity and Durability properties of transaction we must gain a better understanding of these storage media and their access method.

((لفهم كيفية ضمان خصائص Atomicity and Durability للمعاملة، يجب أن نحصل على فهم أفضل لوسائط التخزين هذه وطريقة الوصول إليها))

The storage media can be classified by their:

Speed.

Capacity.

The ability to deal with failure.

They are classified in three classes:

Volatile:

does not survive system crashes.

((لا ينجو من أعطال النظام))

examples: main memory, cache memory.

The Access is extremely fast, because of the speed of the memory and the ability of direct access.

((الوصول سريع للغاية، بسبب سرعة الذاكرة والقدرة على الوصول المباشر))

Non- Volatile:

survives system crashes.

((ينجو من أعطال النظام))

examples: disk, magnetic tape, flash memory, non-volatile (battery backed up) RAM.

Disks are used for online storage; tapes are used for archival storage.

((تستخدم الأقراص للتخزين عبر الإنترنت، وتستخدم الأشرطة لتخزين الأرشفة))

Much slower than volatile storage.

((أبطأ بكثير من التخزين المتطاير))

Stable:

a mythical form of storage that survives all failures.

((شكل أسطوري للتخزين ينجو من كل الإخفاقات))

approximated by maintaining multiple copies on distinct nonvolatile media.

((تقريبًا عن طريق الاحتفاظ بنسخ متعددة على وسائط متميزة غير متطايرة))

Stable-Storage Implementation:

Maintain multiple copies of each block on separate disks copies can be at remote sites to protect against disasters such as fire or flooding.

((الاحتفاظ بنسخ متعددة من كل كتلة على أقراص منفصلة يمكن نسخ في مواقع بعيدة للحماية من الكوارث مثل الحريق أو الفيضانات))

Failure during data transfer can still result in inconsistent copies: Block transfer between memory and disk storage can result in:

((قد يؤدي الفشل أثناء نقل البيانات إلى نسخ غير متناسقة: يمكن أن يؤدي نقل الحظر بين الذاكرة وتخزين القرص إلى))

Successful completion:

The transferred information arrived safely at its destination.

((وصلت المعلومات المنقولة بشكل صحيح))

Partial failure:

A failure occurred during the transfer and the destination block has incorrect information.

((حدث فشل في وسط النقل وكانت كتلة الوجهة بها معلومات غير صحيحة))

Total failure:

The failure occurs very early during the transfer such that the destination block was never updated.

((يحدث الفشل في وقت مبكر جدًا أثناء النقل بحيث لم يتم تحديث كتلة الوجهة مطلقًا))

We require that if a data transfer failure occurs , the system detects it and invokes a recovery procedure to restore the block to a consistent state.

((نطلب أنه في حالة حدوث فشل في نقل البيانات، يكتشفه النظام ويستدعي إجراء استرداد لاستعادة الكتلة إلى حالة متسقة))

Protecting storage media from failure during data transfer (one solution):

((حماية وسائط التخزين من الفشل أثناء نقل البيانات (حل واحد)))

A possible solution:

Redundancy Execute output operation as follows (assuming two copies of each block):

التكرار نفذ عملية الإخراج على النحو التالي (بافتراض نسختين من كل كتلة)))

Write the information onto the first physical block.

((اكتب المعلومات على الكتلة المادية الأولى))

When the first write successfully completes, write the same information onto the second physical block.

((عندما تكتمل الكتابة الأولى بنجاح، اكتب نفس المعلومات على الكتلة المادية الثانية))

The output is completed only after the second write successfully completes.

((يكتمل الإخراج فقط بعد اكتمال الكتابة الثانية بنجاح))

Copies of a block may differ due to failure during output operation.

((قد تختلف نسخ الكتلة بسبب الفشل أثناء عملية الإخراج))

To recover from failure:

First find inconsistent blocks:

Expensive solution:

Compare the two copies of every disk block.

((قارن بين الكتل الموجودة))

Better solution:

Record in-progress disk writes on non-volatile storage (Non-volatile RAM or special area of disk).

((تسجيل القرص قيد التقدم يكتب على التخزين غير المتطاير (ذاكرة الوصول العشوائي غير المتطايرة أو منطقة خاصة من القرص)))

Use this information during recovery to find blocks that may be inconsistent, and only compare copies of these.

((استخدم هذه المعلومات أثناء الاسترداد للعثور على الكتل التي قد تكون غير متسقة، وقارن فقط نسخ منها))

Used in hardware RAID systems:

If either copy of an inconsistent block is detected to have an error (bad checksum), overwrite it by the other copy.

((إذا تم اكتشاف خطأ في أي من نسختين من الكتلة غير المتسقة (مجموع اختباري غير صحيح)، فاكتبها بالنسخة الأخرى))

If both have no error, but are different, overwrite the second block by the first block.

((إذا كان كلاهما لا يحتوي على أي خطأ، ولكنهما مختلفان، فاكتب الكتلة الثانية بواسطة الكتلة الأولى))

Data Access:

Physical blocks are those blocks residing on the disk.

((الكتل المادية هي تلك الكتل الموجودة على القرص))

Buffer blocks are the blocks residing temporarily in main memory.

((الكتل العازلة هي الكتل الموجودة مؤقتًا في الذاكرة الرئيسية))

Block movements between disk and main memory are initiated through the following two operations:

((تبدأ حركات الكتلة بين القرص والذاكرة الرئيسية من خلال العمليتين التاليتين))

input(B) transfers the physical block B to main memory.

((ينقل الإدخال (B) الكتلة المادية B إلى الذاكرة الرئيسية))

output(B) transfers buffer block B to the disk and replaces the appropriate physical block there.

((الإخراج (ب) ينقل كتلة المخزن B إلى القرص، ويحل محل الكتلة المادية المناسبة هناك))

Each transaction Ti has its private work-area in which local copies of all data items accessed and updated by it are kept.

((كل معاملة Ti لها منطقة عمل خاصة بها حيث يتم الاحتفاظ بنسخ محلية من جميع عناصر البيانات التي تم الوصول إليها وتحديثها بواسطتها))

Ti's local copy of a data item X is called xi.

((نسخة Ti المحلية من عنصر البيانات X تسمى xi))

We assume, for simplicity, that each data item fits in, and is stored inside, a single block.

((نحن نفترض، من أجل التبسيط، أن كل عنصر بيانات يتناسب مع كتلة واحدة ويتم تخزينها بداخلها))

Transaction transfers data items between system buffer blocks and its private work-area using the following operations :

((تنقل المعاملة عناصر البيانات بين كتل المخزن المؤقت للنظام ومنطقة العمل الخاصة بها باستخدام العمليات التالية))

read(X) assigns the value of data item X to the local variable xi.

((القراءة (X) تعين قيمة عنصر البيانات X للمتغير المحلي xi))

write(X) assigns the value of local variable xi to data item {X} in the buffer block.

((الكتابة (X) تعين قيمة المتغير المحلي xi لبند البيانات {X} في كتلة المخزن المؤقت))

both these commands may necessitate the issue of an input(BX) instruction before the assignment if the block BX in which X resides is not already in memory.

((قد يستلزم هذان الأمران إصدار تعليمات إدخال (BX) قبل التعيين، إذا لم تكن الكتلة BX التي يوجد بها X موجودة بالفعل في الذاكرة))

Transactions:

Perform read(X) while accessing X for the first time.

((قم بإجراء القراءة (X) أثناء الوصول إلى X لأول مرة))

All subsequent accesses are to the local copy.

((جميع عمليات الوصول اللاحقة إلى النسخة المحلية))

After last access, transaction executes write(X).

((بعد آخر وصول، تنفذ المعاملة كتابة (X)))

output(BX) need not immediately follow write(X).

((الإخراج (BX) لا يحتاج مباشرة إلى متابعة الكتابة (X)))

System can perform the output operation when it deems fit.

((يمكن للنظام إجراء عملية الإخراج عندما يراه مناسبًا))

Recovery and Atomicity:

Modifying the database without ensuring that the transaction will commit may leave the database in an inconsistent state.

((قد يؤدي تعديل قاعدة البيانات دون التأكد من تنفيذ المعاملة إلى ترك قاعدة البيانات في حالة غير متسقة))

Consider transaction Ti that transfers $50 from account A to account B; goal is either to perform all database modifications made by Ti or none.

((ضع في اعتبارك المعاملة Ti التي تنقل 50 دولارًا من الحساب أ إلى الحساب ب؛ الهدف هو إما إجراء جميع تعديلات قاعدة البيانات التي أجراها Ti أو لا شيء على الإطلاق))

Several output operations may be required for Ti (to output A and B).

((قد تكون هناك حاجة للعديد من عمليات الإخراج لـ Ti (لإخراج A وB)))

A failure may occur after one of these modifications has been made but before all of them are made.

((قد يحدث فشل بعد إجراء أحد هذه التعديلات، ولكن قبل إجرائها جميعًا))

To ensure atomicity despite failures, we first output information describing the modifications to stable storage without modifying the database itself.

((لضمان الذرية على الرغم من حالات الفشل، نقوم أولاً بإخراج المعلومات التي تصف التعديلات على التخزين المستقر دون تعديل قاعدة البيانات نفسها))

We study two approaches:

1. log-based recovery.
2. shadow-paging.

We assume (initially) that transactions run serially, that is, one after the other.

((نفترض (في البداية) أن المعاملات تتم بشكل متسلسل، أي واحدة تلو الأخرى))

Log-Based Recovery:

A log is kept in stable storage.

((يتم الاحتفاظ بسجل على تخزين مستقر))

The log is a sequence of log records and maintains a record of update activities on the database.

((السجل عبارة عن سلسلة من سجلات السجل، ويحتفظ بسجل لأنشطة التحديث على قاعدة البيانات))

When transaction Ti starts, it registers itself by writing a <Ti start>log record.

((عندما تبدأ المعاملة Ti، فإنها تسجل نفسها عن طريق كتابة سجل <Ti start>))

Before Ti executes write(X), a log record <Ti, X, V1, V2> is written, where V1 is the value of X before the write, and V2 is the value to be written to X.

((قبل تنفيذ Ti للكتابة (X) ، تتم كتابة سجل السجل < Ti، X، V1، V2> ، حيث V1 هي قيمة X قبل الكتابة ، و V2 هي القيمة التي يجب كتابتها إلى X))

Log record notes that Ti has performed a write on data item Xj Xj had value V1 before the write and will have value V2 after the write.

((يلاحظ سجل السجل أن Ti قد قام بالكتابة على عنصر البيانات Xj Xj كان له القيمة V1 قبل الكتابة، وسيكون له القيمة V2 بعد الكتابة))

When Ti finishes it last statement, the log record <Ti commit> is written.

((عندما تنتهي Ti من العبارة الأخيرة، يتم كتابة سجل السجل <Ti commit>.

We assume for now that log records are written directly to stable storage (that is, they are not buffered)

((نفترض في الوقت الحالي أن سجلات السجل مكتوبة مباشرة في وحدة تخزين مستقرة (أي أنها ليست مخزنة مؤقتًا)))

Two approaches using logs.

Deferred database modification.

Immediate database modification.

Deferred Database Modification:

The deferred database modification scheme records all modifications to the log but defers all the writes to after partial commit.

((يسجل مخطط تعديل قاعدة البيانات المؤجل جميع التعديلات على السجل، لكنه يؤجل جميع عمليات الكتابة بعد الالتزام الجزئي))

Assume that transactions are carried out serially.

((افترض أن المعاملات يتم تنفيذها بشكل متسلسل))

Transaction starts by writing <Ti start> record to log.

((تبدأ المعاملة بكتابة سجل <Ti start> للتسجيل))

A write(X) operation results in a log record <Ti, X, V> being written, where V is the new value for X.

((تؤدي عملية الكتابة (X) إلى كتابة سجل السجل <Ti، X، V>، حيث V هي القيمة الجديدة لـ X))

The writing is not performed on X currently but is deferred.

((لا يتم تنفيذ الكتابة على X في هذا الوقت، ولكن تم تأجيلها))

When Ti partially commits, <Ti commit> is written on the log.

((عندما يرتكب Ti جزئيًا، تتم كتابة <Ti commit> في السجل))

Finally, the log records are read and used to execute the previously deferred writes.

((أخيرًا، تتم قراءة سجلات السجل واستخدامها لتنفيذ عمليات الكتابة المؤجلة مسبقًا))

During recovery after a crash, a transaction needs to be redone if and only if both <Ti start> and<Ti commit> are there in the log.

((أثناء الاسترداد بعد الانهيار، يجب إعادة إجراء المعاملة إذا وفقط في حالة وجود كل من <Ti start> و <Ti commit> في السجل))

Redoing a transaction Ti (redo Ti) sets the value of all data items updated by the transaction to the new values.

((تؤدي إعادة المعاملة Ti (redo) إلى تعيين قيمة جميع عناصر البيانات التي تم تحديثها بواسطة المعاملة إلى القيم الجديدة))

Crashes can occur while the transaction is executing the original updates, or while recovery action is being taken.

((يمكن أن تحدث الأعطال أثناء تنفيذ المعاملة للتحديثات الأصلية، أو أثناء اتخاذ إجراء الاسترداد))

Immediate Database Modification:

scheme allows database updates of an uncommitted transaction to be made as the writes are issued.

((يسمح النظام بإجراء تحديثات لقاعدة البيانات الخاصة بمعاملة غير ملتزمة عند إصدار الكتابات))

Since undoing may be needed, update logs must have both old value and new value.

((نظرًا لأنه قد تكون هناك حاجة إلى التراجع، يجب أن تحتوي سجلات التحديث على قيمة قديمة وقيمة جديدة))

Update log record must be written before database item is written.

((يجب كتابة سجل التحديث قبل كتابة عنصر قاعدة البيانات))

We assume that the log record is output directly to stable storage.

((نفترض أن سجل السجل يتم إخراجه مباشرة إلى تخزين مستقر))

Can be extended to postpone log record output, so long as prior to execution of an output(B) operation for a data block B, all log records corresponding to items B must be flushed to stable storage.

((يمكن تمديده لتأجيل إخراج سجل السجل، طالما أنه قبل تنفيذ عملية الإخراج (B) لكتلة البيانات B ، يجب مسح جميع سجلات السجل المطابقة للعناصر B إلى تخزين مستقر))

Output of updated blocks can take place at any time before or after transaction commit.

((يمكن أن يحدث إخراج الكتل المحدثة في أي وقت قبل أو بعد تنفيذ المعاملة))

The order in which blocks are output can be different from the order in which they are written.

((يمكن أن يختلف ترتيب الكتل التي يتم إخراجها عن الترتيب الذي تمت كتابتها به))

Recovery procedure has two operations instead of one:

undo(Ti) restores the value of all data items updated by Ti to their old values, going backwards from the last log record for Ti.

((قيمة جميع عناصر البيانات التي تم تحديثها بواسطة Ti إلى قيمها القديمة، بالرجوع للخلف من آخر سجل لـ Ti))

redo(Ti) sets the value of all data items updated by Ti to the new values, going forward from the first log record for Ti.

((تعيد الإعادة (Ti) قيمة جميع عناصر البيانات التي تم تحديثها بواسطة Ti إلى القيم الجديدة، بدءًا من سجل السجل الأول لـ Ti))

Both operations must be idempotent:

That is, even if the operation is executed multiple times the effect is the same as if it is executed once.

((أي، حتى لو تم تنفيذ العملية عدة مرات، فإن التأثير هو نفسه كما لو تم تنفيذها مرة واحدة))

Needed since operations may get re-executed during recovery.

((هناك حاجة لأنه قد يتم إعادة تنفيذ العمليات أثناء الاسترداد))

When recovering after failure:

Transaction Ti needs to be undone if the log contains the record <Ti start> but does not contain the record <Ti commit>.

((يجب التراجع عن المعاملة Ti إذا كان السجل يحتوي على السجل <Ti start> ، ولكنه لا يحتوي على السجل <Ti commit>.

Transaction Ti needs to be redone if the log contains both the record <Ti start> and the record <Ti commit>.

((تحتاج المعاملة Ti إلى إعادة بنائها إذا كان السجل يحتوي على كل من السجل <Ti start> والسجل < Ti commit>.

Undo operations are performed first, then redo operations.

Checkpoints:

Problems in recovery procedure as discussed earlier :

((مشاكل في إجراءات الاسترداد كما تمت مناقشته سابقًا))

Searching the entire log is time-consuming. We might unnecessarily redo transactions which have already output their updates to the database.

((يستغرق البحث في السجل بالكامل وقتًا طويلاً، وقد نقوم بإعادة المعاملات غير الضرورية التي أدت بالفعل إلى إخراج تحديثاتها إلى قاعدة البيانات))

Streamline recovery procedure by periodically performing checkpointing.

((تبسيط إجراءات الاسترداد من خلال إجراء checkpoint بشكل دوري))

Output all log records currently residing in main memory onto stable storage.

((قم بإخراج جميع سجلات السجل الموجودة حاليًا في الذاكرة الرئيسية على وحدة تخزين ثابتة))

Output all modified buffer blocks to the disk.

((إخراج جميع الكتل العازلة المعدلة إلى القرص))

Write a log record < checkpoint> onto stable storage.

((اكتب سجل <checkpoint > على تخزين ثابت))

During recovery we need to consider only the most recent transaction Ti that started before the checkpoint, and transactions that started after Ti.

((أثناء الاسترداد، نحتاج إلى النظر فقط في أحدث معاملة Ti التي بدأت قبل نقطة التفتيش ، والمعاملات التي بدأت بعد Ti))

Scan backwards from end of log to find the most recent <checkpoint> record.

((امسح للخلف من نهاية السجل للعثور على أحدث سجل <checkpoint>))

Continue scanning backwards till a record <Ti start> is found.

((استمر في المسح للخلف حتى يتم العثور على سجل <Ti start>))

Need only consider the part of log following above start record.

((تحتاج فقط إلى النظر في جزء السجل الذي يلي سجل البداية أعلاه))

Earlier parts of log can be ignored during recovery and can be erased whenever desired.

((يمكن تجاهل الجزء السابق من السجل أثناء الاسترداد، ويمكن محوه متى رغبت في ذلك))

For all transactions (starting from Ti or later) with no <Ti commit>, execute undo(Ti).

((بالنسبة لجميع المعاملات (بدءًا من Ti أو ما بعده) بدون <Ti commit>، قم بتنفيذ التراجع (Ti)))

Scanning forward in the log, for all transactions starting from Ti or later with a <Ti commit>, execute redo(Ti).

((بالمسح إلى الأمام في السجل، لجميع المعاملات التي تبدأ من Ti أو لاحقًا باستخدام <Ti الالتزام> ، وتنفيذ الإعادة (Ti)))

PL/SQL Block Structure:

DECLARE

```
        (Declarative section)
    BEGIN
        (Executable section)
    EXCEPTION
        (Exception handling section)
    END;
    /
```

((هذا هو الشكل العام في نظامpl/sql تبدا بـdeclare التي يتم فيها تعريف القيم و يجب ان يكون البرنامج بين كلمتين begin و end; ويتم كتابة العمليات بينهما))

dbms_output.put_line (*message*); هذه هي جملة الطباعة في لغة pl/sql

(:= is used to initialize variables)

تستخدم =: لإنشاء متغير جديد لا تنسى علامة :

dbms_output.put_line('Sum = ' || z);

|| هذه لعملية cocatnation او الدمج

```
if condition then
statement.
 End if; '
```

الجمل الشرطية تتكون من if than وبالنهايةend if الكلمتان منفصلتان ويمكن إضافة كلمة else قبل النهاية

في حال وجود أكثر من جملة شرطية يمكنك عمل الهيكلية الاتية

```
IF  condition 1       THEN
 {…statements to be executed…}
ELSIF condition 2  THEN
 {…statements to be executed…}
```

ELSE
{…statements to be executed…}
END IF;

Looping:

1-

```
loop
statement;
counter:=counter +1;
exit when (counter =5);
end loop;
```

2-

```
WHILE (Boolean condition )
LOOP
{.statements.}
END LOOP;
```

3-

```
For counter in 1..5
 loop
statement;
end loop;
```

The

End