Base de Datos

DC. FCEyN
2024-10-9 teórica 7

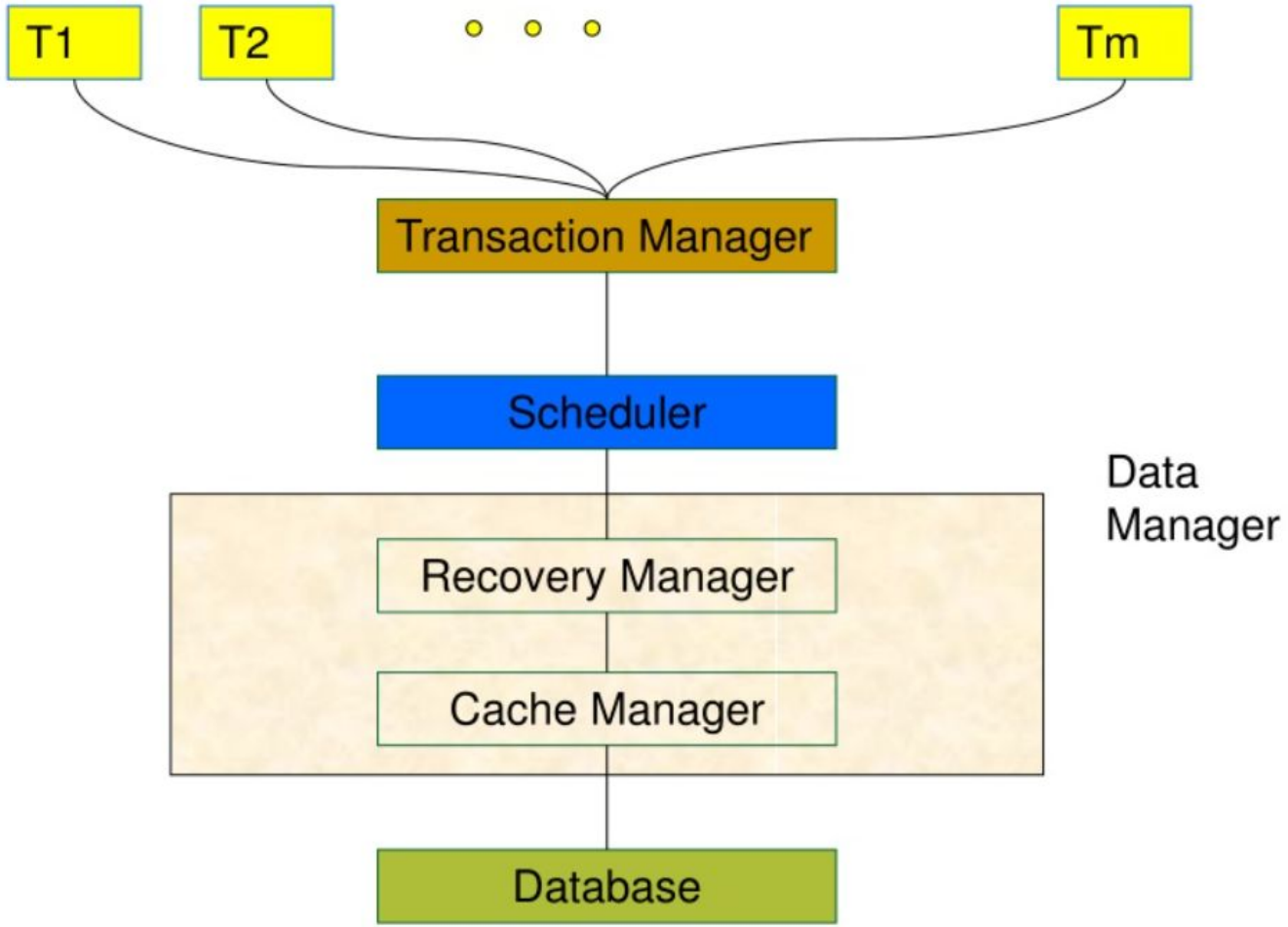
Emilio Platzer
tute@dc.uba.ar

∃ ∄ ⟶ ⟹ ∀

Logging

INTRODUCCIÓN - Objetivo - Tipos de fallas

- El objetivo de la recuperación es asegurar que una base de datos puede procesar transacciones en un modo a prueba de fallas
- Existen básicamente 3 tipos de fallas
    - De transacciones
    - De sistema
    - De almacenamiento
- Estas fallas tienen que ver, básicamente con la pérdida de los datos que están en memora


T1
T2
Tm
Transaction Manager
Scheduler
Recovery Manager
Cache Manager
Data Manager
Database

CACHE MANAGER

• Es el responsable de interactuar con el almacenamiento no volátil. Sus principales funciones son

- Mantener una cache (datos, ubicación, dirty bit)
- Fetch (leer de disco o de la cache)
- Flush (escribir en la disco o liberar la cache)

RECOVERY MANAGER - API

- La interface del RM interfaz está definida por medio de 5 procedimientos:
    - RM-Read(Ti, x): lee el valor de x para la transacción Ti;
    - RM-Write(Ti, x, v): escribe v en x en nombre de la transacción Ti
    - RM-Commit(Ti): commit Ti;
    - RM-Abort(Ti): abort Ti;
    - Restart: lleva a la base de datos al último estado “comiteado” antes de que haya fallado el sistema.

RECOVERY MANAGER - CARACTERÍSTICAS

- Recovery guarda datos a través del Cache Manager
- Recovery depende de
    - Redo: si solo escribe transacciones que haya comiteado
    - Undo: si permite que transacciones no comiteadas escriban en disco:
        - Única versión
        - Multiverso

RECOVERY MANAGER - El log

- Según su tipo el RM puede escribir datos de transacciones activas
- Lo que siempre hace es escribir un log
  - cada vez que hace un RM-Write <Ti, x, v_{nuevo}, v_{viejo}>
  - al iniciar o terminar la transacción: <Ti, start>, <Ti, commit>, <Ti, abort>
  - a veces también se hacen checkpoints: <checkpoint start>, <checkpoint end>
- El log se escribe antes de guardar los datos a disco
  - salvo el <Ti, commit> que se guarda después en los REDO.log

RECOVERY MANAGER - Procesos de recovery

- Tienen una fase de análisis:
  - Detectar la última transacción que efectivamente está reflejada en el almacenamiento principal (tablas)
  - Determinar qué transacciones están incompletas (se descartan las incompletas o que terminan en commit)
- Procesan los logs
  - A partir del más antiguo: agregan los datos en las tablas correspondientes
  - A partir del más nuevo: recuperan los datos sin re-recuperar (sin pisar lo ya recuperado)
  - Multiverso: eliminan (o marcan) las versiones con datos viejos

CHECKPOINT

- Un log muy largo (desde que se inició la base) puede ser muy costoso a la hora de recuperar una base de datos
- Un checkpoint es un proceso para usar log más corto al recuperar
  - El checkpoint debe marcar qué registros del log ya no son necesarios
    - marcando un punto en el log o registro por registro
  - El checkpoint puede aprovechar para asegurarse que todos los datos fueron guardados en disco y fueron hechas las limpiezas necesarias

TIPOS DE CHECKPOINT

- Quiescente, deja de aceptar nuevas transacciones mientras procesa
- Non quiescente, sigue aceptando transacciones mientras procesa, el flush real se produce cuando todas las transacciones que estaban activas al comienzo finalizan con COMMIT o ABORT
    - Marcan en el log al iniciar el proceso de checkpoint y al finalizarlo (dos momentos separados)

De qué hablamos cuando hablamos de Backup, le dijo un soldado romano al Cesar

48 a.C.

BACKUPS

- Objetivos:
  - Recuperarse de una pérdida de información
  - Migrar la base de datos (a otro hardware o a otra versión de la base)
  - Trabajar en una copia (para desarrollo, análisis *what-if*, auditorías)
- Tipos:
  - Muleto (réplica al día de la base de datos funcional)
  - Hot backup (sin detener o molestar el uso)

  } ¡gracias a los logs!
  - Backups binarios (eficientes y compactos)
  - Dump de texto (flexibles, son las instrucciones SQL necesarias)

RÉPLICA DE LA BASE DE DATOS

- Habitualmente alojado en otro servidor
- Recibe un flujo (stream) con los logs que produce la base de datos maestra
- Utilizando el mismo algoritmo de recuperación actualiza la réplica con un delay mínimo
- Es read-only hasta cortar el flujo del log y pasarla a master

Ventajas:

- Super rápida y super eficiente

Desventaja: es la opción más costosa

HOT BACKUPS

Ventajas:

- No hay que detener ni enlentencer la base de datos.
- Se utiliza un comando común a nivel del sistema operativo (ej TAR GZ) primero se copian los archivos de datos y después los logs. No importa si durante el TAR GZ la base de datos siguió escribiendo. Los logs son estrictamente secuenciales

Desventajas:

- Atado a la versión específica del motor de base de datos
- Podría condicionar la marca o versión del sistema operativo

BACKUP BINARIO

- Formato propietario de la base de datos
- Permite copiar entre distintos sistemas operativos y entre versiones no lejanas de la base de datos (en general la compatibilidad se conserva cuando no se cambia el *major version*)

Ventajas

- Es más eficiente (rápido y compacto) tanto para hacerlo como para levantarlo

Desventajas

- Puede haber un costo interrumpiendo o enlenteciendo el uso normal.

DUMP DE TEXTO

- Son todos los comandos SQL que permiten volver a crear la base de datos con todos sus datos, relaciones, restricciones, triggers, procedimientos almacenados y otras configuraciones (ej: dominios)

Ventajas

- Sirven para mover los datos de una base de datos a otra de otra marca (a veces hay que procesar el archivo de texto cuando se usan tipos de datos u opciones de configuración no estandarizada).
- Es el mecanismo usual para hacer upgrade del motor (*major version*)

Desventajas: Más lento, ocupa más espacio (aún cuando se compacte)


ELGATOYLACAJA.COM