



Commitment control

for everyone

Marco Riva
www.markonetools.it

IBM CHAMPION 2021-2025 Power user

Ultimo aggiornamento: 14/01/2026

RPG SQL

IBM DB2

1



Critical

Important

Nice to have

Gruppi attivazione
Ambito
Attivazione
Operazioni implicate

Livello isolamento
Concorrenza accesso
Monitoraggio

Deadlock
Savepoint
Soft commit
Giornali

Commitment control for everyone

2



3



4

MK1 common
ITALY

5

Controllo di sincronia

Concetti base

Commitment control for everyone

This slide is part of a presentation titled 'MK1 common ITALY'. It features a green arrow-shaped logo with the text 'MK1 common' and 'ITALY' inside. The slide number '5' is displayed at the top right. The main title 'Controllo di sincronia' is centered in large green font, with the subtitle 'Concetti base' below it. A small note 'Commitment control for everyone' is at the bottom. The background is light beige.

MK1 common
ITALY

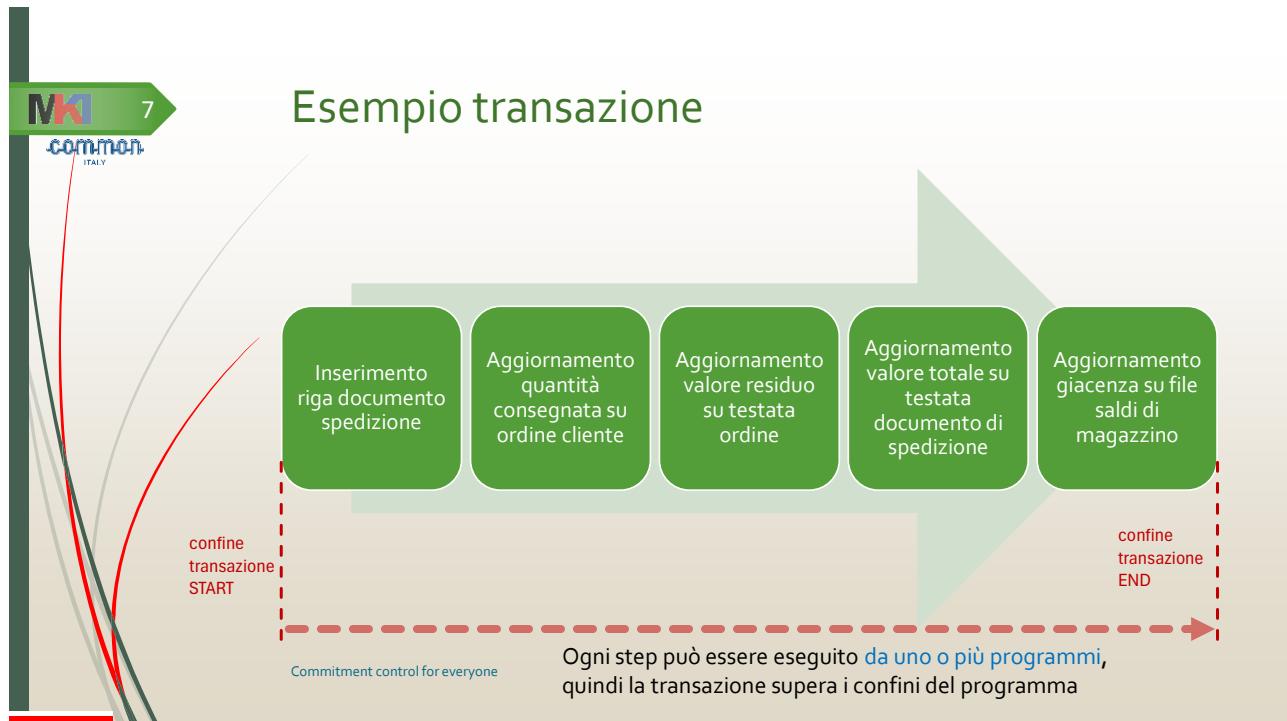
6

Controllo sincronia (commitment control)

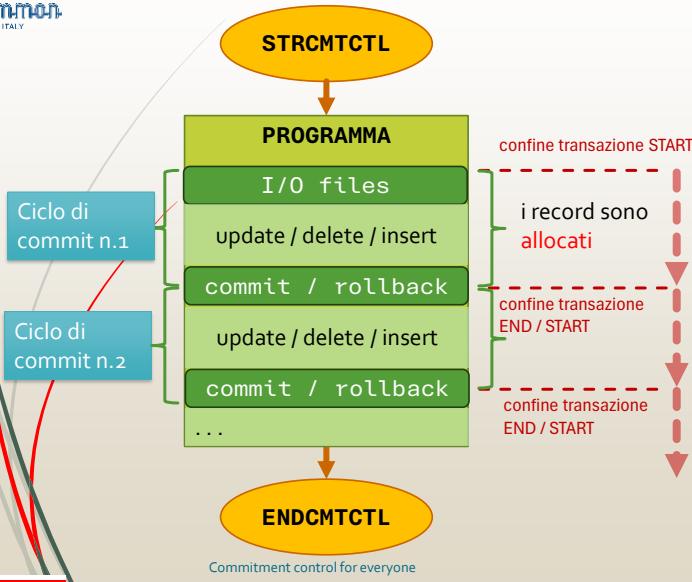
- Consente di definire e consolidare un gruppo di modifiche sul database come **unità logica di lavoro** (LUW) = **transazione**
- Una **transazione** è un gruppo di modifiche a uno o più file di database che dal punto di vista dell'utente appaiono come una singola modifica
- Assicura che l'intero gruppo di modifiche apportate in una transazione o vengano tutte consolidate sul db (*commit*) o tutte annullate (*rollback*) → **integrità del dato**

Commitment control for everyone

This slide is part of a presentation titled 'MK1 common ITALY'. It features a green arrow-shaped logo with the text 'MK1 common' and 'ITALY' inside. The slide number '6' is displayed at the top right. The main title 'Controllo sincronia (commitment control)' is centered in large green font. Below it is a bulleted list of three points explaining what a transaction is and how it works. A small note 'Commitment control for everyone' is at the bottom. The background is light beige.



Controllo di sincronia – schema logico



- La prima operazione di I/O e i successivi commit/rollback definiscono i **confini della transazione**
- per le applicazioni SQL il controllo di sincronia viene attivato automaticamente nell'ambito dell'activation group
- Un **ciclo di commit** è il tempo che intercorre tra due confini di transazione.
Il sistema assegna un identificatore ad ogni ciclo di commit

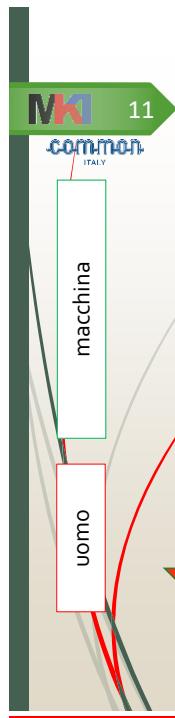
Controllo sincronia: prerequisiti

- I files devono essere registrati sullo **stesso giornale** con entrambe le immagini before e after.
Se il file è registrato solo con immagine after, il sistema registrerà anche l'immagine before
- I/O nativo:** avviare definizione del controllo sincronia (**STRCMTCCTL**) e definire i file con keyword **COMMIT**
- I/O SQL:** impostare il livello di isolamento (**set option commit**) diverso da *NONE

create schema o create collection
creano automaticamente una nuova libreria con il giornale di default QSQJRN e qualsiasi tabella creata nella libreria viene registrata sul giornale

Commitment control for everyone

Se si crea la libreria con **CRTLlib** occorre creare manualmente ricevitore e giornale ed attivare la registrazione con STRJRNLIB

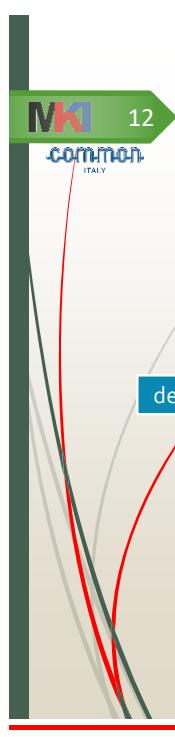


Perché lo si teme?

- ▶ richiede che tutti i file siano registrati sullo stesso giornale. Quindi si evita per timore di decadimento di performance e di occupazione spazio disco
- ▶ prima di RPG IV era macchinoso l'uso opzionale del controllo di sincronia nei programmi
- ▶ prima di ILE il controllo di sincronia agiva solo a livello di job e non dell'activation group
- ▶ poiché i file sono registrati su giornale richiede maggior attenzione nella manipolazione di oggetti e librerie
- ▶ **complessità del disegno applicativo per definire correttamente le transazioni e gli activation group**

Commitment control for everyone

11



Ambito delle transazioni

- ▶ L'ambito (=scope) di una transazione può essere a livello
 - ▶ **Lavoro:** ogni programma chiamato dopo STRCMTCTL CMTSCOPE (*JOB) in esecuzione in qualsiasi gruppo di attivazione che non abbia un controllo di sincronia specifico del proprio gruppo di attivazione userà il controllo di sincronia a livello di lavoro
 - ▶ **Gruppo di attivazione:** STRCMTCTL CMTSCOPE (*ACTGRP) oppure applicazione SQL con SET OPTION COMMIT diverso da *NONE. Solo i programmi in esecuzione nel gruppo di attivazione useranno il controllo di sincronia specifico di quel gruppo
 - ▶ **Transazione:** avviati con XA APIs for Transaction Scoped Locks. Questa API è utilizzata per associare la definizione del controllo sincronia a uno specific thread o a una connessione SQL e non all'activation group

Commitment control for everyone

12

Job e activation group

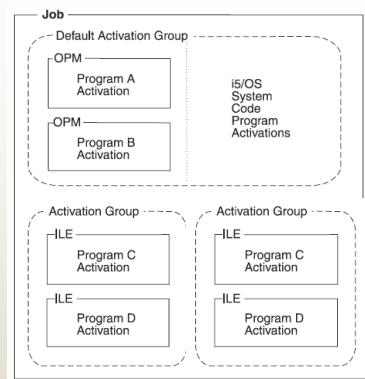
Un breve ripasso

Commitment control for everyone

13

Job e activation group: program isolation

- ▶ L'**activation group** contiene le **risorse** necessarie ad eseguire il programma. Le risorse sono:
 - ▶ Static program variables
 - ▶ Dynamic storage
 - ▶ Temporary data management resources
 - ▶ ODP (Open Data Path)
 - ▶ **Commitment definitions**
 - ▶ SQL cursors
 - ▶ HFS (Hierarchical File System)
- ▶ All'avvio di un job, *automaticamente* vengono creati 2 activation group di **default** usati da tutti i programmi OPM e ILE OPM compatible



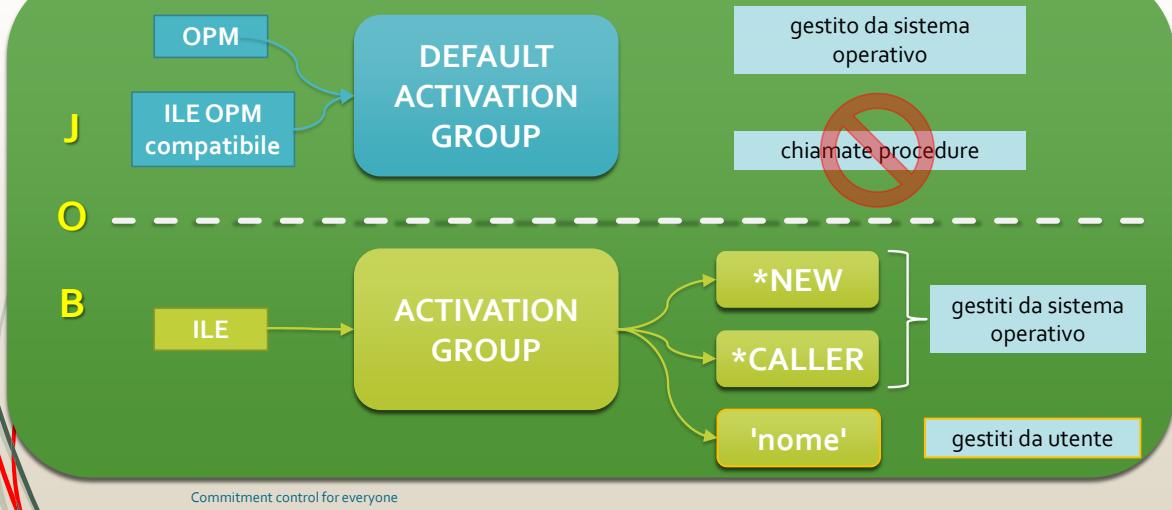
Fonte IBM

Commitment control for everyone

14

MK1
common
ITALY

Activation group in sintesi



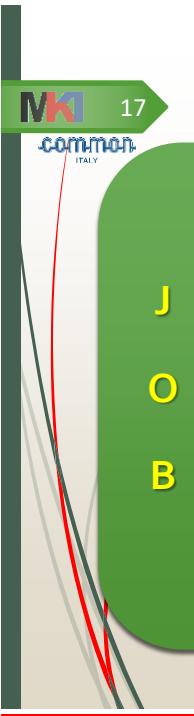
15

MK1
common
ITALY

Activation group: comandi principali

- **RCLACTGRP**: libera le risorse di un AG nominale
- **CRTSQLRPGI** parametro **CLOSQLCSR**: determina quando viene chiuso automaticamente un cursore SQL. *ENDACTGRP o *ENMOD.
- **OVR...** parametro **OVRSCOPE**: determina l'ambito di validità della sostituzione. *ACTGRPDFN o *CALLLVL o *JOB.
- **DLTOVR** parametro **LVL**: determina quale livello di sostituzioni eliminare. *ACTGRPDFN o * o *JOB.

16



Comm.lt. quiz

DEFAULT ACTIVATION GROUP

STRCMTCTL *JOB

PGMC → PGMD

ACTIVATION GROUP

XYZ

PGMA

1

PGMA condivide la stessa definizione di controllo di sincronia del programma PGMC e PGMD?



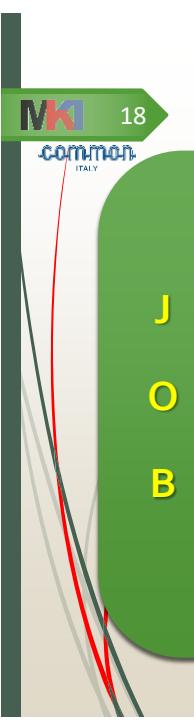
YES



NO



17



Comm.lt. quiz

DEFAULT ACTIVATION GROUP

PGME

*embedded sql
option commit = *chg*

ACTIVATION GROUP 23

PGMF

*embedded sql
option commit = *chg*

2

PGMF condivide la stessa definizione di controllo di sincronia del programma PGME?



YES



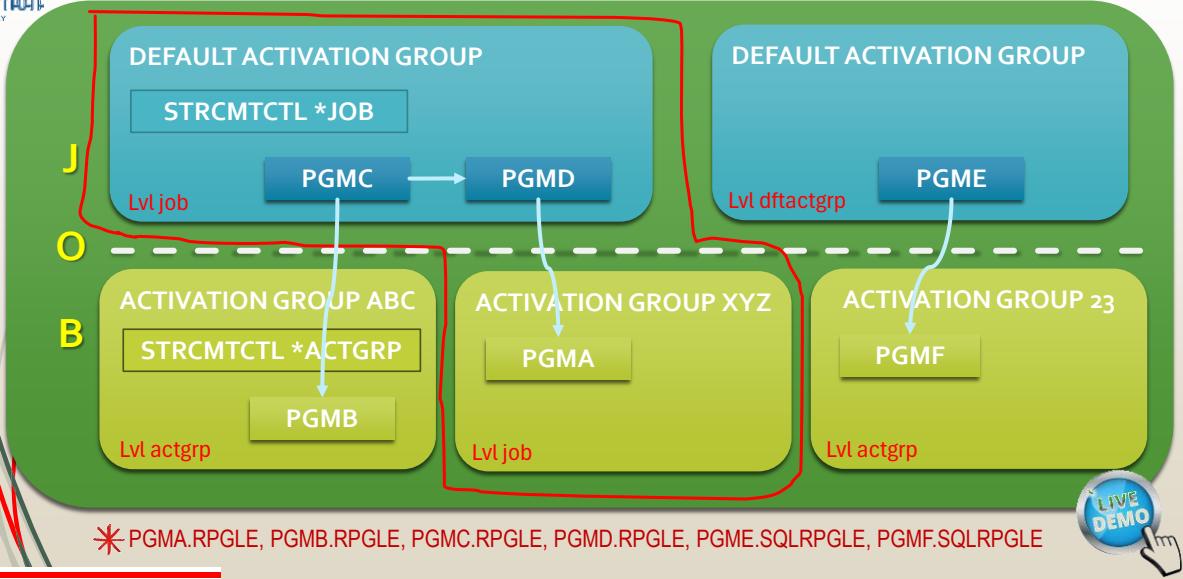
NO



18

MK1
common
ITALY

Controllo sincronia - isolamento transazioni/1



19

MK1
common
ITALY

Controllo sincronia - isolamento transazioni/2

Annotazioni allo schema precedente

- PGMC avvia STRCMTCTL a livello *JOB
- i programmi PGMD e PGMA chiamati da PGMC ereditano la definizione di commit a livello job
- il programma PGMB avvia STRCMTCTL a livello *ACTGRP quindi viene creata una definizione di commit a livello dell'activation group ABC
- il PGME (embedded SQL) nel dftactgrp non avvia esplicitamente STRCMTCTL quindi si crea una definizione implicita di commit a livello *DFTACTGRP
- il PGMF (embedded SQL) chiamato da PGME apre un nuovo activation group e implicitamente crea una nuova definizione di commit a livello dell'activation group

Commitment control for everyone

20

Controllo sincronia - isolamento transazioni/3

Solo programmi in ambiente ILE possono aprire transazioni in un gruppo di attivazione diverso da quello di default

in un lavoro possono esistere più transazioni contemporanee e indipendenti se e solo se nel job sono in esecuzione uno o più programmi ILE

Ogni transazione può allocare al max 500.000.000 di record, ma si consiglia (*IBM documentation*) di non superare i 2.000 record per transazione

Commitment control for everyone

21

Controllo sincronia - isolamento transazioni/4

In un ambiente misto OPM e ILE se si desidera che tutti i programmi condividano la medesima definizione di controllo di sincronia è essenziale che l'ambito sia *JOB

Commitment control for everyone

22

Livelli di isolamento delle transazioni

The diagram illustrates the hierarchy of transaction isolation levels. On the left, a vertical bar shows a gradient from red at the bottom to green at the top, labeled "integrità e sicurezza del dato". An arrow points upwards through the colors. To the right of the bar is a table with five rows, each representing a different isolation level. The levels are ordered from highest isolation (top) to lowest (bottom): *RR/RR, *ALL/RS, *CS/CS, *CHG/UR, and *NONE/NC.

Livello	Descrizione	Attivo	Lettura sporche	Record fantasma	Allocazione record letti	Note
*RR/RR	LETTURA RIPETIBILE (Repeatable Read or Serializable)	✓	✗	✗	✓	Garantisce che i record letti non possono essere modificati da un altro gruppo di attivazione e che ogni record modificato in un altro gruppo di attivazione non può essere letto fino a quando viene consolidato Allocata tutti i record letti e il gruppo di attivazione è completamente isolato rispetto ad altri
*ALL/RS	LETTURA STABILE (Read Stability)	✓	✗	✓	✓	Garantisce che i record letti non possono essere modificati da un altro gruppo di attivazione e che ogni record modificato in un altro gruppo di attivazione non può essere letto fino a quando viene consolidato → come RR Record "fantasma": a differenza di RR eseguendo più volte la stessa query possono comparire i record aggiunti in altri gruppi di attivazione
*CS/CS	LETTURA SINCRONIZZATA (Cursor Stability or Read Committed)	✓	✗	✓	✗	Garantisce che ogni record modificato in un altro gruppo di attivazione non può essere letto fino a quando viene consolidato → come RR o RS A differenza di RR e RS i record letti possono essere modificati da altri gruppi di attivazione
*CHG/UR	LETTURA NON SINCRONIZZATA (Uncommitted Read)	✓	✓	✓	✗	Livello più basso di controllo di sincronia. Consente di leggere record modificati in altri gruppi di attivazione anche se non ancora consolidate
*NONE/NC	NESSUNO	✗	✓	✓	✗	Ogni singolo aggiornamento è effettivamente sincronizzato quando viene completato. Non è possibile eseguire commit/rollback

Commitment control for everyone <https://www.ibm.com/docs/en/i/7.6.0?topic=concepts-isolation-level>

23

Livelli di isolamento delle transazioni/2

- ▶ **Lettura sporche:** è possibile leggere dati che sono stati variati ma non sincronizzati da un altro lavoro.
- ▶ **Record fantasma:** la LUW 1 legge un set di record che soddisfa certi criteri. La LUW 2 inserisce un nuovo record che soddisfa i criteri di selezione della query della LUW 1. La LUW 1 riesegue la query e legge anche i nuovi record
- ▶ **Lettura non ripetibile:** la LUW 1 legge un record. La LUW 2 modifica quel record e lo consolida. La LUW1 rilegge il medesimo record ed ottiene i nuovi dati consolidati dalla LUW 2

<https://www.ibm.com/docs/en/i/7.6.0?topic=level-comparison-isolation-levels>

Commitment control for everyone

24

MK1
common
ITALY

25

Livello di allocazione

- Il parametro **LCKLVL** in STRCMTCTL determina il livello di allocazione dei record di default

Tipo	Descrizione	Conseguenze
*ALL	allocazione dei record modificati e letti durante tutta la transazione	- anche un record letto senza scopo di aggiornamento è allocato
*CS	allocazione dei record modificati durante tutta la transazione allocazione dei record letti fino al rilascio o alla successiva lettura	- un altro lavoro non può leggere record per aggiornamento che sono già stati letti dal lavoro corrente - il lavoro corrente non può leggere record per aggiornamento che sono stati allocati di tipo *update in un altro lavoro
*CHG	allocazione dei record modificati durante tutta la transazione	

Commitment control for everyone

<https://www.ibm.com/docs/en/i/7.6.0?topic=control-commit-lock-level>

25

 MK1
common
ITALY

26

Prerequisiti per attivare controllo di sincronia

Specifiche di controllo e definizione file

Opzioni precompilatore SQL

Commitment control for everyone

26



27

H specifiche di controllo

- ▶ Keyword per activation group
 - ▶ DFTACTGRP (*YES | *NO)
 - ▶ ACTGRP (*STGMDL | *NEW | *CALLER | '*nome*')

F specifiche definizione file

- ▶ keyword COMMIT
- ▶ keyword USROPN COMMIT (*ind-variable*)
se la variabile *ind-variable* è impostata a '1' il controllo di sincronia è attivato, se '0' invece non viene attivato

Commitment control for everyone

27



28

STRCMTCTL / ENDCMTCTL

- ▶ **STRCMTCTL**
avvia una definizione di controllo di sincronia
 - ▶ Se già attivo → errore CPF8351
- ▶ **ENDCMTCTL**
arresta una definizione di controllo di sincronia
 - ▶ Interattivo: se eseguito con modifiche in sospeso o con risorse aperte → CPA8350 o CPF8355
 - ▶ CM: esegue commit e prosegue
 - ▶ RB: esegue rollback e prosegue
 - ▶ Batch: se eseguito con modifiche in sospeso o con risorse aperte → rollback

Commitment control for everyone

28

Embedded SQL

- ▶ Parametro **COMMIT** nei comandi di compilazione
- ▶ Oppure nel sorgente si può specificare la direttiva di compilazione con l'istruzione SQL set option. P.es.:
`exec sql set option commit = *CHG;`
- ▶ Oppure il livello di isolamento può essere specificato in ogni singola istruzione SQL DELETE, INSERT, SELECT, UPDATE (*isolation-clause*)
`... with ur;`
- ▶ **Implicitamente** viene avviato il controllo di sincronia a livello di **activation group**

Commitment control for everyone

<https://www.ibm.com/docs/en/i/7.6.0?topic=statement-isolation-clause>

29

Chiudere una transazione/1

- ▶ Una transazione viene chiusa esplicitamente con le operazioni di commit (vengono consolidate le modifiche) o rollback (vengono annullate le modifiche)
- ▶ I codici operativi RPG per commit e rollback sono:
COMMIT, **ROLBK**
- ▶ Le istruzioni SQL per commit e rollback sono
`exec sql commit;`
`exec sql rollback;`

Commitment control for everyone

30

Chiudere una transazione/2

► COMMIT

- Consolida tutte le modifiche eseguite sui record dal precedente commit/rollback
- Rilascia tutti i lock sui record
- Non viene alterata la posizione dei file

► ROLLBACK

- Annulla tutte le modifiche eseguite sui record a partire dal precedente commit/rollback o dal savepoint (se embedded SQL)
- Rilascia tutti i lock sui record
- Riposiziona i file alla posizione *al momento del precedente commit* (cfr. esempio slide 60)

Commitment control for everyone

31

To be continued...

Commitment control for everyone



32