



Commitment control

for everyone

Marco Riva
www.markonetools.it

IBMCHAMPION 2021-2025 Power area

Ultimo aggiornamento: 18/02/2026

IBM DB2 RPG SQL

1



KEY POINTS

- 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

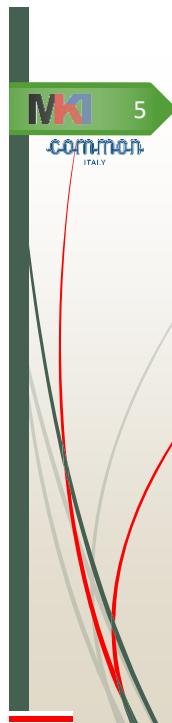
2



3



4



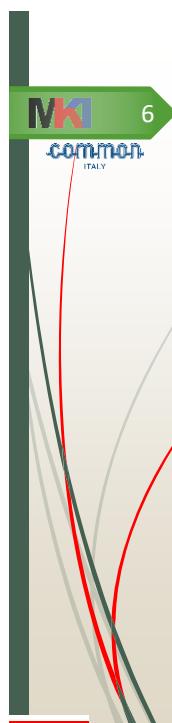
5

Controllo di sincronia

Concetti base

Commitment control for everyone

5



6

Controllo sincronia (commitment control)



- consente di **definire e consolidare** un gruppo di modifiche sul database come **unità logica di lavoro** (LUW) = **transazione**



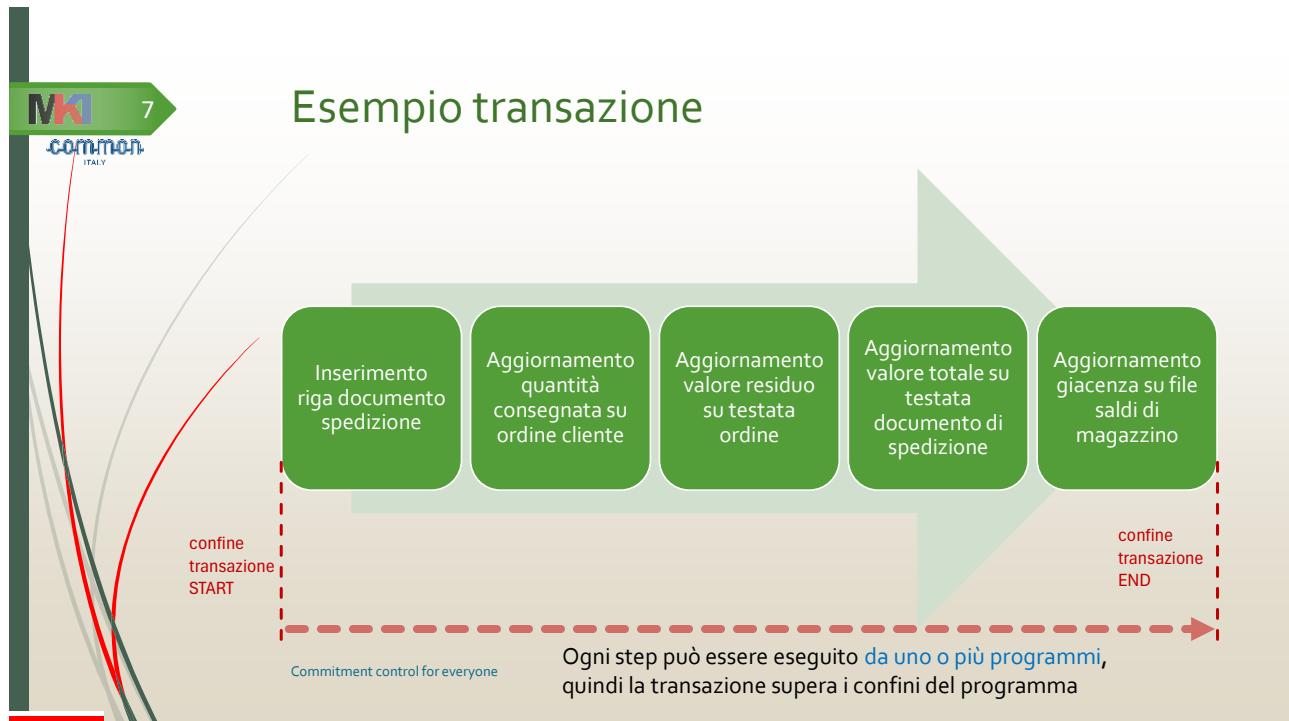
- **transazione**
- è un gruppo di modifiche a uno o più file di database che dal punto di vista dell'utente appaiono come una singola modifica



- **integrità e coerenza del dato**
- Assicura che l'intero gruppo di modifiche apportate in una transazione o vengano tutte consolidate sul db (*commit*) o tutte annullate (*rollback*)

Commitment control for everyone

6



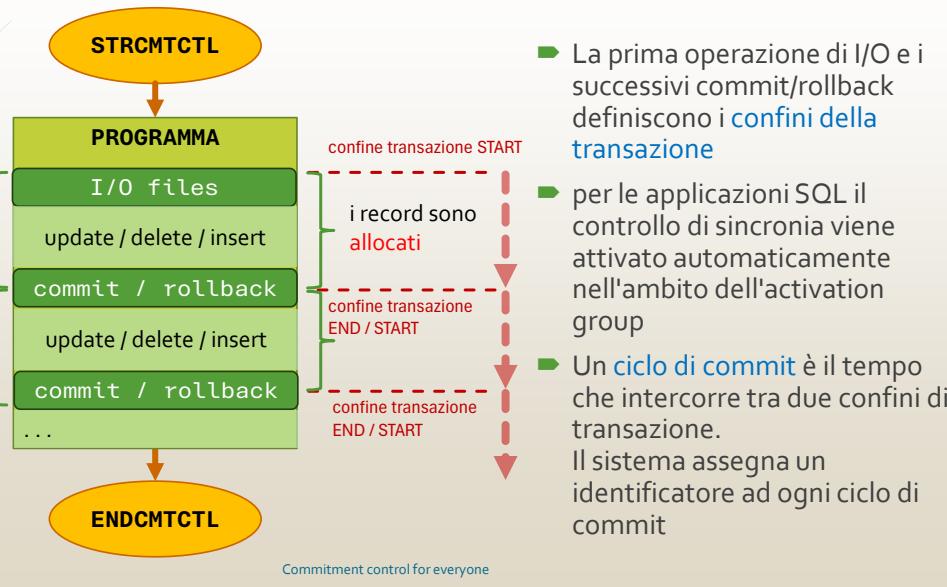
7



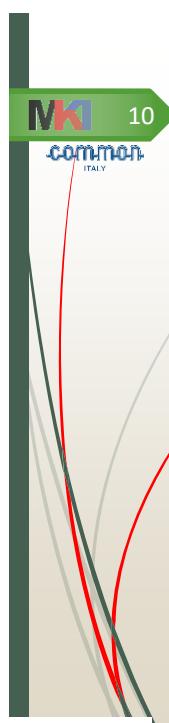
8



Controllo di sincronia – schema logico



9



Controllo sincronia: prerequisiti

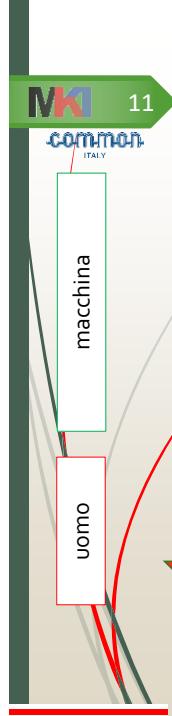
- ▶ I files devono essere registrati sullo **stesso giornale** con entrambe le immagini before e after.
Se il file è registrato solo con immagine after, il sistema registrerà anche l'immagine before
- ▶ **I/O nativo**: avviare definizione del controllo sincronia (**STRCMCTL**) e definire i file con keyword **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**

10



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) delle transazioni 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

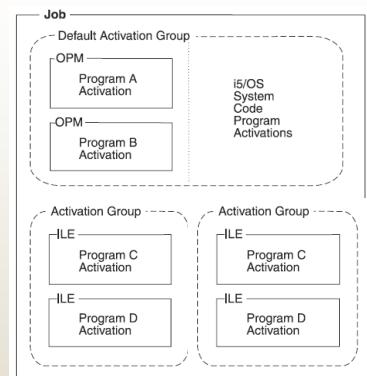
Un breve ripasso

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

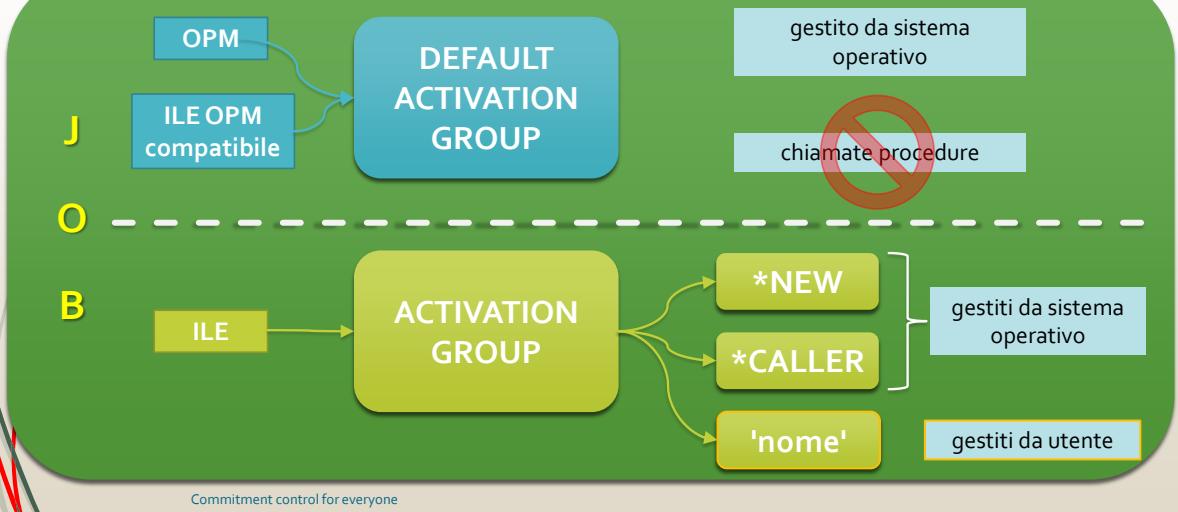
Commitment control for everyone

14

MK1
common
ITALY

15

Activation group in sintesi



15

MK1
common
ITALY

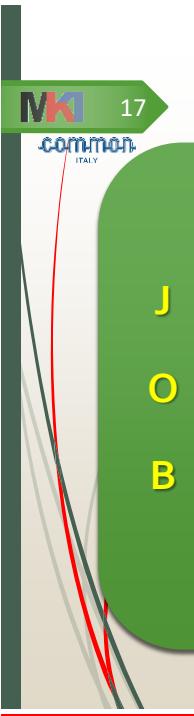
16

Activation group: comandi principali

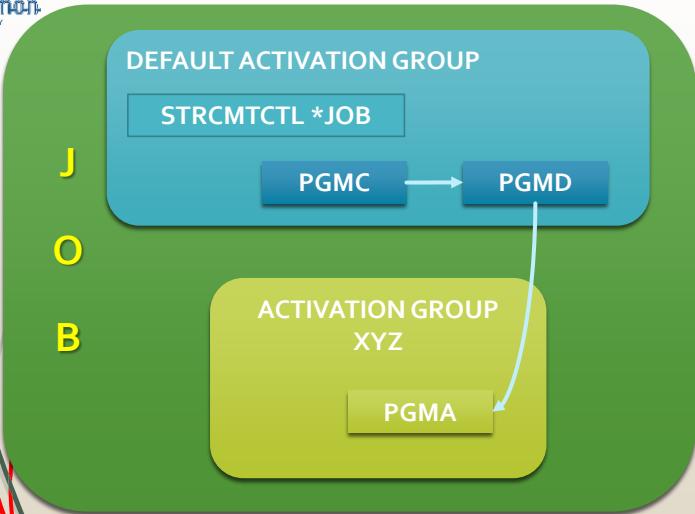
- **RCLACTGRP**: libera le risorse di un AG nominale
- **CRTSQLRPGI** parametro **CLOSQLCSR**: determina quando viene chiuso automaticamente un cursore SQL. *ENDACTGRP o *ENMOD.
- **OVR...** parametro **OVRSCOPE**: determina l'ambito di validità della sostituzione. *ACTGRPDFN o *CALLLVL o *JOB.
- **DLTOVR**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

Commitment control for everyone

16



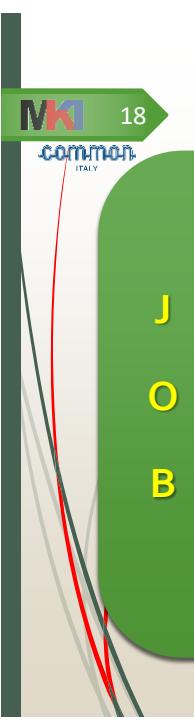
Comm.lt. quiz



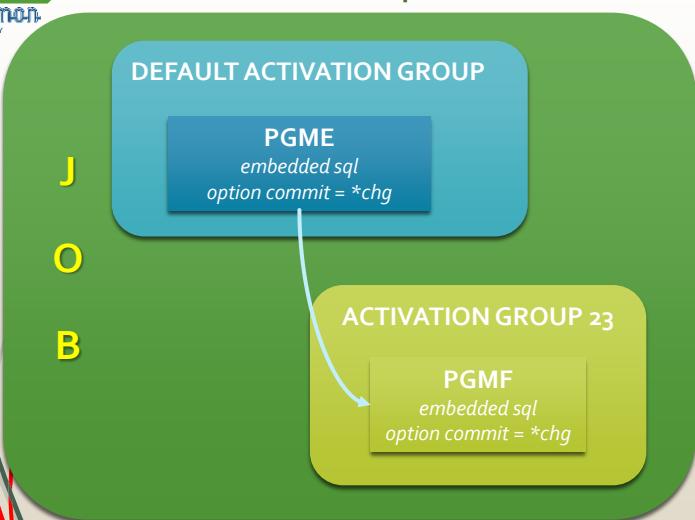
PGMA condivide la stessa definizione di controllo di sincronia del programma PGMC e PGMD?



17



Comm.lt. quiz



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

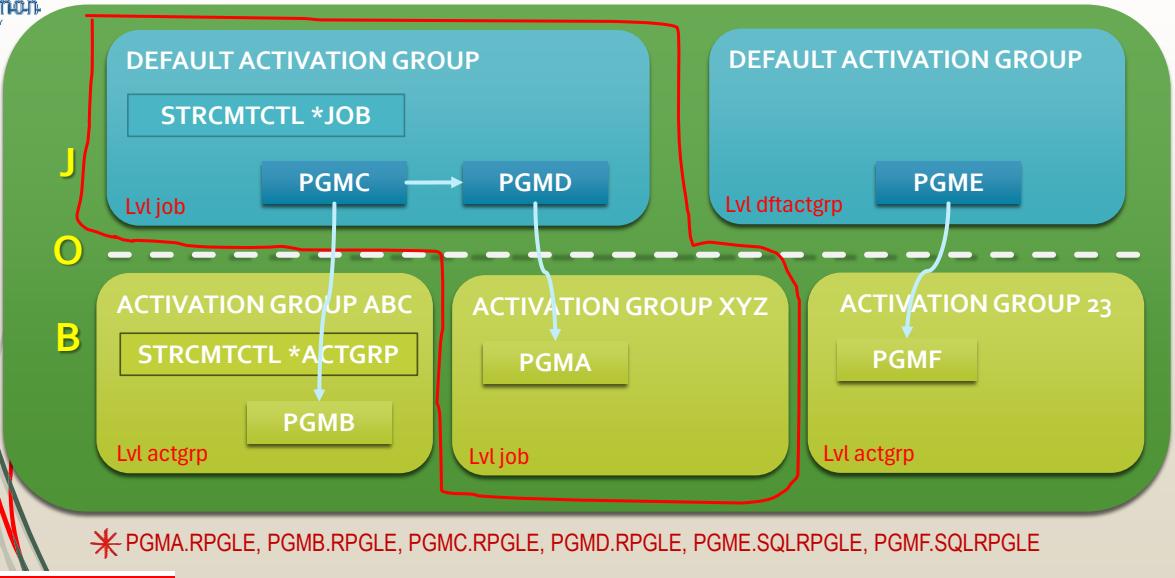


18

MK1
common
ITALY

19

Controllo sincronia - isolamento transazioni/1



19

MK1
common
ITALY

20

Controllo sincronia - isolamento transazioni/2

Annotazioni allo schema precedente

- ▶ PGMC avvia STRCMCTL a livello *JOB
- ▶ i programmi PGMD e PGMA chiamati da PGMC ereditano la definizione di commit a livello job
- ▶ il programma PGMB avvia STRCMCTL 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 STRCMCTL 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



Demo - CALL PGMC/1

WRKCMTDFN

DSPLY Avviato contr.sincr. *JOB. Invio per proseguire

| Opz | Definiz. | N° | Uten. | Nome | Sottopr. | Utente |
|-----|----------|--------|-------|--------|----------|--------|
| ■ | *JOB | 468667 | MRIVA | MRIVAA | *NONE | MRIVA |

| Opz | Definiz. | N° | lav. | ID LUW | Stato |
|-----|----------|--------|-------------------------------------|------------|-------|
| ■ | *JOB | 468667 | APPN.████████.X'BCFEBB5E077B'.00001 | RIPRISTINO | |

Visualizzazione stato definizione commit 28/01/26 12:45:29

| Lavoro: | Utente: | Numero: |
|---------|---------|---------|
| ██████A | ██████ | 468667 |

Sottopr. : *NONE
ID LUW (Logical Unit of Work) : APPN.████████.X'BCFEBB5E077B'.00001
ID spazio blocco : UDB_01000000000409ED
Definizione commit : *JOB
Gruppo attivazione :
Gruppo ASP : *SYSBAS
Ubicazione risorsa : NESSUNO
Livello blocco predefinito : *CHG
Utente :
Modifiche in sospeso locali : NO
Ruolo :
Stato : RIPRISTINO
Registrazione data/ora :

Risincronizz. in corso : NO
Operaz. euristica :
Lavoro attivo : SI
Lavoro server :
Ambito blocco : *JOB
Supero tempo transazione :
Attesa massima blocco :

Commitment control for everyone

21



Demo - CALL PGMC/2

WRKCMTDFN

DSPLY Eseguito update 000200. Proseguire? (S/N)

| Opz | Definiz. | N° | Risinc. | Modif. |
|-----|----------|--------|---------|---------|
| ■ | *JOB | 468667 | in | locali |
| | | | lav. | sospese |

Visualizzazione stato livello record 28/01/26 12:48:19

| Lavoro: | Utente: | Numero: |
|---------|---------|---------|
| ██████A | ██████ | 468667 |

Definizione commit : *JOB
Sistema : APPN
File Libreria Membro Commit Rollback Sospeso
EMPLOYEE MK1SAMPLE EMPLOYEE 0 0 1

File Libreria Membro Commit
EMPLOYEE MK1SAMPLE EMPLOYEE 6727

Ubicazione risorsa : LOCALE
Livello blocco predefinito : *CHG
Utente :
Modifiche in sospeso locali : SI

Visualizzazione dettagli voce di giornale INO 6 12:46:46

| Giornale | Libreria |
|--------------------|--------------------|
| QSQJRN | MK1SAMPLE |

Sequenza : 6729
Codice : R - Operazione su un record specifico
Tipo : UP - Aggiornamento, immagine-successiva

Oggetto : EMPLOYEE
Libreria : MK1SAMPLE
Membro : EMPLOYEE
Tipo : QDOS
Data : 26/01/26
Ora : 12:46:46,837008
Contrassegno : 0
Conteggio/RRN : 18
ID ciclo di commit : 6727
Visualizzazione delle voci di giornale
Le : QSQJRN Libreria : MK1SAMPLE
di sequenza più grande su questa schermata . : 0000000000000000
ere le opzioni e premere Invio.
sualizzazione intera voce

| Sequenza | Cod. | Tipo | Oggetto | Libreria | Lavoro | Or. |
|----------|------|------|-----------------------|-----------|---------|-----|
| 6727 | C | SC | start ciclo di commit | ██████A | ██████ | 12 |
| 6728 | R | UB | EMPLOYEE | MK1SAMPLE | ██████A | 12 |
| 6729 | R | UP | EMPLOYEE | MK1SAMPLE | ██████A | 12 |

22

MK1 23

common
ITALY

PGMC

Demo - CALL PGMC/3

WRKCMTDFN

DSPLY Avvio programma PGMD. Invio per proseguire.

| Opz | Definiz. | N° | Uten. | Nome | Sottopr. | Utente |
|--------|----------|------|-------|------|----------|--------|
| ■ *JOB | commit | lav. | lav. | lav. | *NONE | MRIVA |

| Visualizzazione stack di chiamata | | | | | | |
|-----------------------------------|----------|---------|------------------------------|------------------|------------|------------------|
| Lavoro: | Utente: | Numero: | -----Gruppo attivazione----- | | | |
| Sottoproc. | | | Tipo | Programma | Nome | Numero |
| | | | QCMD | QSYS | *DFTACTGRP | 0000000000000001 |
| | | | QUICMENU | QSYS | *DFTACTGRP | 0000000000000001 |
| 1 | QUIMNDRV | QSYS | *DFTACTGRP | 0000000000000001 | | |
| 2 | QUIMGFLW | QSYS | *DFTACTGRP | 0000000000000001 | | |
| 3 | QUICMD | QSYS | *DFTACTGRP | 0000000000000001 | | |
| | PGMC | MK1SQL | *DFTACTGRP | 0000000000000002 | | |
| | PGMC | MK1SQL | *DFTACTGRP | 0000000000000002 | | |
| → | PGMD | MK1SQL | *DFTACTGRP | 0000000000000002 | | |
| | PGMD | MK1SQL | *DFTACTGRP | 0000000000000002 | | |

Commitment control for everyone

23

MK1 24

common
ITALY

PGMC

Demo - CALL PGMC/4

WRKCMTDFN

DSPLY Eseguito update 000210. Proseguire? (S/N)

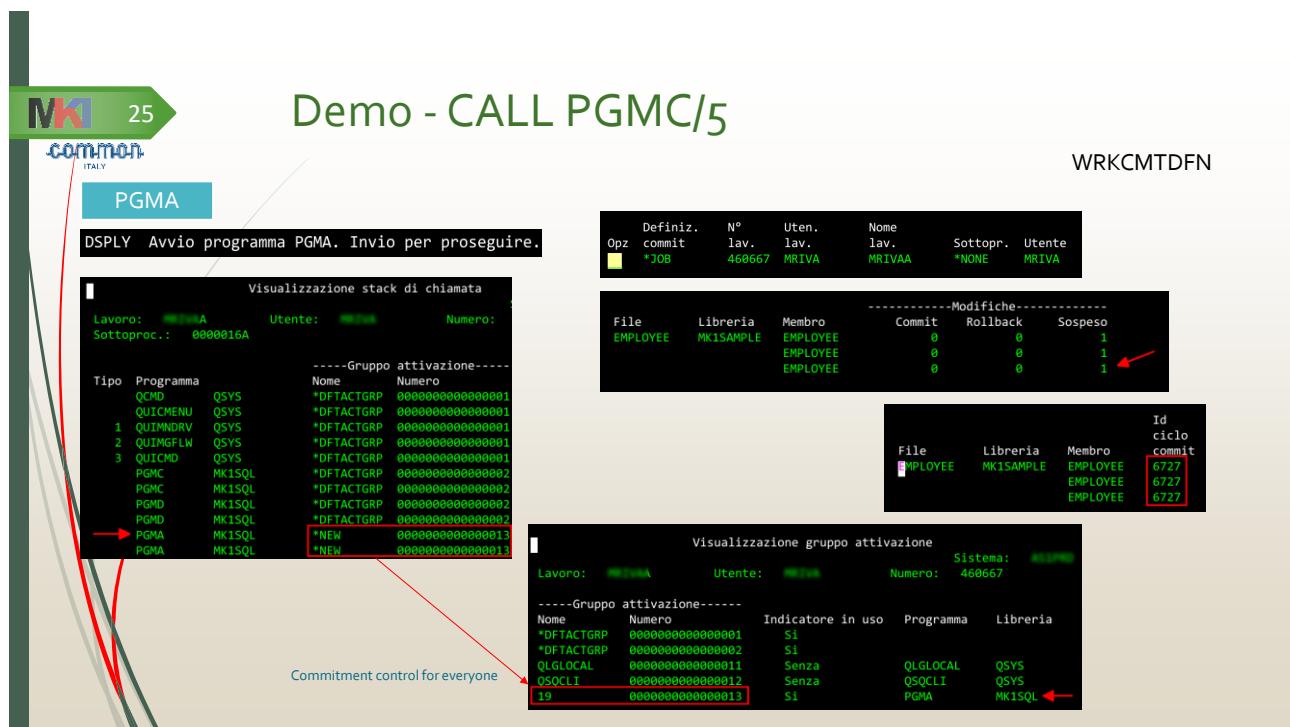
| Visualizz. stato livello record | | | | | | |
|-------------------------------------|-----------|----------|---------------------|----------|---------|---|
| Lavoro: | Utente: | Numero: | Sistema: 460667 | | | |
| Definizione commit : *JOB | | | | | | |
| File | Libreria | Membro | -----Modifiche----- | | | |
| EMPLOYEE | MK1SAMPLE | EMPLOYEE | Commit | Rollback | Sospeso | |
| | | EMPLOYEE | 0 | 0 | 1 | 1 |

| File | Libreria | Membro | Id | ciclo |
|----------|-----------|----------|------|-------|
| EMPLOYEE | MK1SAMPLE | EMPLOYEE | 6727 | 6727 |
| | | | | |

| Opz | Giornale | Libr. | Id | ciclo | Blocchi | Modifiche |
|-----|----------|-----------|------|--------|---------|-----------|
| ■ | QSQJRN | MK1SAMPLE | 6727 | commit | 2 | 2 |
| | | | | | | |

Commitment control for everyone

24



25

Demo - CALL PGMC/6

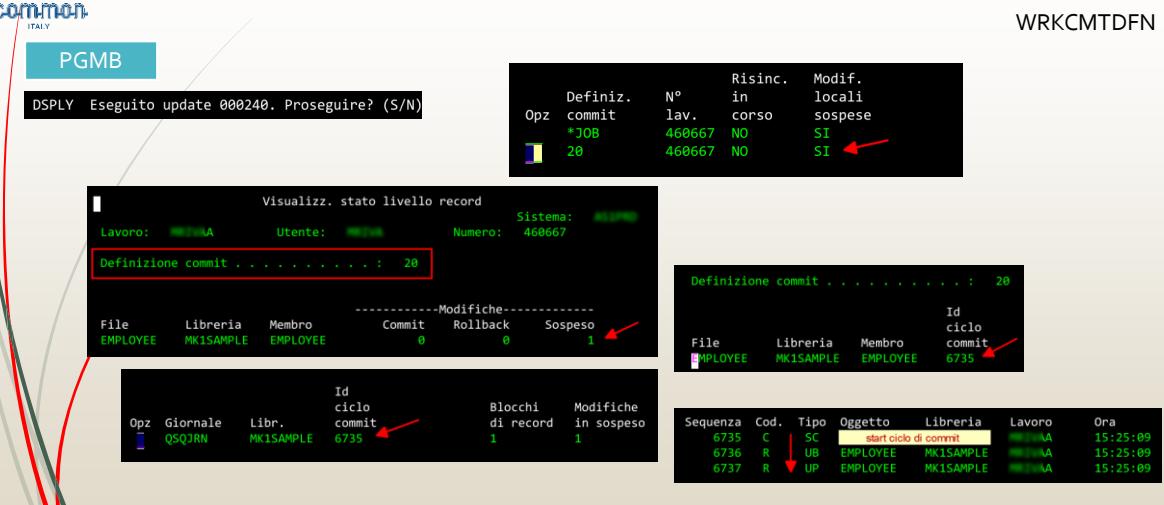


26

MK1 27

Demo - CALL PGMC/7

WRKCMTDFN



Commitment control for everyone

27

MK1 28

Demo - CALL PGMC/8

WRKCMTDFN



Commitment control for everyone

28

MK1 29

common
ITALY

PGMB

DSPLY PGMB arresterà il contr.sincr. Invio per proseguire
 DSPLY Arresto controllo di sincronia con errori
 ENDCMTCTL non è consentito. Sono presenti delle modifiche in sospeso.

DSPLY PGMB ha chiuso il file EMPLOYEE. Invio per proseguire

DSPLY PGMB arresterà il contr.sincr. Invio per proseguire

Commitment control for everyone

Visualizz. stato livello record Sistema: ASIPRD
 Lavoro: MRIVA Utente: MRIVA Numero: 460667
 Definizione commit : *JOB
 File Libreria Membro Liv blocco Stato Risoluzione
 EMPLOYEE MKISAMPLE EMPLOYEE *CHG APERTO accessi contemp.
 *WAIT

Visualizz. stato livello record Sistema: ASIPRD
 Lavoro: MRIVA Utente: MRIVA Numero: 460667
 Definizione commit : *JOB
 File Libreria Membro Liv blocco Stato Risoluzione
 accessi contemp.
 (Non ci sono modifiche a livello record sotto controllo di commit.)

| Definiz. | N° | Uten. | Nome | Sottopr. | Utente |
|------------|--------|-------|--------|----------|--------|
| Opz commit | lav. | lav. | lav. | *NONE | MRIVA |
| *JOB | 460667 | MRIVA | MRIVAA | | |

29

MK1 30

common
ITALY

PGMC

DSPLY PGMC arresterà il contr.sincr. Invio per proseguire
 DSPLY Arresto controllo di sincronia con errori

Chiusura PGMC

SIGNOFF

Visualizz. stato livello record Sistema: ASIPRD
 Lavoro: MRIVA Utente: MRIVA Numero: 460667
 Definizione commit : *JOB
 File Libreria Membro Liv blocco Stato Risoluzione
 EMPLOYEE MKISAMPLE EMPLOYEE *CHG APERTO accessi contemp.
 *WAIT

| Definiz. | N° | Uten. | Nome | Sottopr. | Utente |
|------------|--------|-------|--------|----------|--------|
| Opz commit | lav. | lav. | lav. | *NONE | MRIVA |
| *JOB | 460667 | MRIVA | MRIVAA | | |

(Non ci sono definizioni di commit attive)

CPF8356 Diagnostica 30 20/01/26 15:41:04,336486 QTNEND QSYS 2075 QWTPITP2 QSYS 0439
 Messaggio . . . : Il controllo di commit è terminato con 3 modifiche non sottoposte a commit.
 5770SSI V7RSM0 220415 Registrazione lavoro ASIPRD 20/01/26 15:41:04 CET Pag. 14
 Nome lavoro : MRIVA Utente : MRIVA Numero : 460667
 Descrizione lavoro : QPF2JOB Libreria : QGPL
 IDMSG TIPO GRAV DATA ORA DA PGM LIBERTA INST A PGM LIBRERIA INST
 Causa . . . : È stato eseguito il rollback di modifiche in sospeso poiché vi erano modifiche non sottoposte a commit nel momento in cui è terminato il controllo di commit per la definizione di commit *JOB con un id LWM (logical unit of work/unità logica di lavoro) APPN. X'BCFEBB5E077B'.00001.

30

Comm.lt. quiz

```

graph TD
    DA[DEFAULT ACTIVATION GROUP] --- PGMG[PGMG]
    DA --- PGMH[PGMH]
    PGMG --- ACTGRP[ACTGRP]
    PGMH --- ACTGRP
    style DA fill:#009640,color:#fff
    style ACTGRP fill:#009640,color:#fff
    style PGMG fill:#009640,color:#fff
    style PGMH fill:#009640,color:#fff
    style DA stroke:#009640
    style ACTGRP stroke:#009640
    style PGMG stroke:#009640
    style PGMH stroke:#009640
    
```

The diagram illustrates the activation groups for a job. A large green rounded rectangle labeled "JOB" contains two activation groups: "DEFAULT ACTIVATION GROUP" and "ACTGRP". The "DEFAULT ACTIVATION GROUP" is associated with the program "PGMG.RPGLE" and contains the command "STRCMTCTL *JOB". The "ACTGRP" activation group is associated with the program "PGMH.RPGLE" and contains the command "STRCMTCTL *ACTGRP". A curved arrow points from "PGMG" to "ACTGRP", indicating that "PGMG" can open a definition of control synchronization at the "ACTGRP" level.

QUESTION: PGMH può aprire una definizione di controllo sincronia a livello *ACTGRP?

YES **NO**

31

Controllo sincronia - isolamento transazioni/3

The diagram consists of three colored boxes connected by arrows:

- A green box states: "Solo programmi in ambiente ILE possono aprire transazioni in un gruppo di attivazione diverso da quello di default".
- A light green box with a yellow border states: "in un lavoro possono esistere più transazioni contemporanee e indipendenti se e solo se nel job sono in esecuzione uno o più programmi ILE".
- A blue box states: "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

32

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

33

Livelli di isolamento delle transazioni

integrità e sicurezza del dato

| Livello | Descrizione | Attivo | Letture 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 consolidati |
| *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>

34

Livelli di isolamento delle transazioni/2

- ▶ **Letture 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

35

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>

36

Prerequisiti per attivare controllo di sincronia

Specifiche di controllo e definizione file

Opzioni precompilatore SQL

Commitment control for everyone

37

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

38

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

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>

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

41



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

42

Commit e rollback

Consolidare o annullare una transazione

Commitment control for everyone

43

Comm.it. quiz

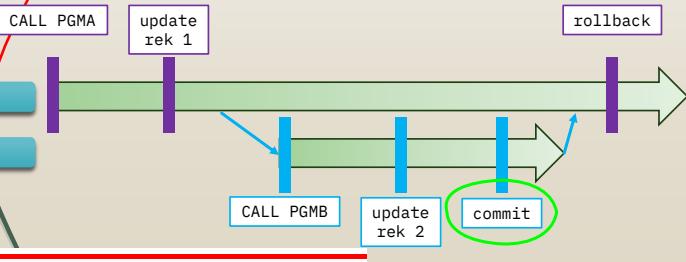
PGMA e PGMB vengono eseguiti entrambi nel *DFTACTGRP con COMMIT = *CHG

- PGMA esegue un update del record 1, quindi chiama il PGMB
- PGMB esegue un update del record 2, esegue commit e ritorna al chiamante
- PGMA esegue rollback



Quanti record vengono consolidati?

0 1 2



44

Esempio transazioni/1

PGMA -> PGMB
dft

- PGMA e PGMB vengono eseguiti entrambi nel *DFTACTGRP con COMMIT = *CHG
 - PGMA esegue un update, quindi chiama il PGMB
 - PGMB esegue un update di un altro record, esegue commit e ritorna al chiamante
 - PGMA esegue rollback

| Programma | Azione | Ambito ctl sincronia | ID LUW | Modifiche in sospeso |
|-----------|-----------------|----------------------|------------------------------------|----------------------|
| PGMA | update 1 record | *DFTACTGRP | APPN.sysname.X'48953C039E92'.00001 | 1 ✓ ↗ |
| PGMB | update 1 record | *DFTACTGRP | APPN.sysname.X'48953C039E92'.00001 | 2 ✓ ↗ |
| PGMB | commit | *DFTACTGRP | APPN.sysname.X'48953C039E92'.00002 | 0 |
| PGMB | return | | | |
| PGMA | rollback | *DFTACTGRP | APPN.sysname.X'48953C039E92'.00003 | 0 |

RISULTATO: entrambi gli update vengono consolidati sul db dal commit eseguito da PGMB

Commitment control for everyone

TCMT1A.SQLRPGLE, TCMT1B.SQLRPGLE

45

Comm.lt. quiz

PGMA viene eseguito in *DFTACTGRP e PGMB in ACTGRP(*NEW) con COMMIT = *CHG

- PGMA esegue un update del record 1, quindi chiama il PGMB
- PGMB esegue un update del record 2, esegue commit e ritorna al chiamante
- PGMA esegue rollback



Quanti record vengono consolidati?

0 1 2



46

MK1
common
ITALY

47

Esempio transazioni/2

PGMA
dftPGMB
new

- PGMA viene eseguito in *DFTACTGRP e PGMB in ACTGRP(*NEW) con COMMIT = *CHG

PGMA esegue un update, quindi chiama il PGMB

PGMB esegue un update di un altro record, esegue commit e ritorna al chiamante

PGMA esegue rollback

| Programma | Azione | Ambito ctl sincronia | ID LUW | Modifiche in sospeso |
|-----------|-----------------|--|------------------------------------|----------------------|
| PGMA | update 1 record | *DFTACTGRP | APPN.sysname.X'5A75B9A8C288'.00001 | 1 ✗ ↗ |
| PGMB | update 1 record | act group 19 | APPN.sysname.X'5A75E299C288'.00001 | 1 ✓ ↘ |
| PGMB | commit | act group 19 | APPN.sysname.X'5A75E299C288'.00002 | 0 ↗ |
| PGMB | return | <i>si chiude automaticamente act group 19 perché gestito dal sistema</i> | | |
| PGMA | rollback | *DFTACTGRP | APPN.sysname.X'5A75B9A8C288'.00002 | 0 |

RISULTATO: solo l'update eseguito da PGMB viene consolidato sul db

TCMT2A.SQLRPGLE, TCMT2B.SQLRPGLE

47

MK1
common
ITALY

48

Comm.lt. quiz

PGMA viene eseguito in *DFTACTGRP e PGMB in ACTGRP(*NEW) con COMMIT = *CHG

- PGMA esegue un update del record 1, quindi chiama il PGMB
- PGMB esegue un update del record 2 e ritorna al chiamante
- PGMA esegue rollback

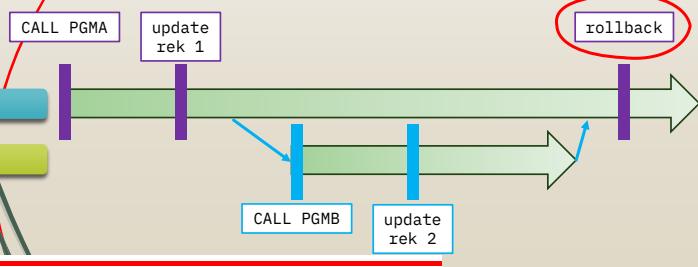


Quanti record vengono consolidati?

0 1 2



Chi ha consolidato il record n. 2?



48

49

Esempio transazioni/3

PGMA
dft

PGMB
new

- ▶ PGMA viene eseguito in *DFTACTGRP e PGMB in ACTGRP(*NEW) con COMMIT = *CHG

PGMA esegue un update, quindi chiama il PGMB

PGMB esegue un update di un altro record, NON esegue commit esplicito e ritorna al chiamante
PGMA esegue rollback

| Programma | Azione | Ambito ctl sincronia | ID LUW | Modifiche in sospeso |
|-----------|-----------------|--|------------------------------------|----------------------|
| PGMA | update 1 record | *DFTACTGRP | APPN.sysname.X'5A75B9A8C288'.00001 | 1 ✘ ↗ |
| PGMB | update 1 record | act group 19 | APPN.sysname.X'5A75E299C288'.00001 | 1 ✓ ↗ |
| PGMB | return | <i>si chiude act group 19 e viene eseguito un commit隐式</i> | | |
| PGMA | rollback | *DFTACTGRP | APPN.sysname.X'5A75B9A8C288'.00002 | 0 |

RISULTATO: solo l'update eseguito da PGMB viene consolidato sul db

Commitment control for everyone

TCMT3A.SQLRPGLE, TCMT3B.SQLRPGLE

49

50

Esempio transazioni/4

PGMA
dft

PGMB
name

- ▶ PGMA viene eseguito in *DFTACTGRP e PGMB in ACTGRP('PGMB') con COMMIT = *CHG

PGMA esegue un update, quindi chiama il PGMB

PGMB esegue un update di un altro record, esegue commit e ritorna al chiamante
PGMA esegue rollback

| Programma | Azione | Ambito ctl sincronia | ID LUW | Modifiche in sospeso |
|-----------|-----------------|--|------------------------------------|----------------------|
| PGMA | update 1 record | *DFTACTGRP | APPN.sysname.X'5A75B9A8C288'.00001 | 1 ✘ ↗ |
| PGMB | update 1 record | act group PGMB | APPN.sysname.X'5A75E299C288'.00001 | 1 ✘ ↗ |
| PGMB | return | <i>non si chiude act group PGMB perché non gestito dal sistema e la transazione rimane sospesa</i> | | |
| PGMA | rollback | *DFTACTGRP | APPN.sysname.X'5A75B9A8C288'.00002 | 0 |

RISULTATO: l'update eseguito da PGMB rimane in sospeso fino a quando non viene chiuso l'actgrp PGMB

Commitment control for everyone

TCMT4A.SQLRPGLE, TCMT4B.SQLRPGLE

50

MK1
common
ITALY

51

Esempio transazioni/5

PGMA
dft

PGMB
new

- ▶ PGMA viene eseguito in *DFTACTGRP e PGMB in ACTGRP(*NEW) con COMMIT = *CHG
- PGMA esegue un update, quindi chiama il PGMB con estensore errore CALLP(E)
- PGMB esegue un update di un altro record e prima di eseguire commit si interrompe per un'eccezione non prevista, quindi ritorna a PGMA che prosegue all'istruzione successiva
- PGMA esegue rollback

| Programma | Azione | Ambito ctl sincronia | ID LUW | Modifiche in sospeso |
|-----------|-----------------|--|------------------------------------|----------------------|
| PGMA | update 1 record | *DFTACTGRP | APPN.sysname.X'5A75B9A8C288'.00001 | 1 ✘ ↗ |
| PGMB | update 1 record | act group 19 | APPN.sysname.X'5A75E299C288'.00001 | 1 ✘ ↗ |
| PGMB | eccezione 🐞 | si chiude act group 19 e viene eseguito un rollback implicito | | |
| PGMA | rollback | *DFTACTGRP | APPN.sysname.X'5A75B9A8C288'.00002 | 0 |

RISULTATO: nessun update viene consolidato sul db

Commitment control for everyone

✳ TCMT5A.SQLRPGLE, TCMT5B.SQLRPGLE

51

MK1
common
ITALY

52

Comm.lt. quiz

PGMA viene eseguito in *DFTACTGRP e PGMB in ACTGRP(*NEW) con COMMIT = *CHG

- PGMA esegue un update del record 1, quindi chiama il PGMB
- PGMB esegue un update del record 2 e ritorna al chiamante
- PGMA si chiude normalmente
- viene chiuso normalmente il job (signoff)

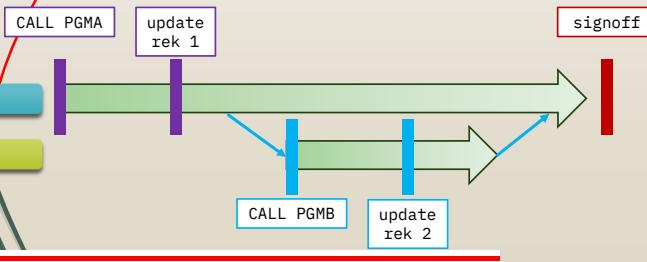


Quanti record vengono consolidati?

0

1

2



52

53

Esempio transazioni/6

PGMA
dft

PGMB
new

- ▶ PGMA viene eseguito in *DFTACTGRP e PGMB in ACTGRP(*NEW) con COMMIT = *CHG

PGMA esegue un update, quindi chiama il PGMB

PGMB esegue un update di un altro record, NON esegue commit esplicito e ritorna al chiamante
PGMA si chiude normalmente senza eseguire commit e poi viene chiuso normalmente il job

| Programma | Azione | Ambito ctl sincronia | ID LUW | Modifiche in sospeso |
|-----------|-----------------|---|------------------------------------|----------------------|
| PGMA | update 1 record | *DFTACTGRP | APPN.sysname.X'5A75B9A8C288'.00001 | 1 ✘ ↗ |
| PGMB | update 1 record | act group 19 | APPN.sysname.X'5A75E299C288'.00001 | 1 ✓ ↗ |
| PGMB | return | <i>si chiude act group 19 e viene eseguito un commit隐式</i> | | |
| PGMA | return | <i>transazione rimane sospesa con il rek aggiornato da PGMA ancora allocato</i> | | |
| == | endjob | <i>si chiude il job e viene eseguito un rollback隐式</i> | | |

RISULTATO: solo l'update eseguito da PGMB viene consolidato sul db

TCMT6A.SQLRPGLE, TCMT6B.SQLRPGLE

53

54

Esempio transazioni/7

PGMA -> PGMB
new - caller

- ▶ PGMA viene eseguito in ACTGRP(*NEW) e PGMB in ACTGRP(*CALLER) con COMMIT = *CHG

PGMA esegue un update, quindi chiama il PGMB

PGMB esegue un update di un altro record, NON esegue commit esplicito e ritorna al chiamante
PGMA esegue commit

| Programma | Azione | Ambito ctl sincronia | ID LUW | Modifiche in sospeso |
|-----------|-----------------|---|------------------------------------|----------------------|
| PGMA | update 1 record | act group 19 | APPN.sysname.X'48953C039E92'.00001 | 1 ✓ ↗ |
| PGMB | update 1 record | act group 19 | APPN.sysname.X'48953C039E92'.00001 | 1 ✓ ↗ |
| PGMB | return | <i>NON si chiude act group 19 e quindi non viene eseguita nessuna azione隐式</i> | | |
| PGMA | commit | act group 19 | APPN.sysname.X'48953C039E92'.00002 | 0 |
| PGMA | return | <i>si chiude act group 19 e viene eseguito un commit隐式 anche se non eseguito esplicitamente da pgmA</i> | | |

RISULTATO: entrambi gli update vengono consolidati sul db dal commit eseguito da PGMA

TCMT7A.SQLRPGLE, TCMT7B.SQLRPGLE

54

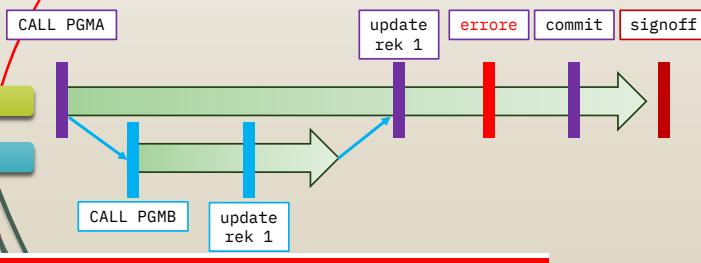
MK1
common
ITALY

55

Comm.lt. quiz

PGMA viene eseguito in ACTGRP(*NEW) e PGMB in *DFTACTGRP con COMMIT = *CHG

- PGMA chiama il PGMB
- PGMB esegue update del record 1 e ritorna al chiamante
- PGMA tenta di eseguire update sullo stesso record modificato da PGMB



Quanti record vengono consolidati?

0

1

55

MK1
common
ITALY

56

Esempio transazioni/8

PGMA
new

PGMB
dft

PGMA viene eseguito in ACTGRP(*NEW) e PGMB in *DFTACTGRP con COMMIT = *CHG
PGMA chiama il PGMB; PGMB esegue un update di un record ma NON esegue commit esplicito e ritorna al chiamante; PGMA tenta di eseguire update sullo stesso record modificato da PGMB

| Programma | Azione | Ambito ctl sincronia | ID LUW | Modifiche in sospeso |
|-----------|----------------------|----------------------|---|----------------------|
| PGMA | si apre act group 22 | | APPN.sysname.X'5A779697C288'.00001 | |
| PGMB | update record n | *DFTACTGRP | APPN.sysname.X'5A77D18EC288'.00001 | 1 ✘ |
| PGMB | return | | NON si chiude dftactgrp e non viene eseguita nessuna azione implicita transazione rimane sospesa con il rek aggiornato da PGMB allocato | |
| PGMA | update record n | act group 22 | errore record allocato | 🐞 |
| PGMA | commit | act group 22 | APPN.sysname.X'5A779697C288'.00002 | |
| PGMA | return | | si chiude act group 22 ma la transazione del *dftactgrp rimane sospesa con il rek aggiornato da PGMB ancora allocato | |
| -- | endjob | | si chiude il job e viene eseguito un rollback隐式 | |

RISULTATO: l'update di PGMA fallisce, l'update di PGMB rimane in sospeso fino a che si chiude il job

* TCMT8A.SQLRPGLE, TCMT8B.SQLRPGLE

56

Esempio transazioni/9

PGMA
newPGMB
new

- ▶ PGMA e PGMB vengono eseguiti in due ACTGRP(*NEW) con COMMIT = *CHG
- PGMA esegue un update, quindi chiama il PGMB
- PGMB esegue un update di un altro record, esegue commit e ritorna al chiamante
- PGMA esegue commit

| Programma | Azione | Ambito ctl sincronia | ID LUW | Modifiche in sospeso |
|-----------|-----------------|----------------------|------------------------------------|----------------------|
| PGMA | update 1 record | act group 21 | APPN.sysname.X'5A6EF3A9C263'.00001 | 1 ✓ ↘ |
| PGMB | update 1 record | act group 22 | APPN.sysname.X'5A6FE4BBC263'.00001 | 1 ✓ ↘ |
| PGMB | commit | act group 22 | APPN.sysname.X'5A6FE4BBC263'.00002 | 0 ↗ |
| PGMA | commit | act group 21 | APPN.sysname.X'5A6EF3A9C263'.00002 | 0 ↗ |

RISULTATO: ogni update eseguito da PGMA e PGMB viene consolidato sul db dal proprio commit in modo indipendente

Commitment control for everyone

✳ TCMT9A.SQLRPGLE, TCMT9B.SQLRPGLE

57



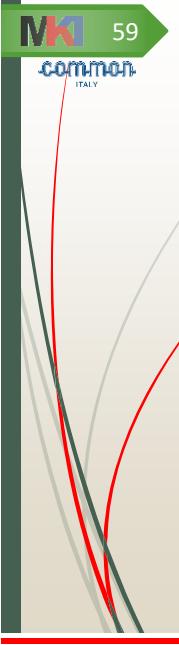
Commit o rollback implicito

| Contesto | Azione implicita |
|--|------------------|
| ✗ Programma si interrompe con eccezione non prevista | rollback |
| ✓ Gruppo di attivazione si chiude normalmente | commit |
| ✗ Gruppo di attivazione si chiude con errori | rollback |
| ✓ Definizioni di commit in ambito *JOB o *DFTACTGRP | nessuna |
| ✓ Chiusura di un lavoro *JOB normale o anomala | rollback |

<https://www.ibm.com/docs/en/i/7.6.0?topic=control-system-initiated-end-commitment>

Commitment control for everyone

58



Concorrenza di accesso

Livello di isolamento delle transazioni

Commitment control for everyone

59



Comm.it. quiz

Cosa succede se tento di leggere i record modificati in una transazione in esecuzione su un altro job o activation group?

Dipende dall'impostazione del livello di isolamento delle transazioni:
*CHG, *CS, *ALL, *RR

Il livello di isolamento di default è *CHG

60



61

Concorrenza di accesso: esempi

common
ITALY

Programma che apre una transazione per aggiornare dei record

```
dcl-s Domanda char(51);
dcl-s Risposta char(1);

exec sql
  set option COMMIT = *CHG;
  *CS;
  *RS;
  *RS;

exec sql
  update EMPLOYEE
    set BONUS = BONUS + 10
    where EMPNO = '000010';

Domanda = 'Confermi la modifica di EMPLOYEE? (S/N)';
dsply Domanda ' ' Risposta;

if %upper(Risposta) = 'S';
  exec sql
    commit;
else;
  exec sql
    rollback;
endif;

*inlr = *on;
return;
```

* TCMTCTL1*.SQLRPGLE

Programma che legge lo stesso record durante la transazione

```
dcl-s Domanda char(51);
dcl-s Risposta char(1);
dcl-s wBonus packed(9:2);

exec sql
  set option COMMIT = *CHG;
  *CS;
  *RS;
  *RS;

dou %upper(Risposta) <> 'S';
  exec sql
    select BONUS
      into :wBonus
      from EMPLOYEE
      where EMPNO = '000010';

  Domanda = 'BONUS di EMPNO 000010 è ' +
    %char(wBonus) + '. Rileggo? (S/N)';
  dsply Domanda ' ' Risposta;
enddo;

*inlr = *on;
return;
```

* TCMTCTL2*.SQLRPGLE

Commitment control for everyone

61



62

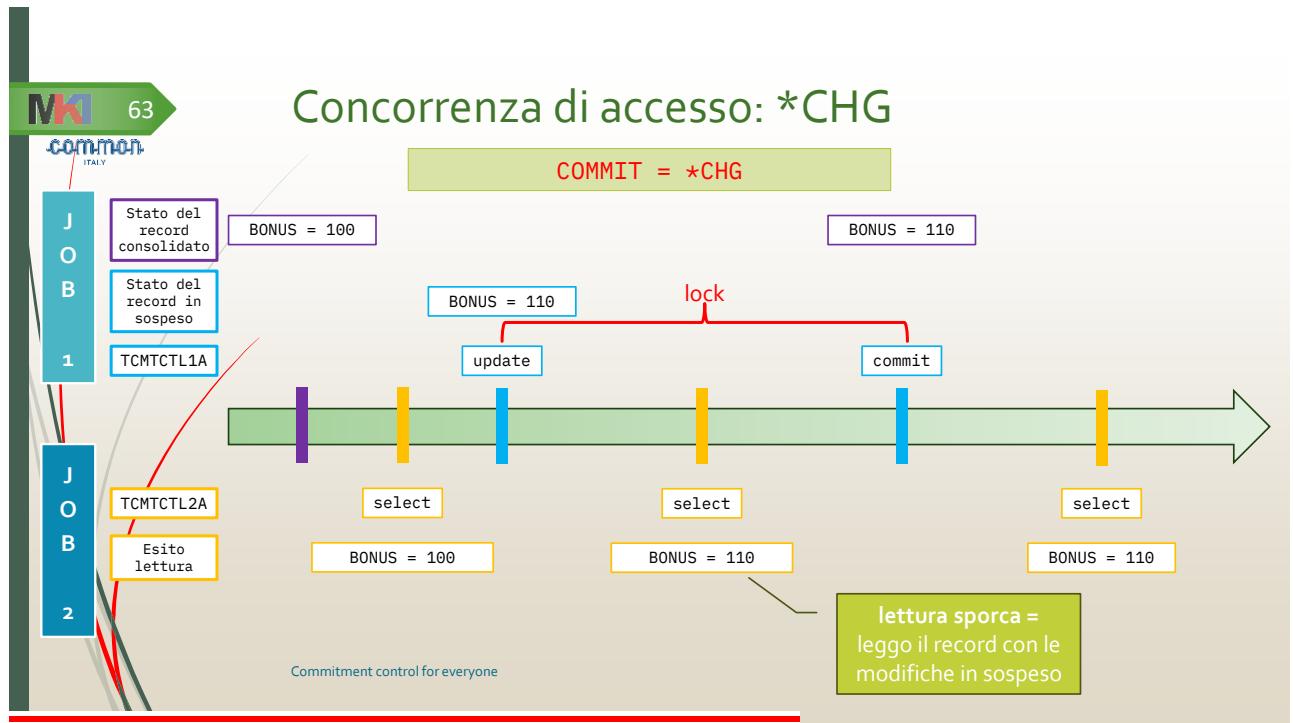
*CHG vs *CS

common
ITALY

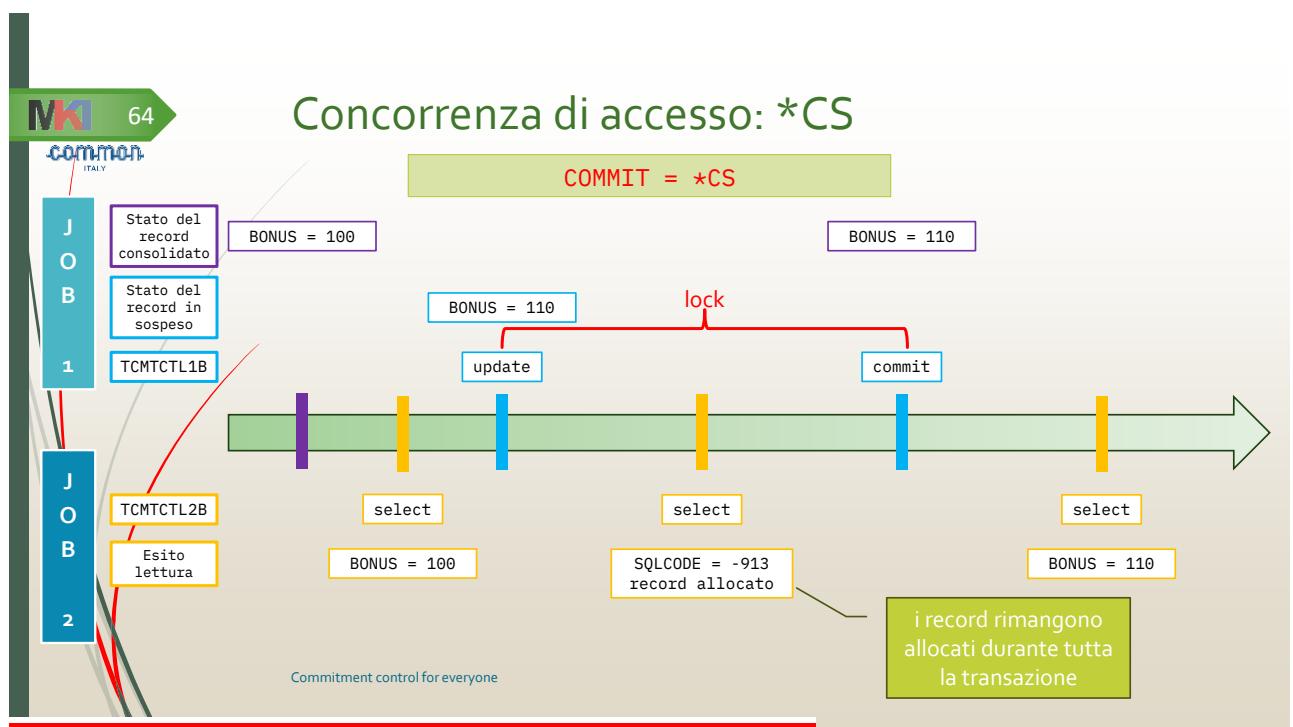
| Livello | Descrizione | Attivo | Lettura sporche | Record fantasma | Allocazione record letti | Note |
|----------------|--|--------|-----------------|-----------------|--------------------------|--|
| *CS/CS | LETTURA SINCRONIZZATA (Cursor Stability or Read Committed) | ✓ | ✗ | ✓ | ✗ | Garantisce 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 |

Commitment control for everyone

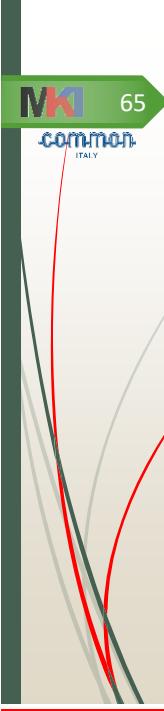
62



63



64

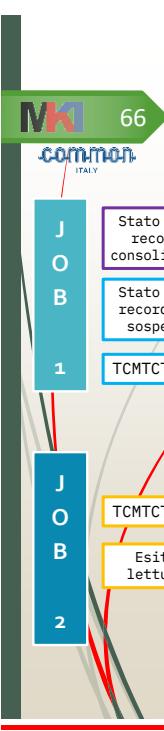


Concurrent access resolution

- ▶ opzione CONACC
- ▶ *CURCMT (currently committed): vengono letti i dati correntemente in stato consolidato ignorando le modifiche in sospeso
- ▶ *WAIT: attende che la transazione che alloca il record venga chiusa con commit o rollback; se viene superato il tempo di attesa del record (WAITRCD) viene restituito sqlcode = -913
- ▶ *DFT: non viene modificata l'impostazione e viene ereditata quella precedentemente impostata

Commitment control for everyone

65



Concorrenza di accesso con *CURCMT

COMMIT = *CS, CONACC = *CURCMT

Commitment control for everyone

66

MK1 common ITALY

67

***CHG vs *ALL**

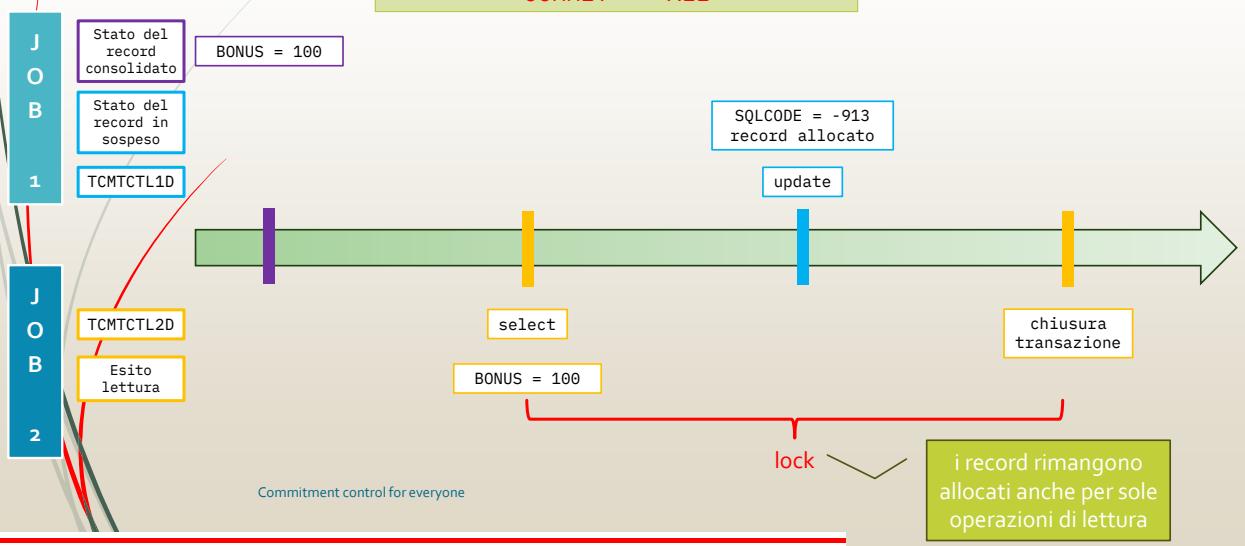
| Livello | Descrizione | Attivo | Lettura sporche | Record fantasma | Allocazione record letti | Note |
|----------------|--|--------|-----------------|-----------------|--------------------------|---|
| *ALL/RS | LETTURA STABILE (Read Stability) | ✓ | ✗ | ✓ | ✓ | Garantisce che i record letti non possono 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 |

Commitment control for everyone

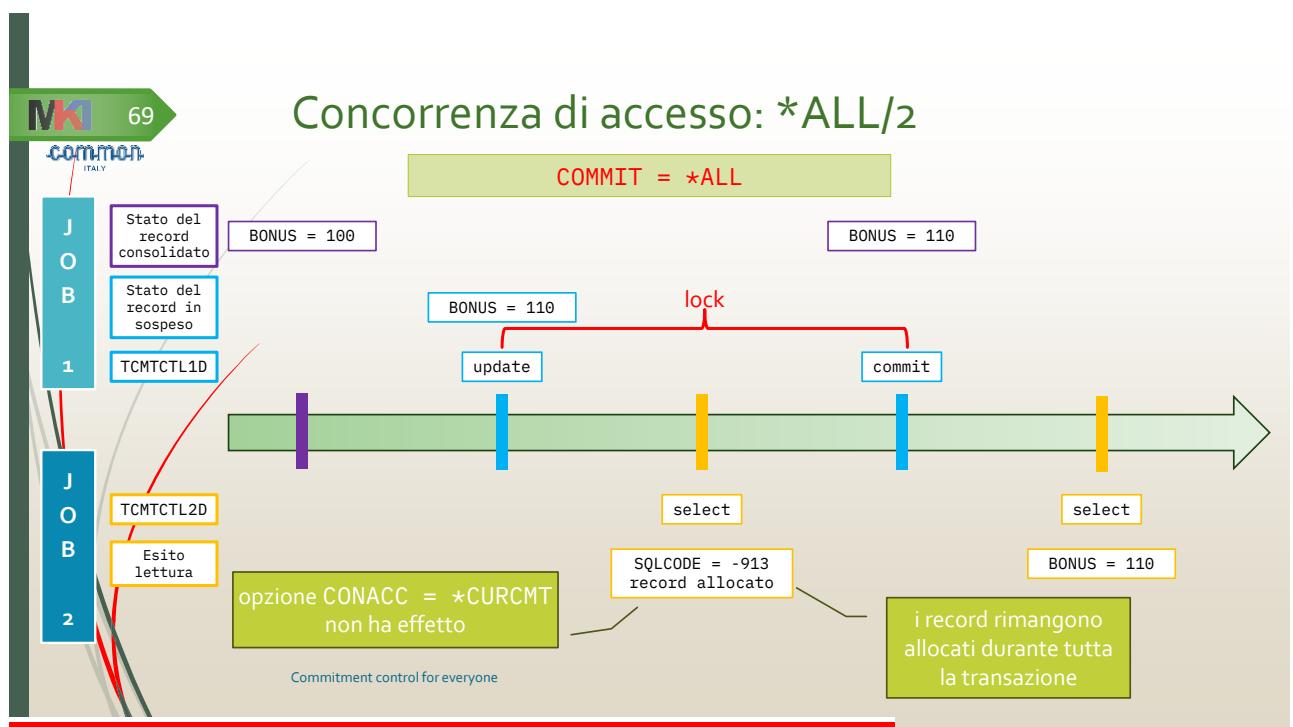
67

MK1 common ITALY

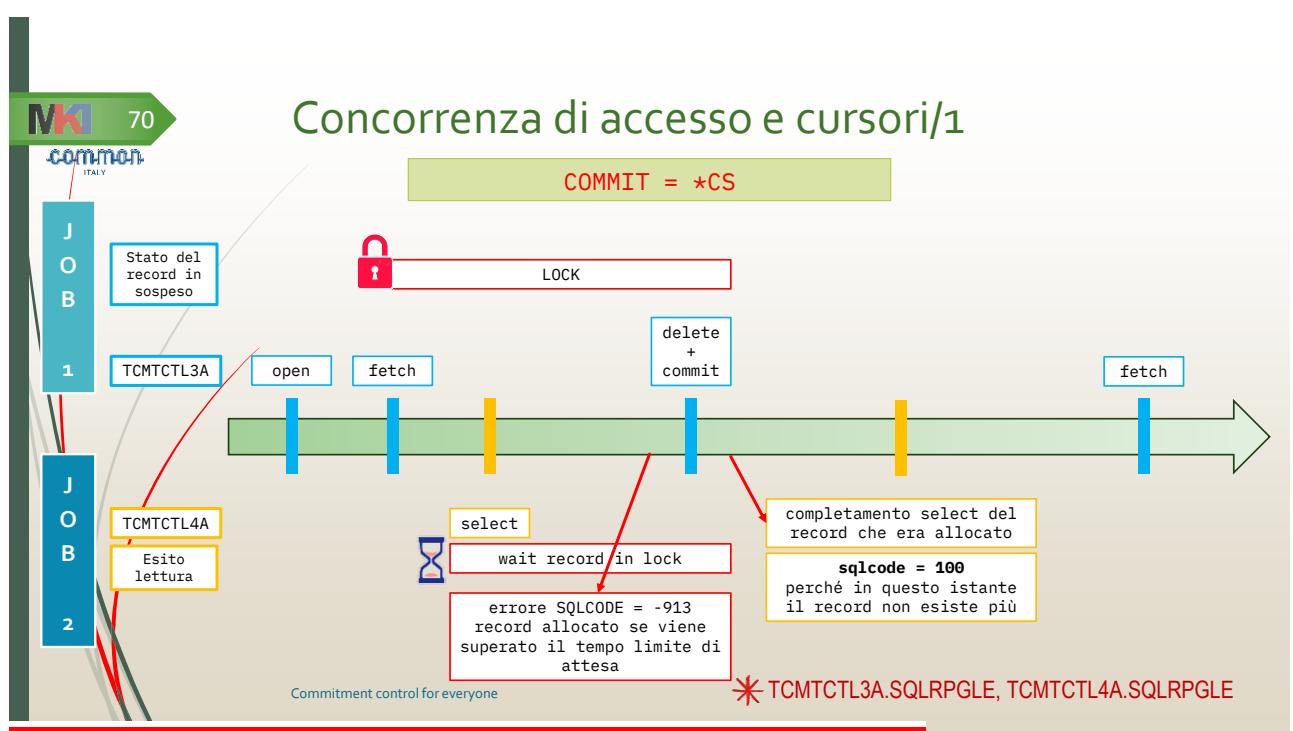
68

Concorrenza di accesso: *ALL/1**COMMIT = *ALL**

68



69

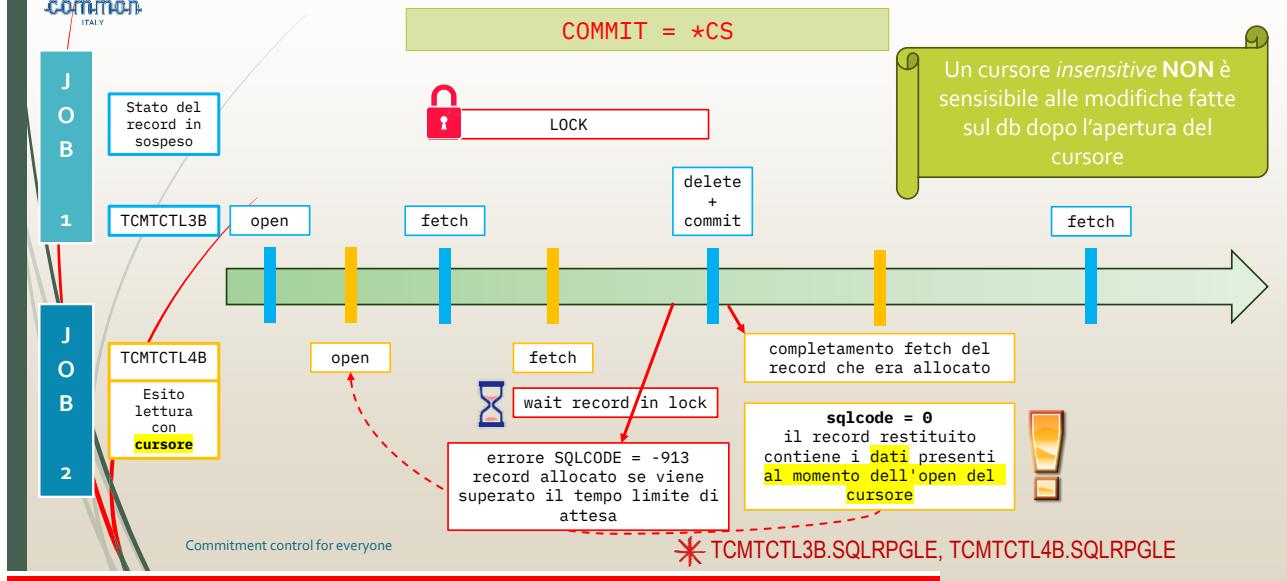


70

MK1
common
ITALY

71

Concorrenza di accesso e cursori/2

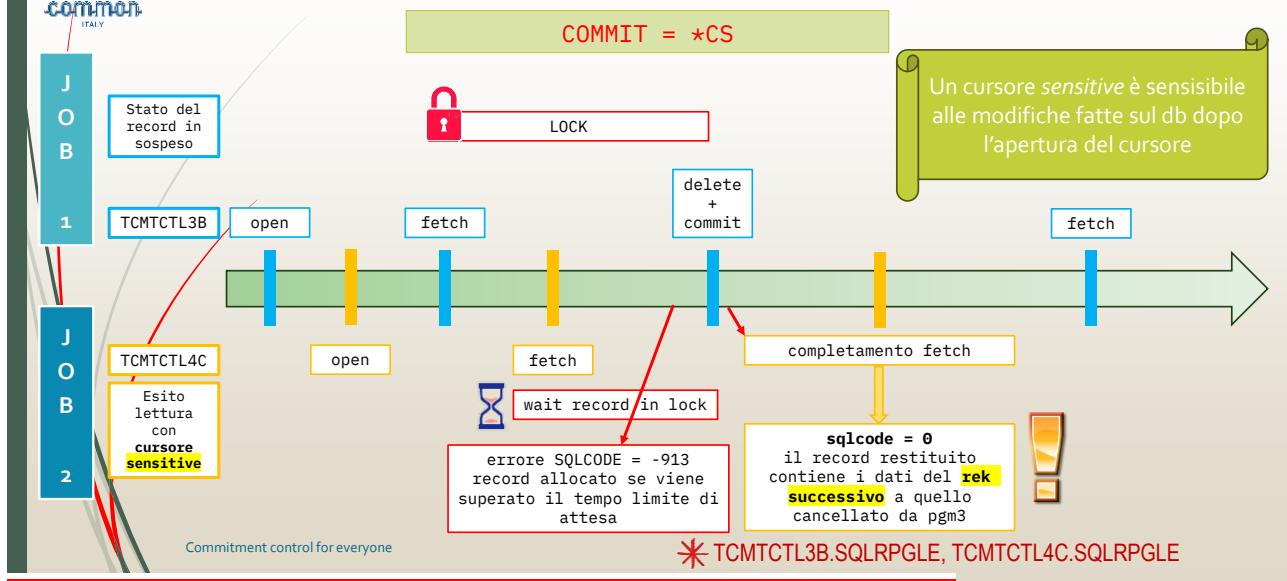


71

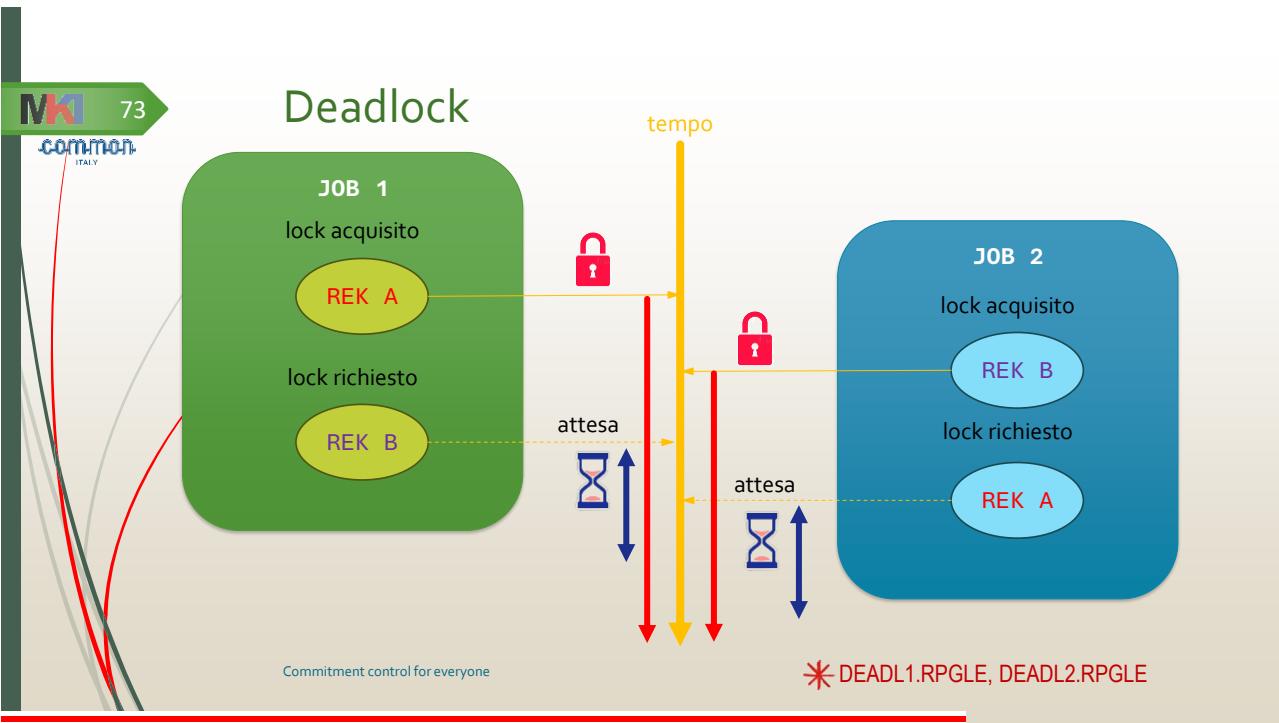
MK1
common
ITALY

72

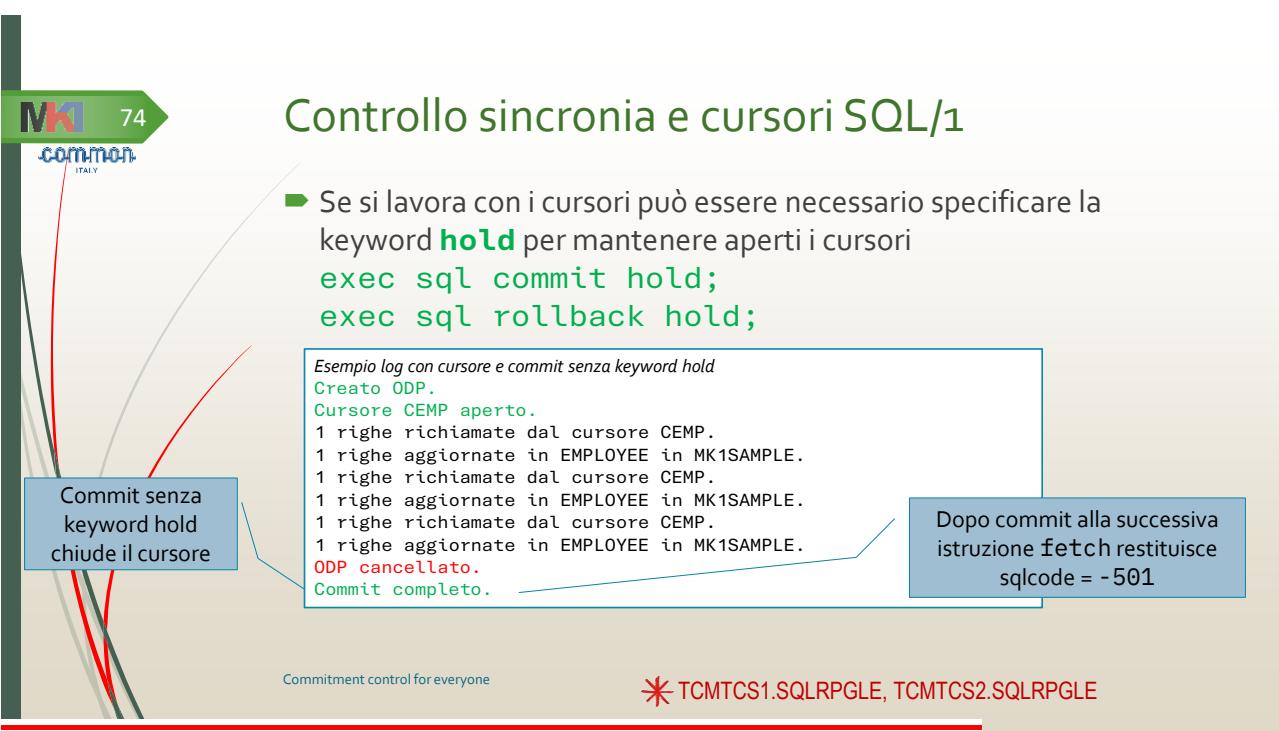
Concorrenza di accesso e cursori/3



72



73



74

MK1
common
ITALY

Controllo sincronia e cursori SQL/2

- ▶ durante il ciclo di lettura di un *cursor di sola lettura* l'istruzione **rollback hold** non modifica il posizionamento del cursore
- ▶ se il *cursor è di aggiornamento* un rollback eseguito durante il ciclo **riposiziona il cursore** al record sul quale si trovava a inizio del ciclo di commit, quindi dopo il rollback non è possibile eseguire un'istruzione **update/delete where current of...** e la successiva **fetch** riparte dal record letto all'inizio del ciclo di commit
- ▶ se il *cursor è di aggiornamento* un commit eseguito durante il ciclo consolida le modifiche pendenti e rilascia i lock dei record quindi non è possibile eseguire un'istruzione **update/delete where current of...**

Commitment control for everyone

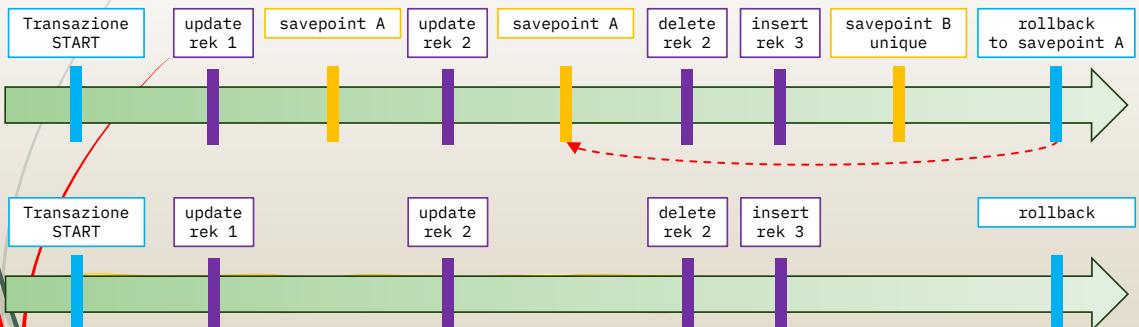
* TCSR BK1A.SQLRPGLE, TCSR BK1B.SQLRPGLE, TCSR BK1C.SQLRPGLE

75

MK1
common
ITALY

Savepoint/1

- ▶ L'istruzione **savepoint** consente di identificare uno o più punti all'interno di un ciclo di commit dove è possibile ritornare con un'istruzione di rollback



Commitment control for everyone

76

Savepoint/2

- ▶ `exec sql savepoint HERE
on rollback retain cursors`
p.es. il savepoint con nome HERE consente di poter eseguire un rollback a questo punto mantenendo aperto i cursori e mantenendo anche la posizione del cursore al momento in cui è stato salvato il savepoint
- ▶ si può tornare al savepoint HERE con l'istruzione SQL
`exec sql rollback to savepoint HERE`

<https://www.ibm.com/docs/en/i/7.6.0?topic=integrity-savepoints>

Commitment control for everyone

* TCSR BK3C.SQLRPGLE

77

Savepoint/3

- ▶ Impostare un savepoint
`exec sql savepoint nome [unique] [on rollback
retain cursors]`
- ▶ Eseguire un rollback ritornando ad un savepoint salvato
`exec sql rollback to savepoint nome`
- ▶ Eseguire un rollback ritornando all'ultimo savepoint
`exec sql rollback to savepoint`
- ▶ Rilasciare un savepoint
`exec sql release savepoint nome`
in ogni caso il savepoint viene rilasciato al termine della transazione

Commitment control for everyone

78

Gestione errori

- ▶ Codici di errore ([program status code](#))
 - ▶ 00802: Commitment control not active
 - ▶ 00803: Rollback operation failed
 - ▶ 00804: Error occurred on COMMIT operation
 - ▶ 00805: Error occurred on ROLBK operation
- ▶ SQL code
 - ▶ -774, -426, -30090: Invalid Transaction Termination
 - ▶ -7017: SQL statements cannot be executed under commitment control, because commitment control is already active to another relational database
 - ▶ -7007: COMMIT or ROLLBACK is not allowed, because commitment control has not been started.
 - ▶ -175: The commit operation failed, because a resource in the unit of work was not able to commit its resources.

Commitment control for everyone

79

Stato controllo sincronia

- ▶ Per visualizzare le informazioni sulla stato del controllo sincronia di un job è possibile
 - ▶ Menu gestione lavoro (Rich.sistema > opz. 3 Visual. lavoro corrente > opz. 16 Visualizzazione stato controllo convalida)
 - ▶ **WRKCMTDFN**: consente di visualizzare le informazioni del controllo di sincronia anche di altri lavori.
P.es. tutti i lavori con transazioni pendenti
WRKCMTDFN JOB(*ALL) STATUS(*PENDING)

```

Gestione definizioni di commit          Sistema: [REDACTED]
Immettere le opzioni e premere Invio.
  5=Visual. stato   12=Gestione lavoro   14=Commit forzato
  16=Rollback forzato ...
  ...

  Definiz.    N°      Utent.      Nome
  Opz.  commit    lav.    lav.    lav.  Sottopr.  Utente
  1     *DFTACTGRP  110229  [REDACTED]  OPRADEV000C *NONE
  19      110229  [REDACTED]  OPRADEV000C *NONE
  ...
  
```

Commitment control for everyone

80

MK1
common
ITALY

81

Stato controllo sincronia in ACS

Gestione schemi > Database > Nome_sistema > Transazioni > Transazioni database

N.B. occorre avere autorizzazione per il comando ADDPFTRG

Commitment control for everyone

Commitment control for everyone.sql

81

MK1
common
ITALY

82

Stato controllo sincronia

Informazioni visualizzate in base all'ambito della definizione del controllo di sincronia

| | commitment definition | activation group | job | thread |
|-------------------------------------|-----------------------|-----------------------|----------|--------|
| JOB-LEVEL | *JOB | == | nome job | *NONE |
| ACTIVATION-GROUP-LEVEL | nome activation group | nome activation group | nome job | *NONE |
| TRANSACTION-SCOPED | *TNSOBJ | == | nome job | thread |
| EXPLICITLY-NAMED¹ | nome definizione (Q*) | == | nome job | *NONE |

¹ possono essere avviati solo dal sistema operativo per gestire sue transazioni indipendenti dall'applicazione

Commitment control for everyone

82

Nome definizione controllo sincronia

| Activation group | Ambito | Nome |
|-------------------------------|------------------|-----------------------|
| qualsiasi | job | *JOB |
| default activation group | activation group | *DFTACTGRP |
| user-named activation group | activation group | nome activation group |
| system-named activation group | activation group | nome activation group |
| == | explicitly named | Q* |
| == | transaction | *TNSOBJ |

Commitment control for everyone

Identificatore del ciclo di commit

- ▶ Un ciclo di commit è il tempo che intercorre tra due confini di commit
- ▶ Il ciclo di commit è identificato dal **numero di sequenza della voce di giornale C SC** (start commit cycle)
- ▶ L'identificatore del ciclo di commit è **presente in ogni voce di giornale** scritta durante il ciclo di commit

Commitment control for everyone

ID transazione

- ▶ L'ID di una transazione o LUW è una stringa così composta
APPN.sysname.X' 48953C039E92 ' . 00001
- ▶ la porzione evidenziata in **azzurro** è diversa per ogni *definizione di controllo sincronia*
- ▶ all'interno della stessa definizione di controllo sincronia ad ogni commit/rollback la porzione finale evidenziata in **rosso** si incrementa identificando il *ciclo di commit*

Commitment control for everyone

85

Voci di giornale per controllo sincronia/1

Voci di giornale per cicli di controllo sincronia

- ▶ C BC: begin commitment control -> STRCMTCTL
- ▶ C **SC**: start commit cycle -> al primo record modificato
- ▶ C PC: Preparazione blocco di commit -> inizio commit implicito
- ▶ C **CM**: commit -> commit concluso con successo
- ▶ C R1: inizio del rollback
- ▶ C **RB**: rollback -> rollback concluso con successo
- ▶ C CN: fine del rollback
- ▶ C LW: fine transazione -> quando si conclude il commit/rollback
- ▶ C EC: fine commitment control -> ENDCMTCTL

Per le voci CM e RB viene indicato se il commit è implicito (contrassegno = 2) o esplicito (contrassegno = 0)

Commitment control for everyone

86

Voci di giornale per controllo sincronia/2

Voci di giornale per operazioni relative ai savepoint:

- ▶ C SB: start of savepoint -> creazione di un SQL savepoint
- ▶ C SQ: release of savepoint -> quando viene rilasciato un SQL savepoint
- ▶ C SU: rollback of savepoint -> quando viene eseguito un rollback di un savepoint

Non vengono scritte su giornale se la variabile di ambiente
QTN_JRNSAVPT_MYLIB_MYJRN = *NO
con MYLIB e MYJRN rispettivamente nome librerie e nome giornale

Commitment control for everyone

87

Voci di giornale per controllo sincronia/3

Voci di giornale per aggiornamenti dei record a seguito di un'operazione di rollback.

- ▶ R BR: immagine precedente del record aggiornato per rollback
- ▶ R DR: record cancellato per rollback
- ▶ R UR: immagine successiva del record aggiornato per rollback

Commitment control for everyone

88

89

Gestire i giornali con SQL

- ▶ Proprietà del giornale: [JOURNAL_INFO](#)
- ▶ Proprietà del ricevitore: [JOURNAL_RECEIVER_INFO](#)
- ▶ Pulizia ricevitori: [DELETE_OLD_JOURNAL_RECEIVERS](#)
- ▶ Tipi voci di giornale: [JOURNAL_CODE_INFO](#)
- ▶ Visualizzazione voci di giornale: [DISPLAY_JOURNAL](#)

[Creare SQL Views per interpretare
QSYS2.DISPLAY JOURNAL](#)
(FAQ400, 10-mar-2023, Roberto De Pedrini)
Procedura [Create_Display_Journal_Table_View](#)

[Tool EXPJRNE](#)
(Thomas Raddatz, 10-apr-2025)

Commitment control for everyone

Commitment control for everyone.sql

89

90

Perché lo si teme?

- perché che tutti i file siano salvati sullo stesso giornale. Quindi si teme per timore di decaimento di performance e di occupazione spazio disk.**
- prima di DB2 V era pernitoso l'uso o sincronia tra i file**
- prima di ILE non c'era il modo di sincronia agiva solo dell'activation**
- poiché gli oggetti sono registrati su giornale il rischio maggiore è attaccare nella manutenzione di oggetti e librerie**
- la necessità del disegno architettonico per definire correttamente le transazioni e gli activation**

DON'T BE AFRAID

Commitment control for everyone

90



91

A slide titled 'Glossario del controllo di sincronia' in green text. To the left of the title is a green arrow-shaped graphic containing the text 'MK1 common ITALY' and the number '92'. Below the title is a table with two columns: 'It' and 'En'. The table lists eight terms and their English equivalents. At the bottom center, the text 'Commitment control for everyone' is visible.

| It | En |
|------------------------------------|----------------------------|
| transazione | logical unit of work (LUW) |
| ciclo di commit | commit cycle |
| livello di isolamento | isolation level |
| confine della transazione | boundary |
| ambito | scope |
| definizione di controllo sincronia | commitment definition |
| commitment control | controllo di sincronia |

92

Bibliografia

- ▶ Database commitment control 7.6:
https://www.ibm.com/docs/en/ssw_ibm_i_76/pdf/rzakjpdf.pdf
- ▶ Documentazione IBM commitment control 7.6:
<https://www.ibm.com/docs/en/i/7.6.0?topic=database-commitment-control>
- ▶ Documentazione IBM SQL reference 7.6:
<https://www.ibm.com/docs/en/i/7.6.0?topic=reference-sql>
- ▶ Documentazione IBM SQL Programming 7.6 - Commitment control:
<https://www.ibm.com/docs/en/i/7.6.0?topic=integrity-commitment-control>
- ▶ Looking for Commitment part 1, di Paul Tuohy, 19-giu-2019,
<https://www.itjungle.com/2019/06/19/guru-classic-looking-for-commitment-part-1/>
- ▶ Looking for Commitment part 2, di Paul Tuohy, 17-lug-2019,
<https://www.itjungle.com/2009/03/18/fhg031809-story01/>
- ▶ Looking for Commitment part 3, di Paul Tuohy, 14-ago-2019,
<https://www.itjungle.com/2019/08/14/guru-classic-looking-for-commitment-part-3/>
- ▶ Soft Commit: Worth a Try on IBM i5/OS V5R4, di Hernando Bedoya, 31-lug-2006,
<https://www.redbooks.ibm.com/abstracts/tipso623.html#4>

Commitment control for everyone

Riferimenti



- ▶ E-mail aziendale: marco.riva@ivolution.it



- ▶ Blog: www.markonetools.it



- ▶ E-mail blog: info@markonetools.it



- ▶ Linkedin: www.linkedin.com/in/marcoriva-mk1



- ▶ Twitter: [@MarcoRiva73](https://twitter.com/@MarcoRiva73)



- ▶ Facebook: <https://www.facebook.com/markonetools/>



- ▶ YouTube: youtube.com/@markonetools

Commitment control for everyone


 MK1
common
ITALY

95

Disclaimer

DISCLAIMER 2022 MarkOneTools

Questa pubblicazione è fornita "as is" senza garanzia di alcun tipo.

Tutti i riferimenti in questo documento a siti web sono forniti solo per comodità e non possono essere considerati come approvazione di tali siti web. I materiali di tali siti web non fanno parte del materiale di questo documento e l'uso di tali siti web è a proprio rischio.

Tutti i marchi appartengono ai rispettivi proprietari.

Le informazioni contenute nella presente documentazione possono essere riservate e sono, comunque, destinate esclusivamente ai partecipanti al corso. I contenuti del documento sono stati redatti con la massima cura e diligenza, e sottoposti ad un accurato controllo. Si declina tuttavia ogni responsabilità, diretta e indiretta, nei confronti degli utenti e in generale di qualsiasi terzo, per eventuali imprecisioni, errori, omissioni derivanti dai suddetti contenuti. Nessuna parte di questa pubblicazione può essere riprodotta o trasmessa in qualsiasi forma o per qualsiasi scopo senza la preventiva ed espressa autorizzazione del proprietario.

Questo documento contiene esempi di programmi in linguaggio sorgente, i quali illustrano le tecniche di programmazione. E' possibile copiare, modificare e distribuire questi esempi in ogni forma allo scopo di sviluppo ed utilizzo. Questi esempi non sono stati sottoposti a un processo di test sotto tutte le condizioni. Perciò non si garantisce l'affidabilità e la funzionalità completa dei programmi di esempio.

Commitment control for everyone

95


 MK1
common
ITALY

96

Personal disclaimer

DISCLAIMER 2022 MarkOneTools

Ho cercato in tutti i modi di produrre una documentazione ben fatta, precisa ed affidabile.

Le informazioni tecniche derivano dallo studio dei manuali IBM, della letteratura disponibile nel mondo del web, dalla pratica ed esperienza personale, dalla tradizione orale e dal confronto con persone di ogni livello.

Le opinioni sono opinioni. Ognuno ha le proprie:

- le mie opinioni sono mie. Non dell'azienda per cui lavoro, non di IBM
- le mie opinioni non sono necessariamente corrette
- ma d'altronde nemmeno le tue lo sono

Commitment control for everyone

96