







Marco Riva

[www.markonetools.it](http://www.markonetools.it)

IBM CHAMPION

2021-2025

Privacy area









Ultimo aggiornamento: 18/02/2026

1







 <div>Critical</div>	 <div>Important</div>	 <div>Nice to have</div>
<div>Gruppi attivazione</div> <div>Ambito</div> <div>Attivazione</div> <div>Operazioni implicite</div>	<div>Livello isolamento</div> <div>Concorrenza accesso</div> <div>Monitoraggio</div>	<div>Deadlock</div> <div>Savepoint</div> <div>Soft commit</div> <div>Giornali</div>

Commitment control for everyone

2

MK1

common

ITALY

3

21/gen

1

4/feb

2

18/feb

3

4/mar

4

Sommario

Un ripasso di job e activation group

Concetti base del controllo di sincronia e glossario

Prerequisiti

Attivare il controllo di sincronia

Ambito del controllo di sincronia

Perché usare il controllo di sincronia

Livello di isolamento

Commit e rollback impliciti

Esempi di transazioni

Concorrenza di accesso

Cursori SQL vs commit/rollback

Savepoint

Monitoraggio del controllo sincronia

Controllo di sincronia e voci di giornale

Istruzioni SQL per i giornali

Commitment control for everyone

3

MK1

common

ITALY

4

Esempi e documentazione del corso

<https://github.com/mk1tools/Commitment-control-tutorial.git>

GitHub

MK1

Commitment control for everyone

4

MK

5

common  
ITALY

# Controllo di sincronia

Concetti base

Commitment control for everyone

5

MK

6

common  
ITALY

# Controllo sincronia (commitment control)

COSA

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

COME

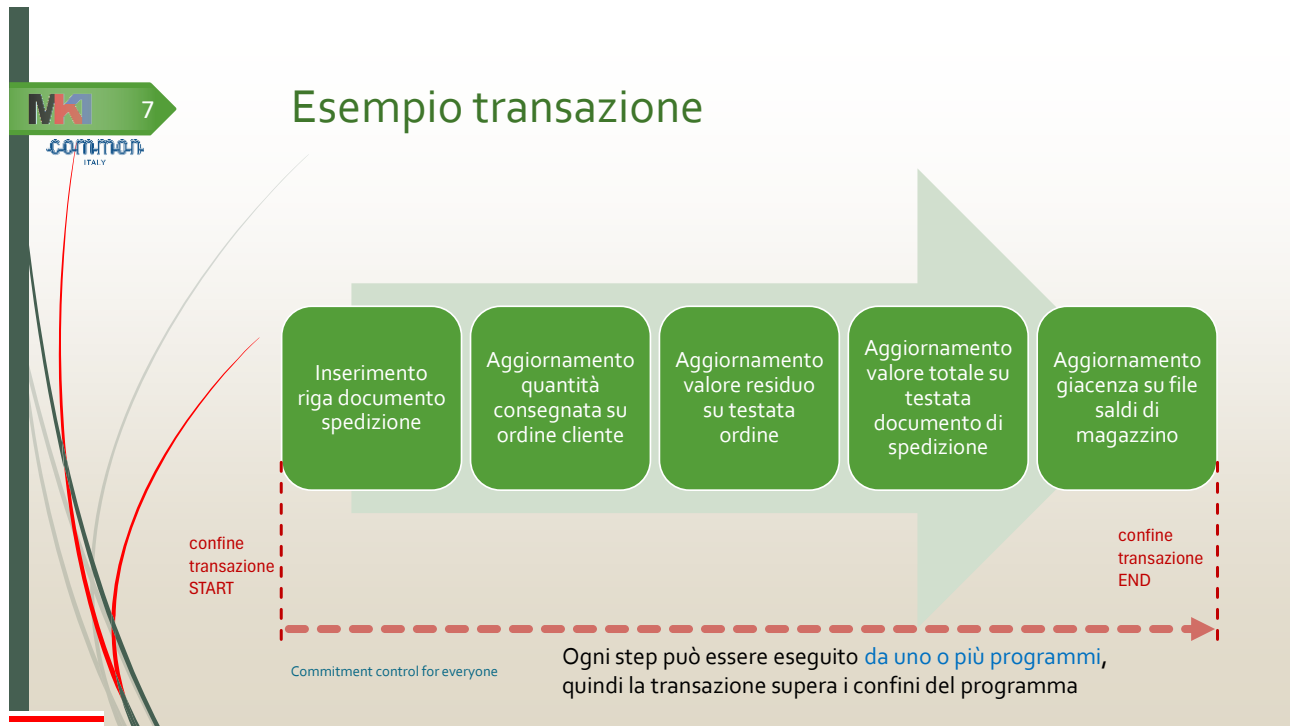
• **transazione**  
• è un gruppo di modifiche a uno o più file di database che dal punto di vista dell'utente appaiono come una singola modifica

PERCHÉ

• **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

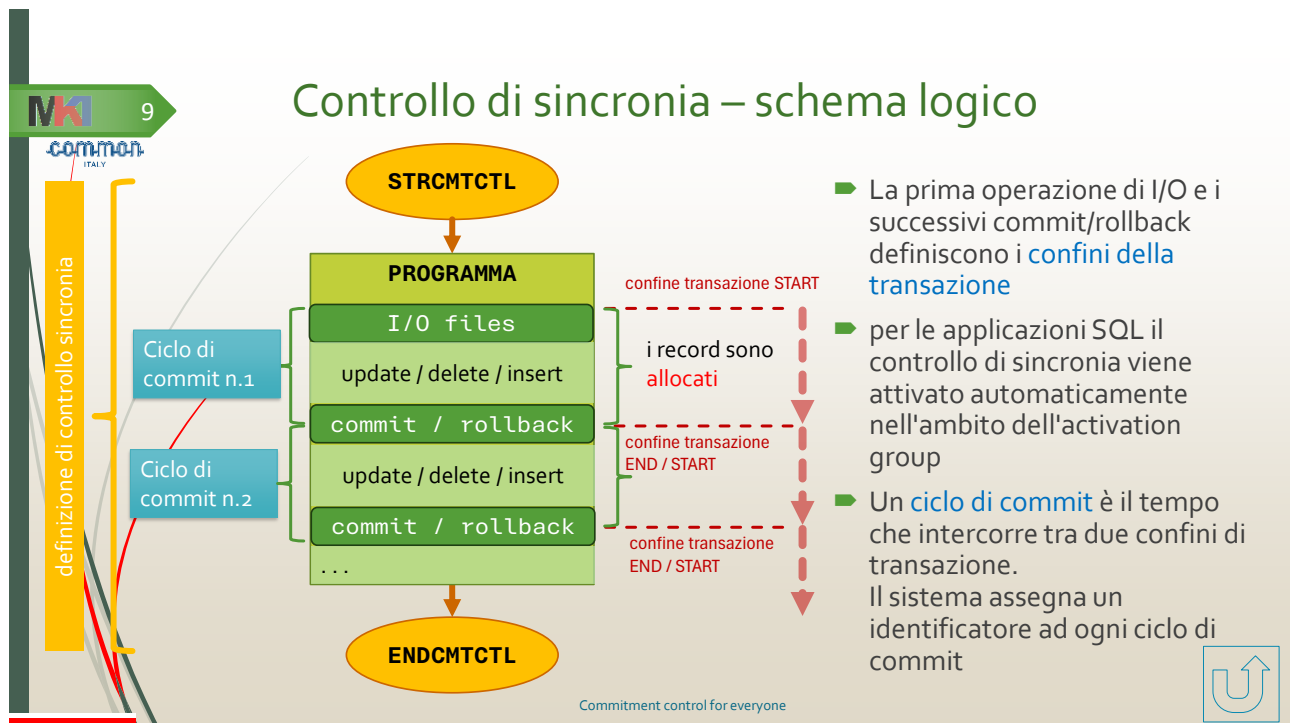
6



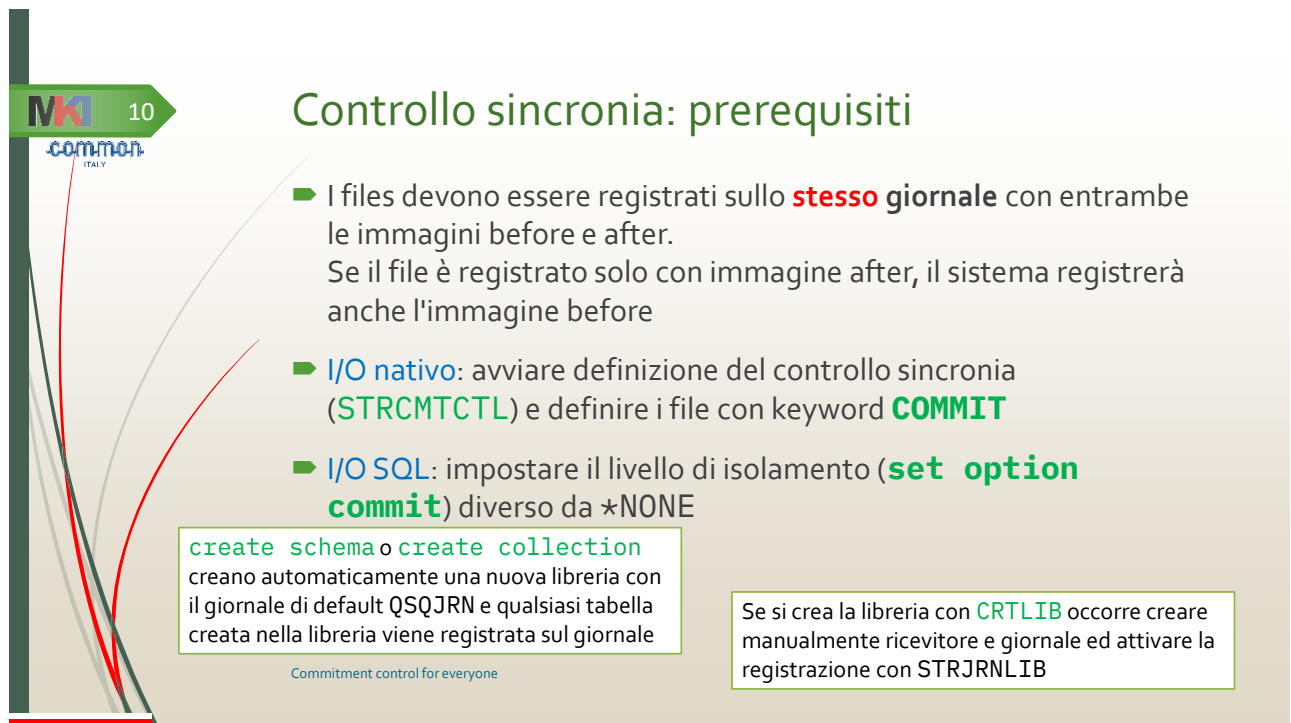
7



8



9



10



11

common  
ITALY

macchina

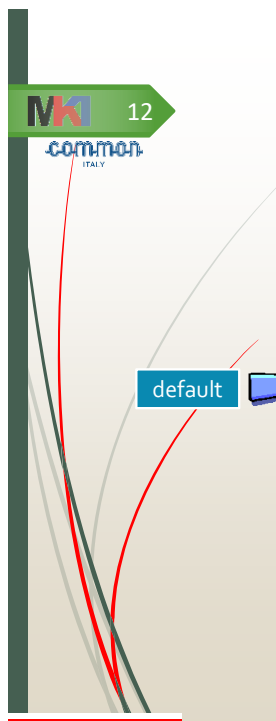
uomo

## 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



12


common  
ITALY

default

## 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

MK 13

common  
ITALY

## Job e activation group

Un breve ripasso

Commitment control for everyone

13

MK 14

common  
ITALY

## Job e activation group: program isolation

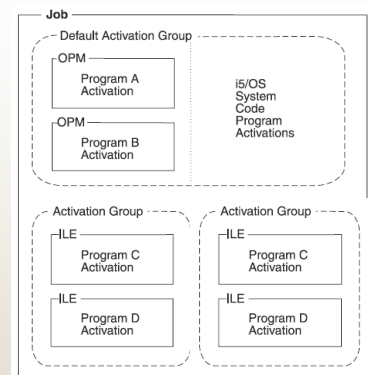
- L'**activation group** contiene le **risorse** necessarie ad eseguire il programma.

Le risorse sono:

- Static program variables
- Dynamic storage
- Temporary data management resources
- ODP (Open Data Path)
- **Commitment definitions**
- SQL cursors
- HFS (Hierarchical File System)



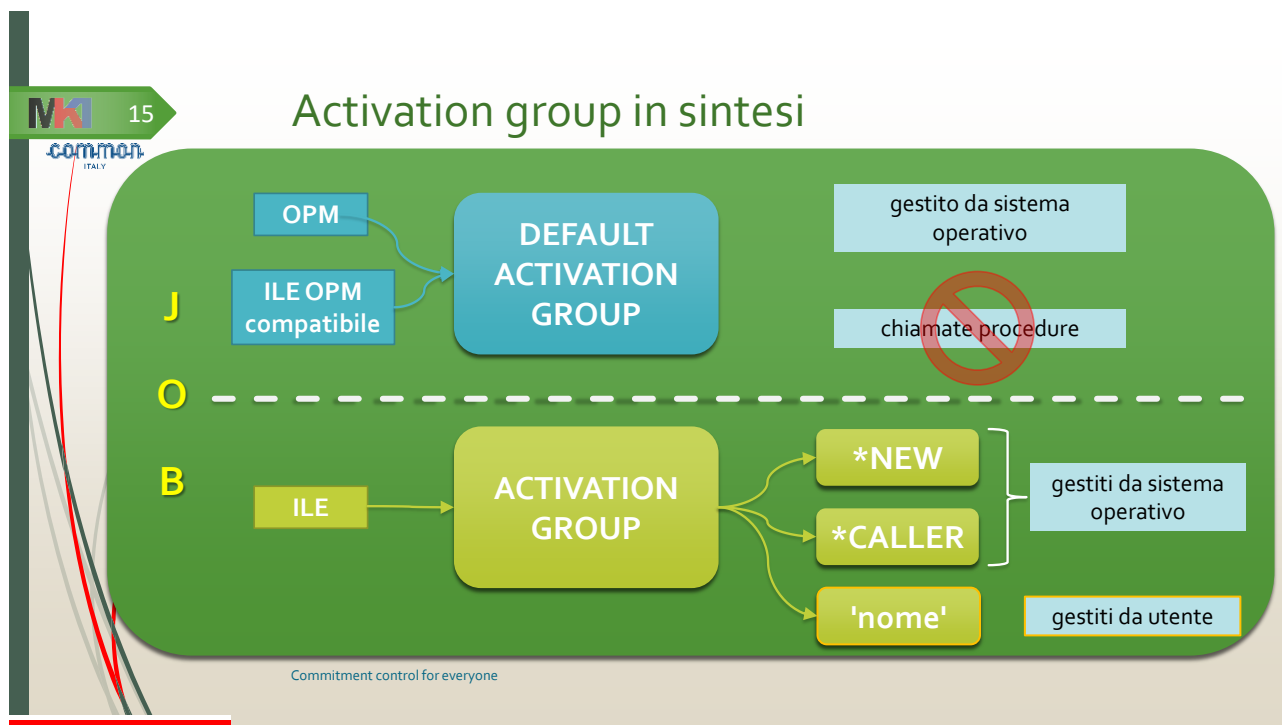
- All'avvio di un job, *automaticamente* vengono creati 2 activation group di **default** usati da tutti i programmi OPM e ILE OPM compatibile



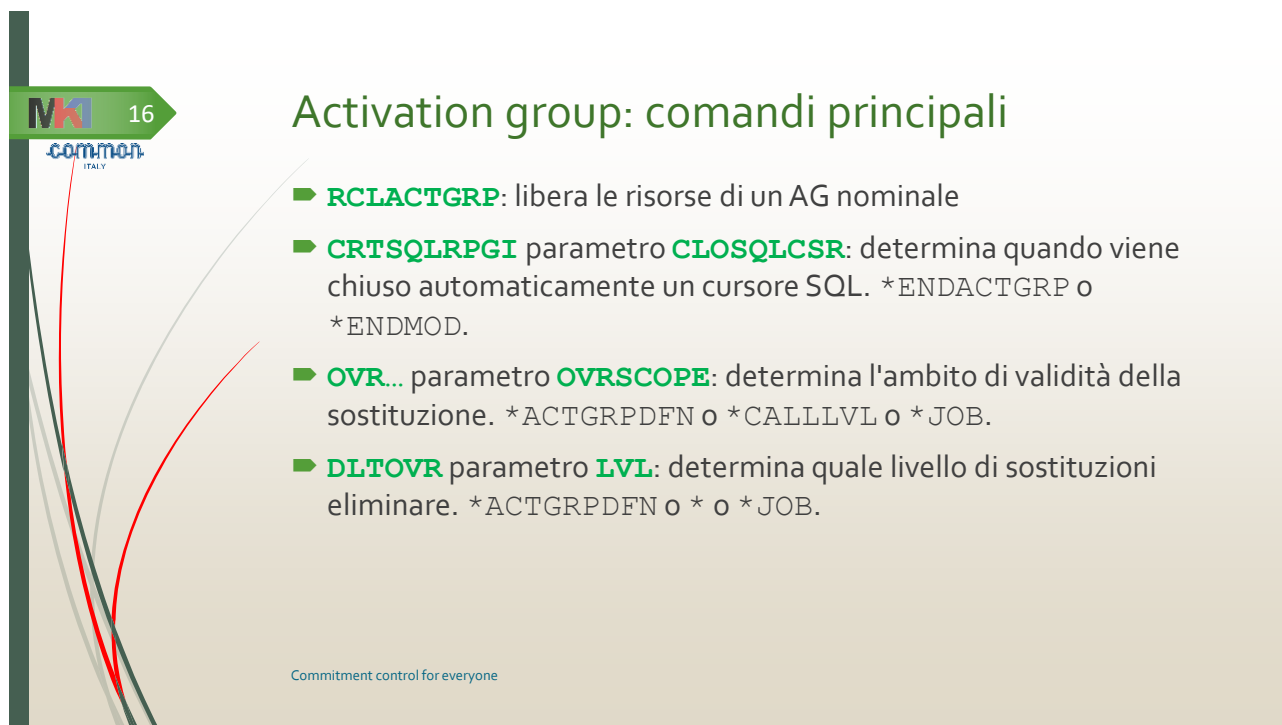
Fonte IBM

Commitment control for everyone

14



15



16



17

18



21

22

MK

23

common

ITALY

PGMC

DSPLY

Avvio programma PGMD. Invio per proseguire.

### Demo - CALL PGMC/3

WRKCMTDFN

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

Visualizzazione stack di chiamata

Lavoro: MRIVAA Utente: MRIVA Numero: 0000016A  
Sottoproc.: 0000016A

-----Gruppo attivazione-----

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
	PGMD MK1SQL	*DFTACTGRP	0000000000000002
	PGMD MK1SQL	*DFTACTGRP	0000000000000002

Commitment control for everyone

23

MK

24

common

ITALY

PGMC

DSPLY

Eseguito update 000210. Proseguire? (S/N)

### Demo - CALL PGMC/4

WRKCMTDFN

Visualizz. stato livello record

Lavoro: MRIVAA Utente: MRIVA Numero: 460667 Sistema: 460667  
Definizione commit . . . . . : \*JOB

-----Modifiche-----

File	Libreria	Membro	Commit	Rollback	Sospeso
EMPLOYEE	MK1SAMPLE	EMPLOYEE	0	0	1
EMPLOYEE	EMPLOYEE	EMPLOYEE	0	0	1

Id ciclo commit

File	Libreria	Membro	Id ciclo commit
EMPLOYEE	MK1SAMPLE	EMPLOYEE	6727
EMPLOYEE	EMPLOYEE	EMPLOYEE	6727

Opz Giornale Libr. Id ciclo commit Blocchi di record Modifiche in sospeso

QSQRN MK1SAMPLE 6727 2 2

Commitment control for everyone

24

MK1

25

common

ITALY

PGMA

DSPLY Avvio programma PGMA. Invio per proseguire.

Visualizzazione stack di chiamata

Lavoro: MRIVA Utente: MRIVA Numero: 0000016A

Sottoproc.: 0000016A

-----Gruppo attivazione-----

Tipo	Programma	Nome	Numero
	QCMD	QSYS	*DFTACTGRP 0000000000000001
	QUICMENU	QSYS	*DFTACTGRP 0000000000000001
1	QUIMNDRV	QSYS	*DFTACTGRP 0000000000000001
2	QUINGFLW	QSYS	*DFTACTGRP 0000000000000001
3	QUICMD	QSYS	*DFTACTGRP 0000000000000001
	PGMC	MKISQL	*DFTACTGRP 0000000000000002
	PGMC	MKISQL	*DFTACTGRP 0000000000000002
	PGMD	MKISQL	*DFTACTGRP 0000000000000002
	PGMD	MKISQL	*DFTACTGRP 0000000000000002
	PGMA	MKISQL	*NEW 00000000000000013
	PGMA	MKISQL	*NEW 00000000000000013

Visualizzazione gruppo attivazione

Lavoro: MRIVA Utente: MRIVA Numero: 460667

Sistema: 460667

-----Gruppo attivazione-----

Nome	Numero	Indicatore in uso	Programma	Libreria
*DFTACTGRP	0000000000000001	SI		
*DFTACTGRP	0000000000000002	SI		
QLGLOCAL	0000000000000011	Senza	QLGLOCAL	QSYS
QSQCCLI	0000000000000012	Senza	QSQCCLI	QSYS
19	0000000000000013	SI	PGMA	MKISQL

-----Modifiche-----

File	Libreria	Membro	Commit	Rollback	Sospeso
EMPLOYEE	MKISAMPLE	EMPLOYEE	0	0	1
EMPLOYEE		EMPLOYEE	0	0	1
EMPLOYEE		EMPLOYEE	0	0	1

-----Id ciclo commit-----

File	Libreria	Membro	Id ciclo	commit
EMPLOYEE	MKISAMPLE	EMPLOYEE	6727	6727
EMPLOYEE		EMPLOYEE	6727	6727

Visualizzazione gruppo attivazione

Lavoro: MRIVA Utente: MRIVA Numero: 460667

Sistema: 460667

-----Gruppo attivazione-----

Nome	Numero	Indicatore in uso	Programma	Libreria
*DFTACTGRP	0000000000000001	SI		
*DFTACTGRP	0000000000000002	SI		
QLGLOCAL	0000000000000011	Senza	QLGLOCAL	QSYS
QSQCCLI	0000000000000012	Senza	QSQCCLI	QSYS
19	0000000000000013	SI	PGMA	MKISQL

Commitment control for everyone

25

MK1

26

common

ITALY

PGMB

DSPLY Avviato contr.sincr. \*ACTGRP. Invio per proseguire

Visualizzazione stack di chiamata

Lavoro: MRIVA Utente: MRIVA Numero: 0000016A

Sottoproc.: 0000016A

-----Gruppo attivazione-----

Tipo	Programma	Nome	Numero
	QCMD	QSYS	*DFTACTGRP 0000000000000001
	QUICMENU	QSYS	*DFTACTGRP 0000000000000001
1	QUIMNDRV	QSYS	*DFTACTGRP 0000000000000001
2	QUINGFLW	QSYS	*DFTACTGRP 0000000000000001
3	QUICMD	QSYS	*DFTACTGRP 0000000000000001
	PGMC	MKISQL	*DFTACTGRP 0000000000000002
	PGMC	MKISQL	*DFTACTGRP 0000000000000002
	PGMB	MKISQL	*NEW 0000000000000014
	PGMB	MKISQL	*NEW 0000000000000014

Visualizzazione gruppo attivazione

Lavoro: MRIVA Utente: MRIVA Numero: 460667

Sistema: 460667

-----Gruppo attivazione-----

Nome	Numero	Indicatore in uso	Programma	Libreria
*DFTACTGRP	0000000000000001	SI		
*DFTACTGRP	0000000000000002	SI		
QLGLOCAL	0000000000000011	Senza	QLGLOCAL	QSYS
QSQCCLI	0000000000000012	Senza	QSQCCLI	QSYS
20	0000000000000014	SI	PGMB	MKISQL

Visualizzazione stato definizione commit

Lavoro: MRIVA Utente: MRIVA Numero: 460667

Sistema: 460667

20/01/26 15:22:52

Sottoproc.: 0000016A

-----Stato definizione commit-----

Sottoproc.	0000016A	*NONE
ID LUM (Logical Unit of Work)	APPN.0000016A.X'BD21C5AF077B'.00001	
ID spazio blocco	UDB_01000000000409F5	
Definizione commit	20	
Gruppo attivazione	20	
Gruppo ASP	*SYSBAS	
Ubicazione risorsa	NESSUNO	
Livello blocco predefinito	*CHG	
Utente	MRIVA	
Modifiche in sospeso locali	NO	
Ruolo	RIPRISTINO	
Stato	RIPRISTINO	
Registrazione data/ora		

Risincronizz. in corso

Operaz. euristica

Lavoro attivo

Lavoro server

Ambito blocco

Supero tempo transazione

Attesa massima blocco

-----Id ciclo commit-----

File	Libreria	Membro	Id ciclo	commit
EMPLOYEE	MKISAMPLE	EMPLOYEE	6727	6727
EMPLOYEE		EMPLOYEE	6727	6727

ACTGRP

26

WRKCMTDFN


DSPLY Eseguito update 000240. Proseguire? (S/N)

Opz	Definiz. commit	N° lav.	Risc. in corso	Modif. locali sospese
*JOB		460667	NO	SI
20		460667	NO	SI

```
Visualizz. stato livello record
Lavoro: 468667 Utente: 468667 Sistema: 468667
Definizione commit . . . . . : 20
-----Modifiche-----
File      Libreria  Membro  Commit  Rollback  Sospeso
EMPLOYEE  MKISAMPLE EMPLOYEE  0        0          1
```

```
Definizione commit . . . . . : 20
```

File	Libreria	Membro	Id ciclo commit
EMPLOYEE	MK1SAMPLE	EMPLOYEE	6735

Opz	Giornale	Libr.	Id ciclo commit	Blocchi di record	Modifiche in sospeso
	QSQJRN	MK1SAMPLE	6735	1	1

Sequenza	Cod.	Tipo	Oggetto	Libreria	Lavoro	Ora
6735	C	SC	start ciclo di commit		MS12A	15:25:09
6736	R	UB	EMPLOYEE	MK1SAMPLE	MS12A	15:25:09
6737	R	UP	EMPLOYEE	MK1SAMPLE	MS12A	15:25:09

## Commitment control for everyone

27

WRKCM TDFN

DSPLY PGMB ha eseguito commit. Invio per proseguire

Opz	Definiz.	N°	Risinc.	Modif.
	commit	lav.	in	locali
			corso	sospese
	*JOB	460667	NO	SI
	20	460667	NO	NO

File	Libreria	Membro	Commit	Rollback	Sospeso
EMPLOYEE	MK1SAMPLE	EMPLOYEE	1	0	0

```
Visualizzazione stato definizione commit
```

Lavoro:	TESTA	Utente:	TESTA	Numero:	406667	20/01/26	15:31:25
Sottopr.	:	:	:	:	:	:	: *NONE
ID LUW (Logical Unit of Work)	:	:	:	:	:	:	: APPN, 'X'BD21C5AF077B',00002
ID spazio blocco	:	:	:	:	:	:	: UDB_01000000000409F5
Definizione commit	:	:	:	:	:	:	: 20
Gruppo attivazione	:	:	:	:	:	:	: 20
Gruppo ASP	:	:	:	:	:	:	: *SYSBAS
Limite blocco	:	:	:	:	:	:	: 500000000
Commit durevole	:	:	:	:	:	:	: SI
Numero di commit	:	:	:	:	:	:	: 1
Numero di rollback	:	:	:	:	:	:	: 0
Giornale predefinito	:	:	:	:	:	:	:
Libreria	:	:	:	:	:	:	:

## Commitment control for everyone

28

29

common

PGMB

DSPLY PGMB arresterà il contr.sincr. Invio per proseguire

DSPLY Arresto controllo di sincronia con errori

ENDCMCTL 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

Demo - CALL PGMC/9

WRKCMTDFN

Visualizz. stato livello record

Lavoro: MK1A Utente: MK1A Numero: 460667 Sistema: AS1PRD

Definizione commit : 20

File	Libreria	Membro	Liv blocco	Stato	Risoluzione accessi contemp.
EMPLOYEE	MK1SAMPLE	EMPLOYEE	*CHG	APERTO	*WAIT

Visualizz. stato livello record

Lavoro: MK1A Utente: MK1A Numero: 460667 Sistema: AS1PRD

Definizione commit : 20

File	Libreria	Membro	Liv blocco	Stato	Risoluzione accessi contemp.

(Non ci sono modifiche a livello record sotto controllo di commit.)

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

29

30

common

PGMC

DSPLY PGMC arresterà il contr.sincr. Invio per proseguire

DSPLY Arresto controllo di sincronia con errori

Chiusura PGMC

SIGNOFF

Demo - CALL PGMC/10

WRKCMTDFN

Visualizz. stato livello record

Lavoro: MK1A Utente: MK1A Numero: 460667 Sistema: AS1PRD

Definizione commit : \*JOB

File	Libreria	Membro	Liv blocco	Stato	Risoluzione accessi contemp.
EMPLOYEE	MK1SAMPLE	EMPLOYEE	*CHG	APERTO CHIUSO	*WAIT

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

Definiz. Opz	commit	N° lav.	Uten. lav.	Nome lav.	Sottopr.	Utente

(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.

Registrazione lavoro

Nome lavoro : MK1A Utente : MRIVA Numero : 20/01/26 15:41:04 CET Pag. 14

Descrizione lavoro : QDFTJOB00 Libreria : QGPL

IDMSG TIPO GRAV DATA ORA DA PGH LIBRERIA INST A PGH 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 LUM (logical unit of work/unità logica di lavoro) APPN.AS1PRD.X'BCFEBB5E077B'.00001.

30

31

common

ITALY

31

Comm.It. quiz

J

O

B

DEFAULT ACTIVATION GROUP

STRCMTCTL \*JOB

PGMG

STRCMTCTL \*ACTGRP

PGMH

\* PGMG.RPGLE, PGMH.RPGLE

QUIZ TIME?

3

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

YES

NO

↑

31

32

common

ITALY

32

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

32



MK1

33

common  
ITALY

## 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

MK1

34

common  
ITALY

## 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. Alloca 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.o?topic=concepts-isolation-level>

34

MK1

35

common

ITALY

## 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.o?topic=level-comparison-isolation-levels>

Commitment control for everyone

35

MK1

36

common

ITALY

## Livello di allocazione

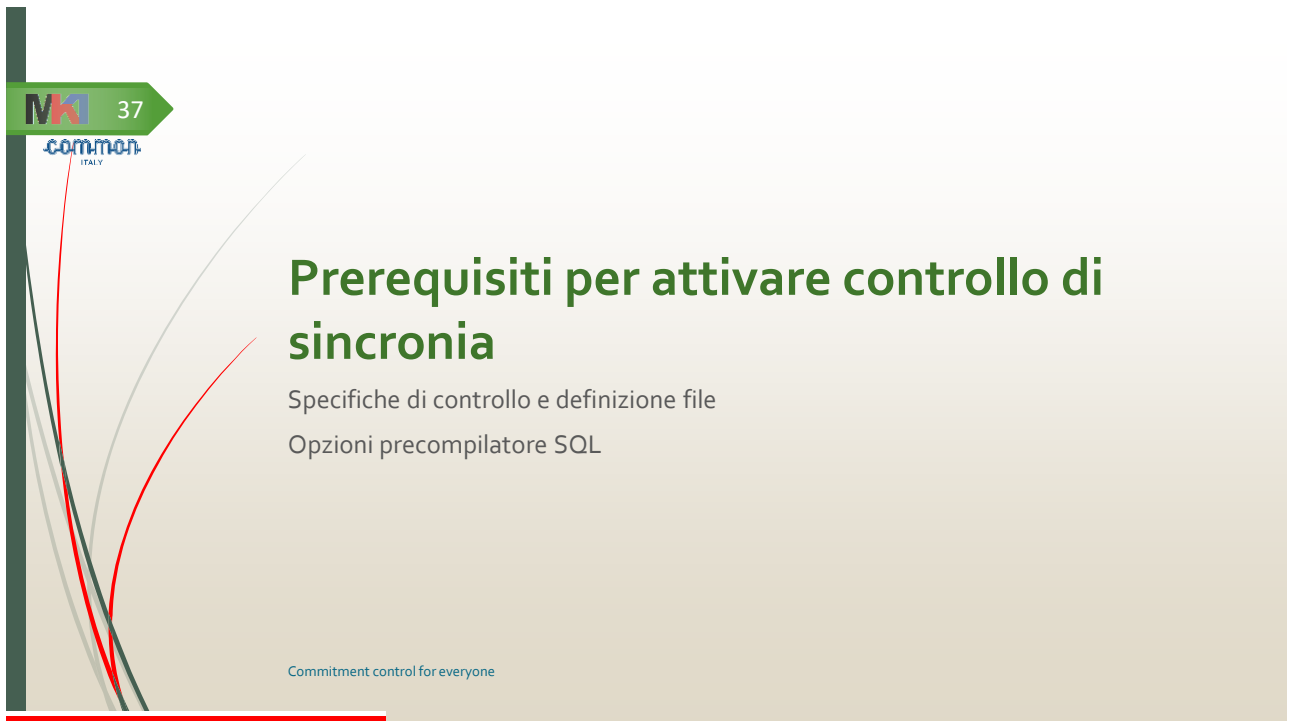
- Il parametro **LCKLVL** in STRCMTCTL determina il livello di allocazione dei record di default

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	

<https://www.ibm.com/docs/en/i/7.6.o?topic=control-commit-lock-level>

Commitment control for everyone

36

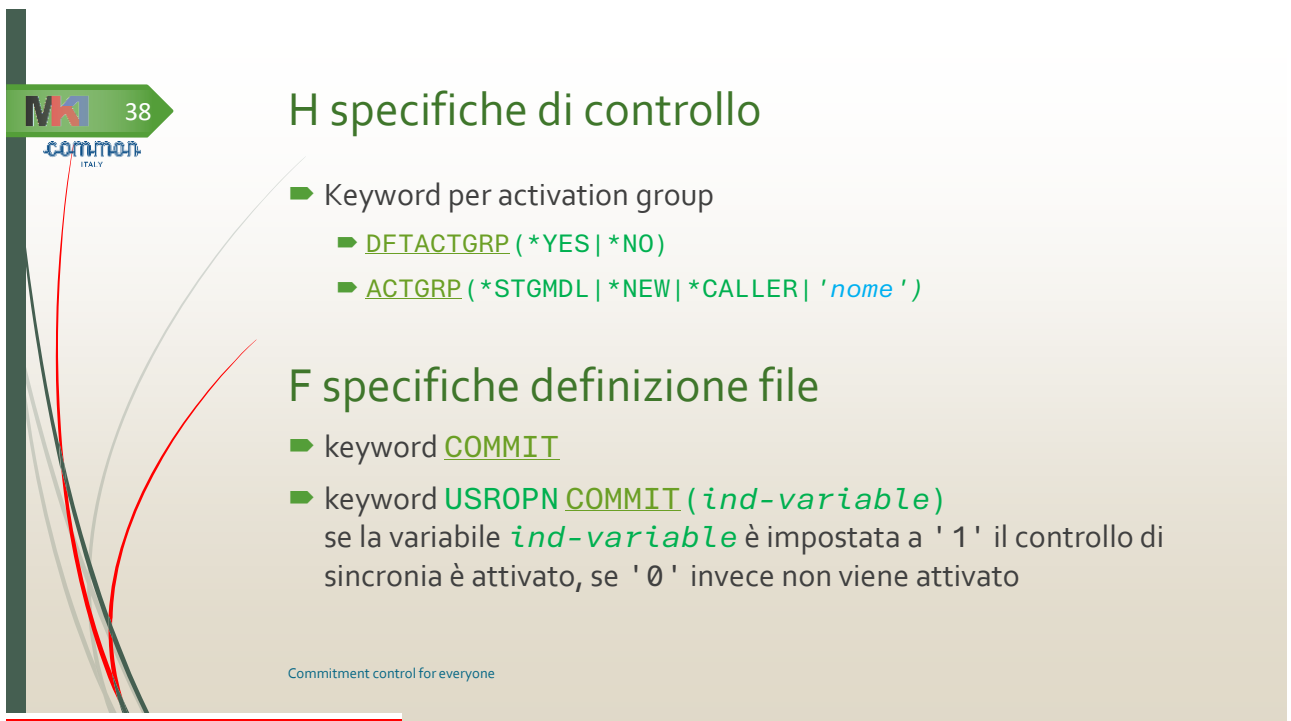


## Prerequisiti per attivare controllo di sincronia

- 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


 39  
common  
ITALY

## 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

39

 40  
common  
ITALY


## 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.o?topic=statement-isolation-clause>

40


 41  
common  
ITALY

## 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

 42  
common  
ITALY

## 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

MK

43

common

ITALY

# Commit e rollback

Consolidare o annullare una transazione

Commitment control for everyone

43

MK

44

common

ITALY

# 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

CALL PGMA

update rek 1

CALL PGMB

update rek 2

commit

rollback

QUIZ TIME?

4

Quanti record vengono consolidati?

0

1

2

↑

44

MK

45

common

ITALY

### 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 sospenso
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

LIVE DEMO

45

MK

46

common

ITALY

### Comm.It. 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

CALL PGMA

update rek 1

CALL PGMB

update rek 2

commit

rollback

QUIZ TIME?

5

Quanti record vengono consolidati?

0

1

2

↑

46

MarkOneTools - MK1

47

common

PGMA

dft

PGMB

new

Esempio transazioni/2

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 sospenso	
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	si chiude automaticamente act group 19 perché gestito dal sistema			
PGMA	rollback	*DFTACTGRP	APPN.sysname.X'5A75B9A8C288'.00002	0	

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

\* TCMT2A.SQLRPGLE, TCMT2B.SQLRPGLE

LIVE DEMO

47

48

common

Comm.It. 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

CALL PGMA

update rek 1

CALL PGMB

update rek 2

rollback

QUIZ TIME?

6

Quanti record vengono consolidati?

0

1

2

Chi ha consolidato il record n. 2?

48

MarkOneTools - MK1



49

common

PGMA

dft

PGMB

new

Esempio transazioni/3

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 sospenso
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	si chiude act group 19 e viene eseguito un <b>commit implicito</b>		
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

LIVE DEMO

49

50

common

PGMA

dft

PGMB

name

Esempio transazioni/4

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 sospenso
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	non si chiude act group PGMB perché non gestito dal sistema e la transazione rimane sospesa		
PGMA	rollback	*DFTACTGRP	APPN.sysname.X'5A75B9A8C288'.00002	0

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

Commitment control for everyone

\* TCMT4A.SQLRPGLE, TCMT4B.SQLRPGLE

50

51

common

PGMA

dft

PGMB

new

Esempio transazioni/5

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 sospenso
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 <b>rollback implicito</b>		
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

52

common

Comm.It. 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)

```
sequenceDiagram
    participant PGMA
    participant PGMB
    PGMA->>PGMA: CALL PGMA
    PGMA->>PGMA: update rek 1
    PGMA->>PGMB: CALL PGMB
    PGMB->>PGMB: update rek 2
    PGMB-->>PGMA: 
    PGMA->>PGMA: signoff
```

QUIZ TIME?

7

Quanti record vengono consolidati?

0

1

2

↑

52

53

common

PGMA  
dft

PGMB  
new

Esempio transazioni/6

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 sospenso
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	si chiude act group 19 e viene eseguito un <b>commit implicito</b>		
PGMA	return	transazione rimane sospesa con il rek aggiornato da PGMA ancora allocato		
==	endjob	si chiude il job e viene eseguito un <b>rollback implicito</b>		

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

\* TCMT6A.SQLRPGLE, TCMT6B.SQLRPGLE

LIVE DEMO

53

54

common

PGMA -> PGMB  
new - caller

Esempio transazioni/7

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 sospenso
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	NON si chiude act group 19 e quindi non viene eseguita nessuna azione implicita		
PGMA	commit	act group 19	APPN.sysname.X'48953C039E92'.00002	0
PGMA	return	si chiude act group 19 e viene eseguito un <b>commit implicito</b> anche se non eseguito esplicitamente da pgmA		

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

\* TCMT7A.SQLRPGLE, TCMT7B.SQLRPGLE

54

55

common

Comm.It. 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

CALL PGMA

CALL PGMB

update rek 1

update rek 1

errore

commit

signoff

QUIZ TIME?

8

Quanti record vengono consolidati?

0

1

55

56

common

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 implicito		

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

57

common

PGMA  
new

PGMB  
new

### Esempio transazioni/9

- 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 sospenso
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

58

common

U

### Commit o rollback implicito

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

<https://www.ibm.com/docs/en/i/7.6.o?topic=control-system-initiated-end-commitment>

Commitment control for everyone

58


 59  
common  
ITALY

## Concorrenza di accesso

Livello di isolamento delle transazioni

Commitment control for everyone

59

 60  
common  
ITALY

## 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



Programma che apre una transazione per aggiornare dei record

```
dc1-s Domanda char(51);
dc1-s Risposta char(1);

exec sql
  set option COMMIT =
    *CHG;
    *CS;
    *CS, CONACC = *CURCMT;
    *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

```
dc1-s Domanda char(51);
dc1-s Risposta char(1);
dc1-s wBonus packed(9:2);

exec sql
  set option COMMIT =
    *CHG;
    *CS;
    *CS, CONACC = *CURCMT;
    *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

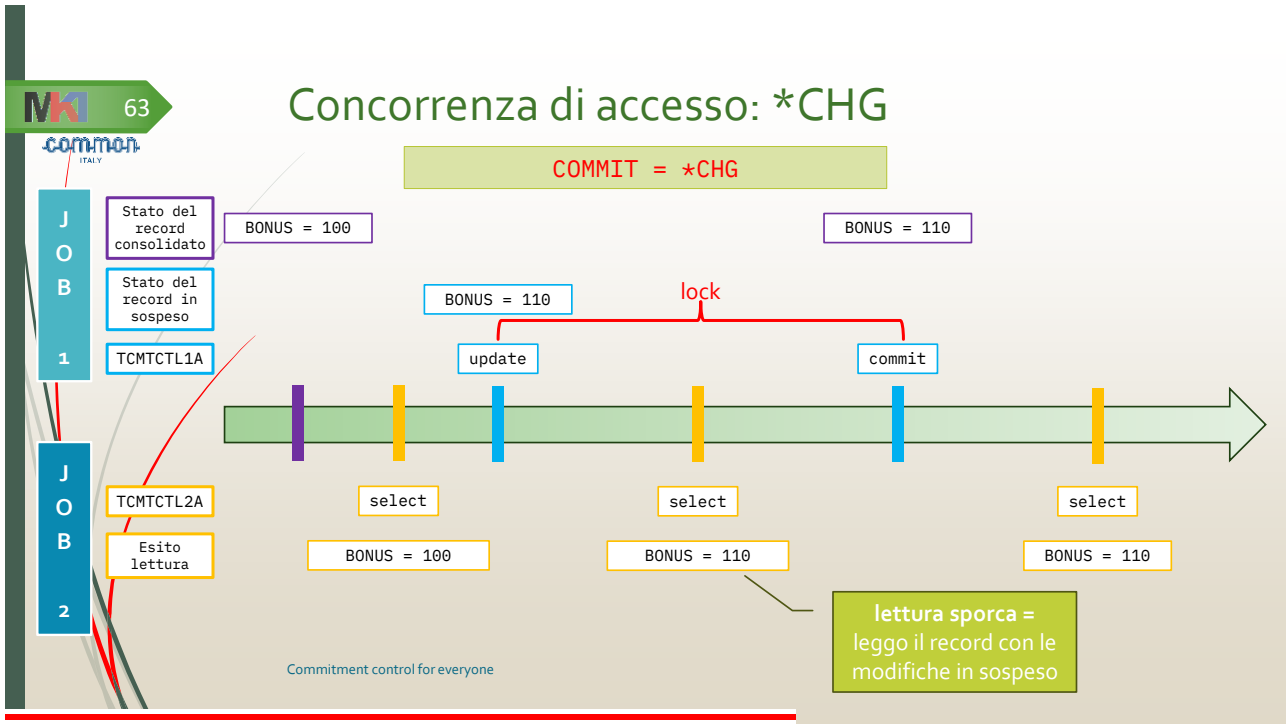
## \*CHG vs \*CS



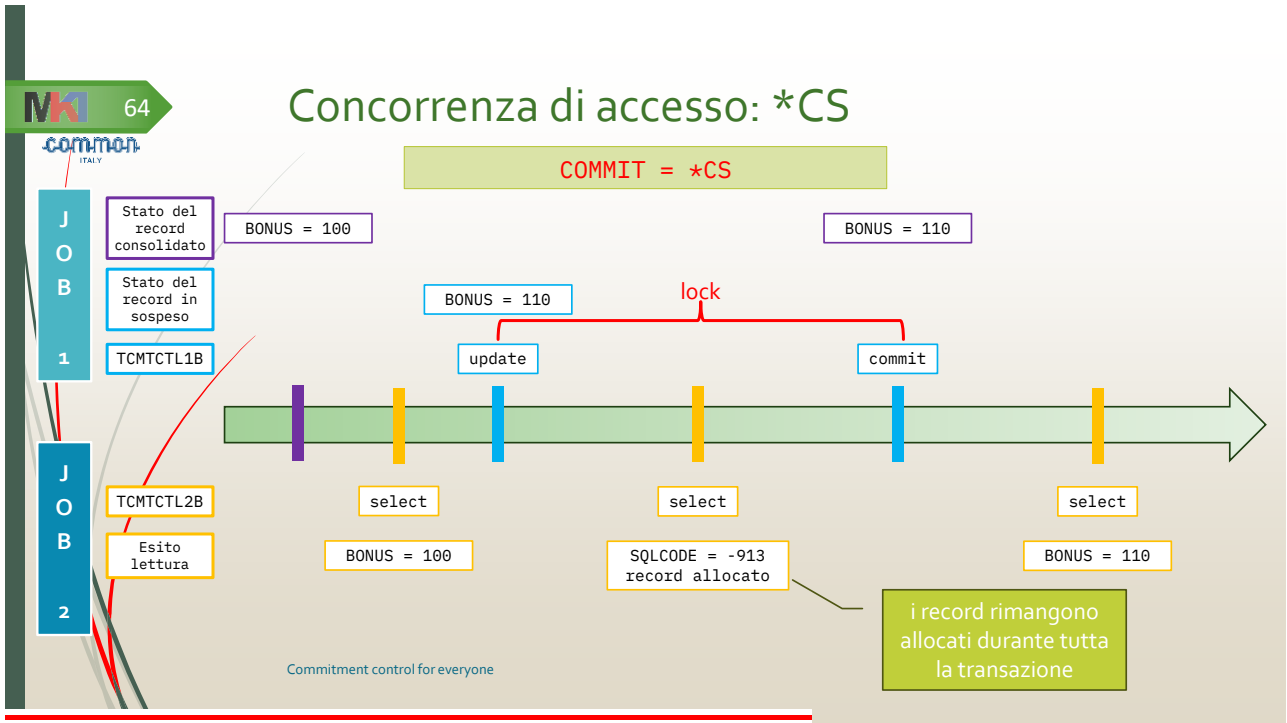
Livello	Descrizione	Attivo	Letture sporche	Record fantasma	Allocazione record letti	Note
*CS/CS	LETTURA SINCRONIZZATA (Cursor Stability or Read Committed)	✓	✗	✓	✗	Garantisce che ogni record modificato in un altro gruppo di attivazione non può essere letto fino a quando viene consolidato → come RR o RS 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



**MK** 65  
common ITALY

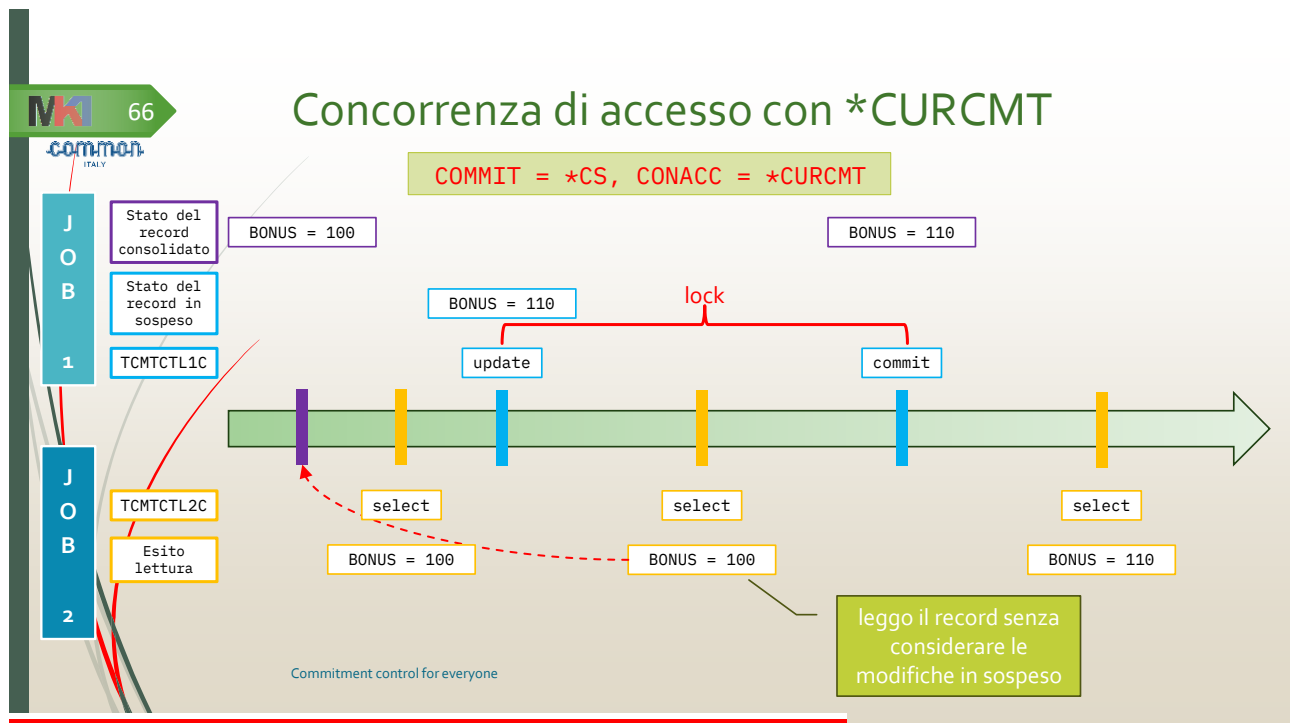
## Concurrent access resolution

➤ 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



66

MK1

67

common

ITALY

\*CHG vs \*ALL

Livello	Descrizione	Attivo	Letture 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

68

common

ITALY

Concorrenza di accesso: \*ALL/1

COMMIT = \*ALL

J  
O  
B  
1

Stato del record consolidato  
BONUS = 100

Stato del record in sospeso  
TCMTCTL1D

J  
O  
B  
2

TCMTCTL2D

Esito lettura

select  
BONUS = 100

update  
SQLCODE = -913  
record allocato

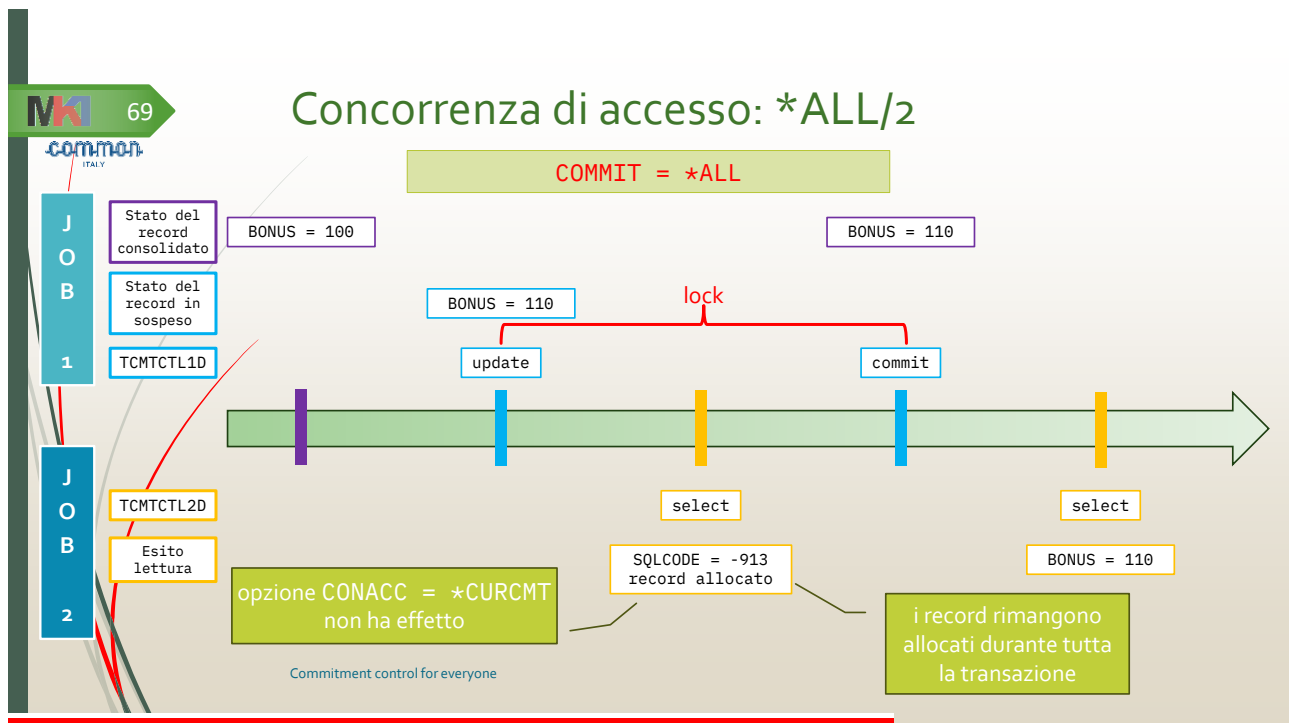
chiusura transazione

lock

i record rimangono allocati anche per sole operazioni di lettura

Commitment control for everyone

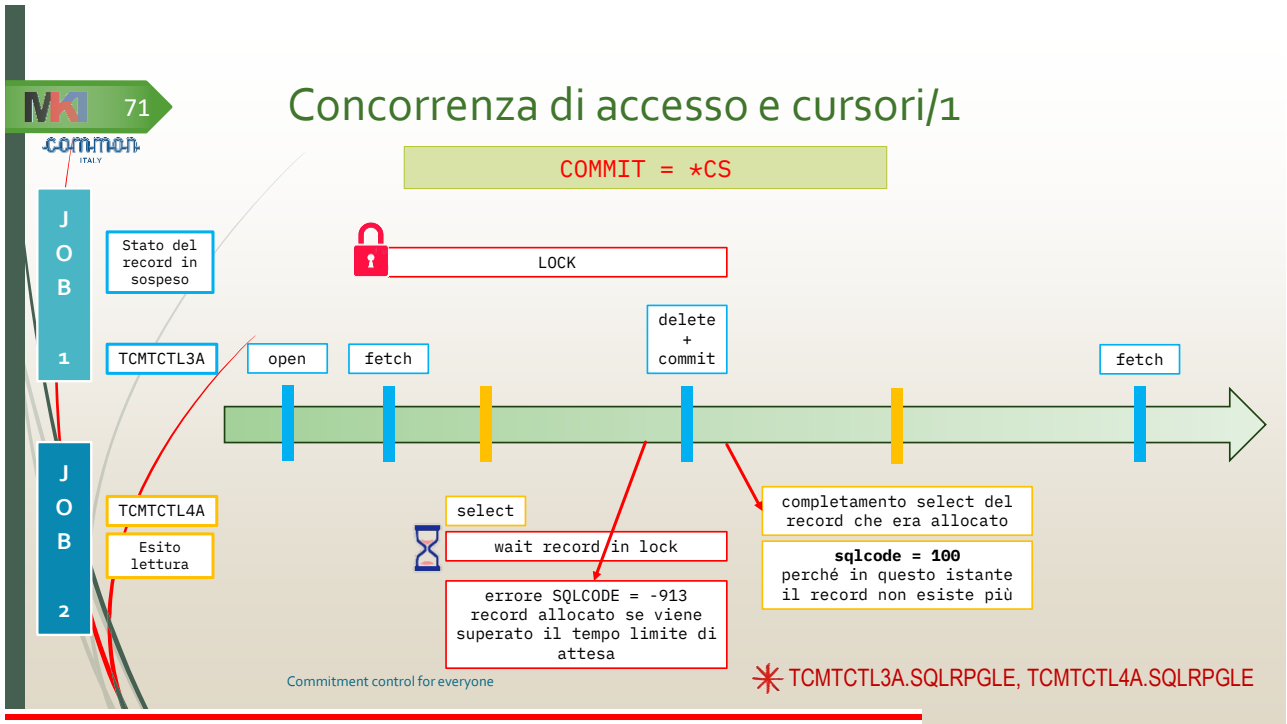
68



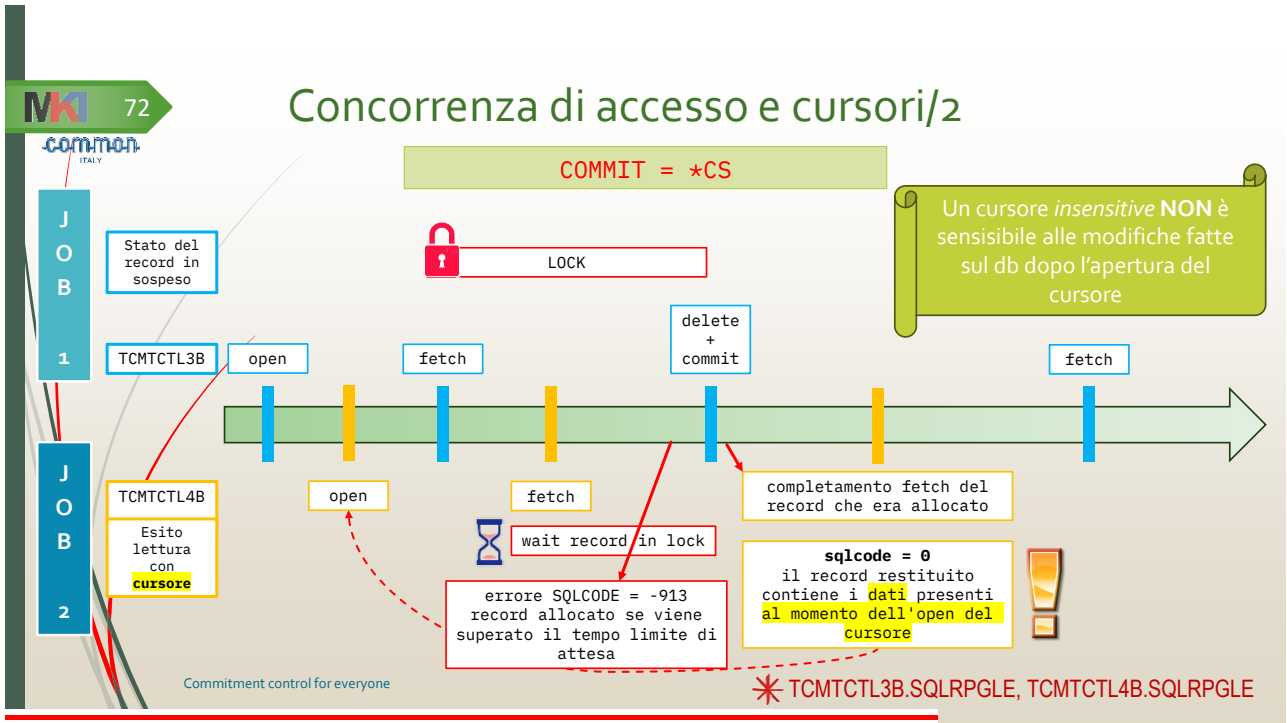
69



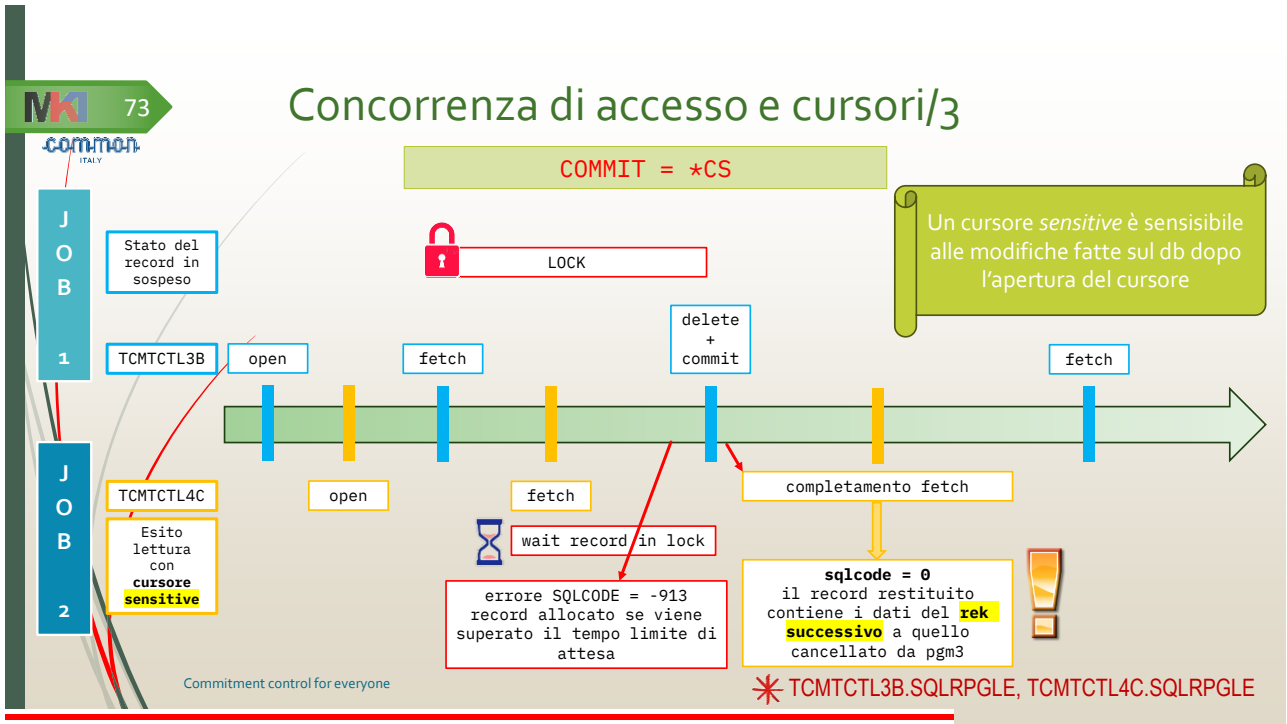
70



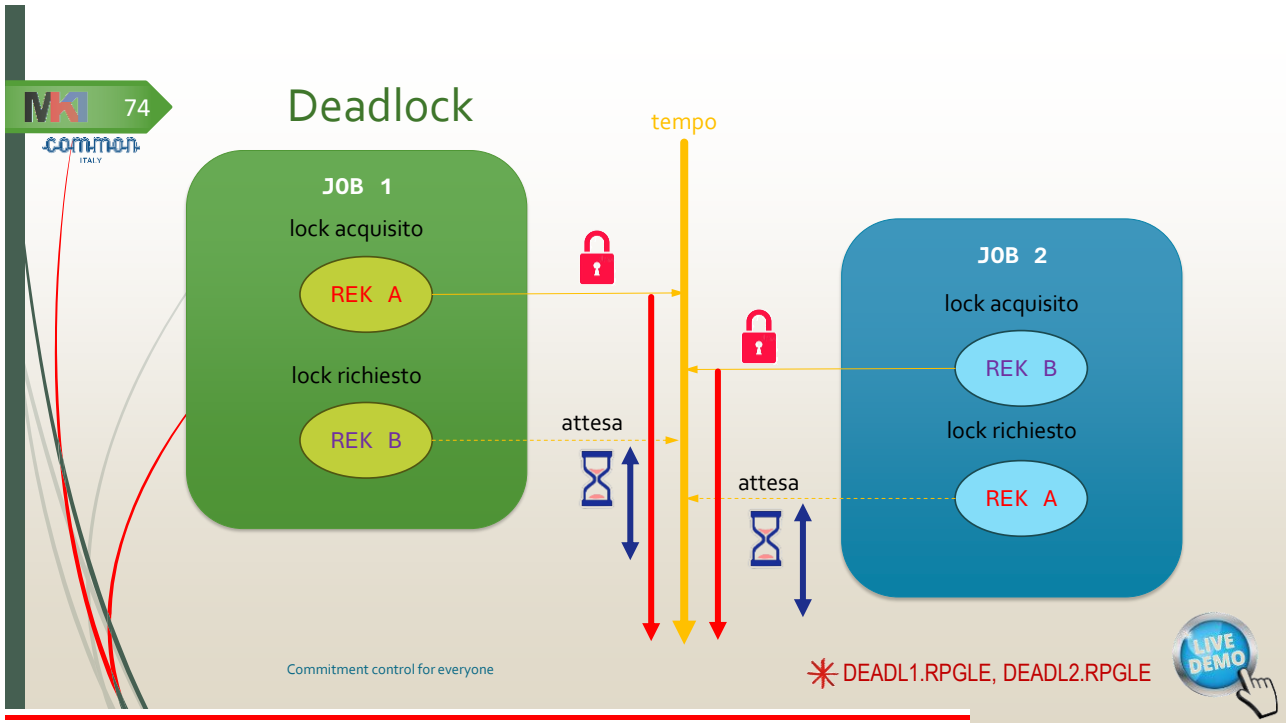
71



72



73



74

MK 75

common  
ITALY

## Controllo sincronia e cursori SQL/1

- Se si lavora con i cursori può essere necessario specificare la keyword **hold** per mantenere aperti i cursori  
`exec sql commit hold;`  
`exec sql rollback hold;`

Esempio log con cursore e commit senza keyword hold  
Creato ODP.  
Cursore CEMP aperto.  
1 righe richiamate dal cursore CEMP.  
1 righe aggiornate in EMPLOYEE in MK1SAMPLE.  
1 righe richiamate dal cursore CEMP.  
1 righe aggiornate in EMPLOYEE in MK1SAMPLE.  
1 righe richiamate dal cursore CEMP.  
1 righe aggiornate in EMPLOYEE in MK1SAMPLE.  
ODP cancellato.  
Commit completo.

Commit senza  
keyword hold  
chiude il cursore

Dopo commit alla successiva  
istruzione `fetch` restituisce  
sqlcode = -501

Commitment control for everyone

✱ TCMTCS1.SQLRPGL, TCMTCS2.SQLRPGL

75

MK 76

common  
ITALY

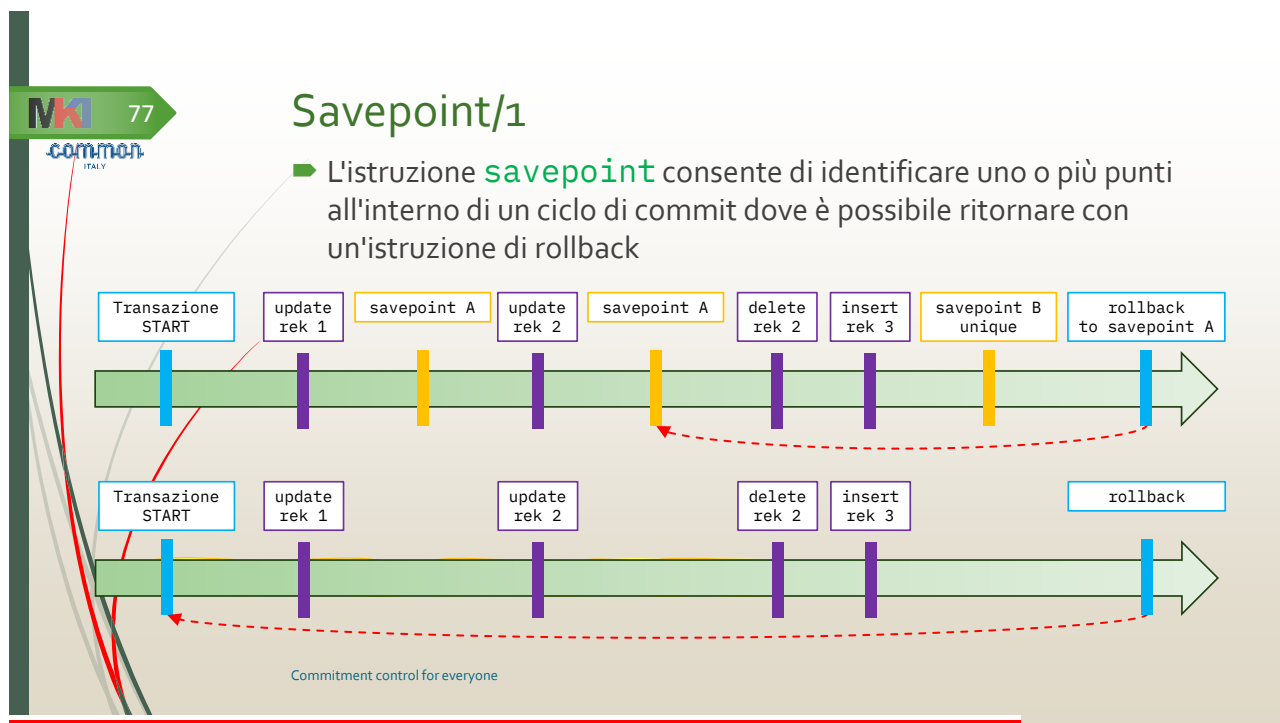
## Controllo sincronia e cursori SQL/2

- durante il ciclo di lettura di un *cursore di sola lettura* l'istruzione `rollback hold` non modifica il posizionamento del cursore
- se il *cursore* 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 *cursore* 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

✱ TCSRKB1A.SQLRPGL, TCSRKB1B.SQLRPGL, TCSRKB1C.SQLRPGL

76



77

**78**

## 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.o?topic=integrity-savepoints>

Commitment control for everyone

\* TCSRBK3C.SQLRPGLE

78

MK 79

common  
ITALY

## 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

79

MK 80

common  
ITALY

## 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

80



81

common

ITALY

## 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:

Immettere le opzioni e premere Invio.

5=Visual. stato 12=Gestione lavoro 14=Commit forzato

18=Rollback forzato ...

Opz	Definiz. commit	N° lav.	Uten. lav.	Nome lav.	Sottopr.	Utente
5	*DFACTGRP	110228		QPADEV000C	*NONE	
18		110228		QPADEV000C	*NONE	

Commitment control for everyone

81

82

common

ITALY

## Stato controllo sincronia in ACS

Gestione schemi > Database > Nome\_sistema > Transazioni > **Transazioni database**

Transazioni database - Include

Includi

ID unità di lavoro: Tutti

Stato unità di lavoro: Ripristina

Lavoro: Tutti

Utente: Tutti

Numero: Tutti

Risincronizzazione in corso: Tutti

Ripristina valori predefiniti

OK

Annulla

Stato risorsa della transazione APPN

X'5431BF98717'00001

Riga	Oggetto	Giornale	Comunicazione SNA	Connessione TCP/IP	Tabella remota	RDB	API
1	EMPLOYEE	EMPLOYEE	0	0	1	65217	

Proprietà di APPN

X'5431BF98717'00001

Generale

Dettagli

Opzioni di commit

ID unità di lavoro: APPN

Stato unità di lavoro: Ripristina

Registrazione data/ora: 16/09/2024, 22:46:16

Ruolo:

Operazione euristica:

Risincronizzazione in corso: No

Lavoro: 112407

Utente: QPADEV000F

Attivazione lavoro: SI

Definizione di commit: \*DFACTGRP

Descrizione: Gruppo di attivazione predefinito

Gruppo di attivazione: 2

Lotto dischi: \*SYSBAS

Lavoro ripristina:

Chiudi

N.B. occorre avere autorizzazione per il comando ADDPFTRG

Commitment control for everyone

82

83

common

ITALY

## 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-NAMED <sup>1</sup>	nome definizione (Q*)	==	nome job	*NONE

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

Commitment control for everyone

83

84

common

ITALY

## Nome definizione controllo sincronia

Activation group	Ambito	Nome
qualsiasi	job	*JOB
default activation group	activation group	*DFACTGRP
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

84

MK 85

common  
ITALY

## 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

85

MK 86

common  
ITALY

## 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

86



## 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

Commitment control for everyone

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

87



## 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 libreria e nome giornale

Commitment control for everyone

88



## 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

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

90

91

common

macchina

uomo

### Perché lo si teme?

- ...de che tutti i file siano ...ati sullo stesso giornale. Quindi ...er timore di dec...ento di perf...ce e di ...ne spazio dis...
- prima di ...V era ...minoso l'uso o ...sincronia ...ni
- prima di ILE ...llo di sincronia agiv ...dell'activa...
- poiché ...sono reg...i su giornale ...e ma...
- ...e nella manip...one di oggetti e librerie
- ...essità del disegno a...ativo per definire correttamente ...nsazioni e gli activation...


DON'T BE AFRAID

Commitment control for everyone

91

92

common



Commitment control for everyone

92

MK1

93

common

ITALY

## Glossario del controllo di sincronia

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

Commitment control for everyone

93

MK1

94

common

ITALY

## Bibliografia

- Database commitment control 7.6:  
[https://www.ibm.com/docs/en/ssw\\_ibm\\_i\\_76/pdf/rzakjpdf.pdf](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.o?topic=database-commitment-control>
- Documentazione IBM SQL reference 7.6:  
<https://www.ibm.com/docs/en/i/7.6.o?topic=reference-sql>
- Documentazione IBM SQL Programming 7.6 - Commitment control:  
<https://www.ibm.com/docs/en/i/7.6.o?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/fhgo31809-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

94

 95  




## Riferimenti



- E-mail aziendale: [marco.riva@ivolution.it](mailto:marco.riva@ivolution.it)
- Blog: [www.markonetools.it](http://www.markonetools.it)
- E-mail blog: [info@markonetools.it](mailto:info@markonetools.it)
- LinkedIn: [www.linkedin.com/in/marcoriva-mk1](http://www.linkedin.com/in/marcoriva-mk1)
- Twitter: [@MarcoRiva73](https://twitter.com/MarcoRiva73)
- Facebook: <https://www.facebook.com/markonetools/>
- YouTube: [youtube.com/@markonetools](https://youtube.com/@markonetools)

Commitment control for everyone

95

 96  


## 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.

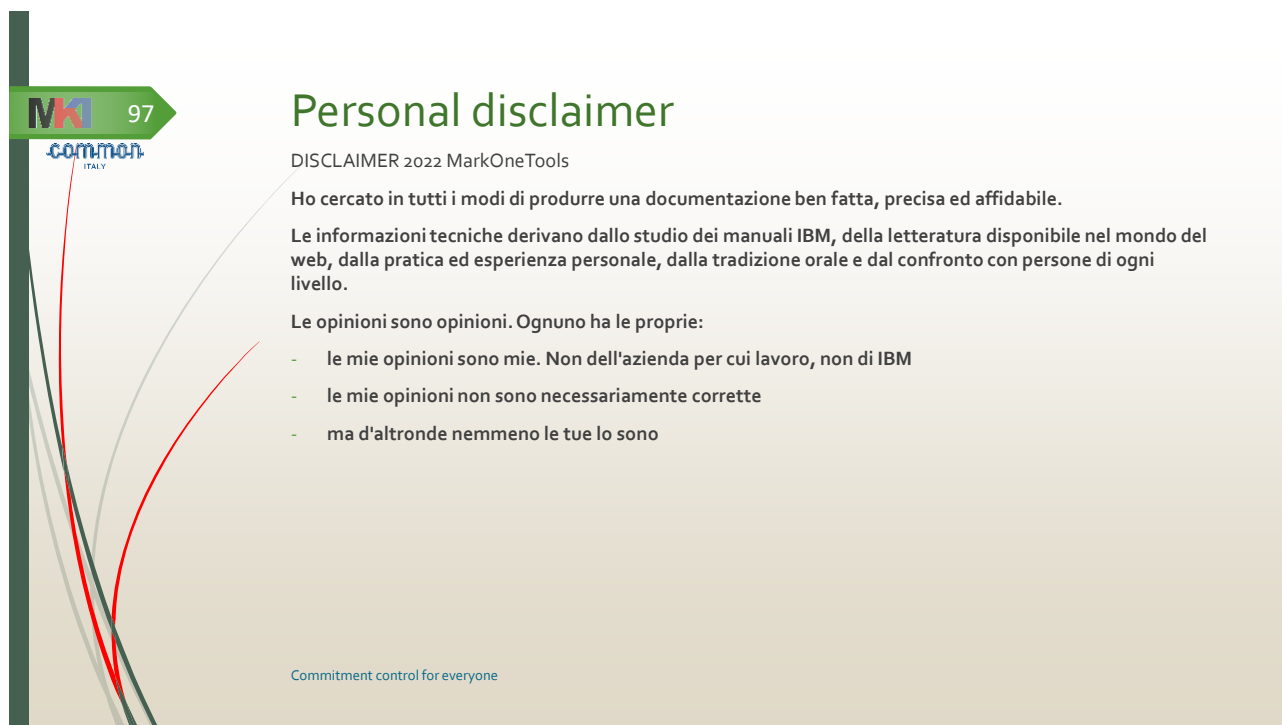
Le informazioni contenute nella presente documentazione possono essere riservate e sono, comunque, **destinate esclusivamente ai partecipanti al corso**. I contenuti del documento sono stati redatti con la massima cura e diligenza, e sottoposti ad un accurato controllo. **Si declina tuttavia ogni responsabilità**, 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

96



The slide features a light beige background with a vertical dark green bar on the left. A green arrow-shaped graphic on the left contains the 'MK1' logo and the number '97'. Below it is the 'common ITALY' logo. Three curved lines (one red, two grey) originate from the left and point towards the text. The text is in a clean, sans-serif font.

**97**

## 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