







Marco Riva

www.markonetools.it

IBM CHAMPION

2021-2025

Privacy area









Ultimo aggiornamento: 14/01/2026

1







		
<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 del corso

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



GitHub

MK1

Commitment control for everyone

4



Controllo di sincronia

Concetti base

Commitment control for everyone

5

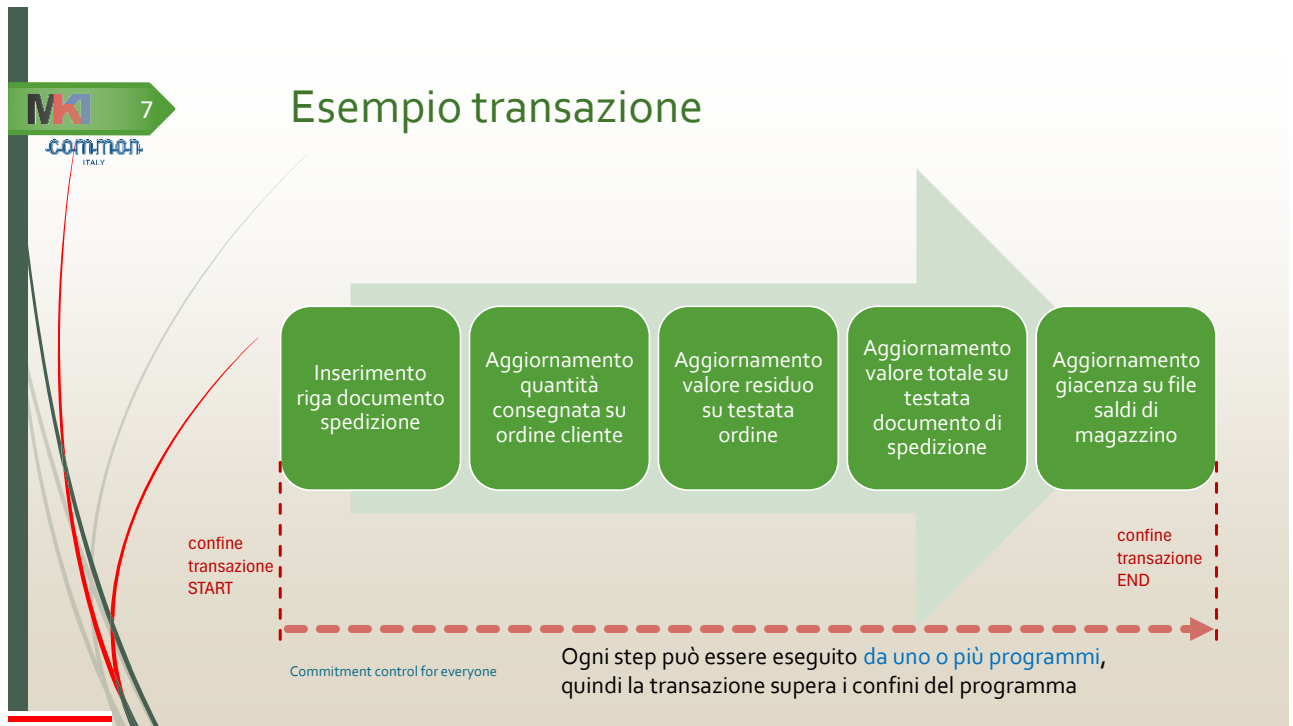


Controllo sincronia (commitment control)

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

Commitment control for everyone

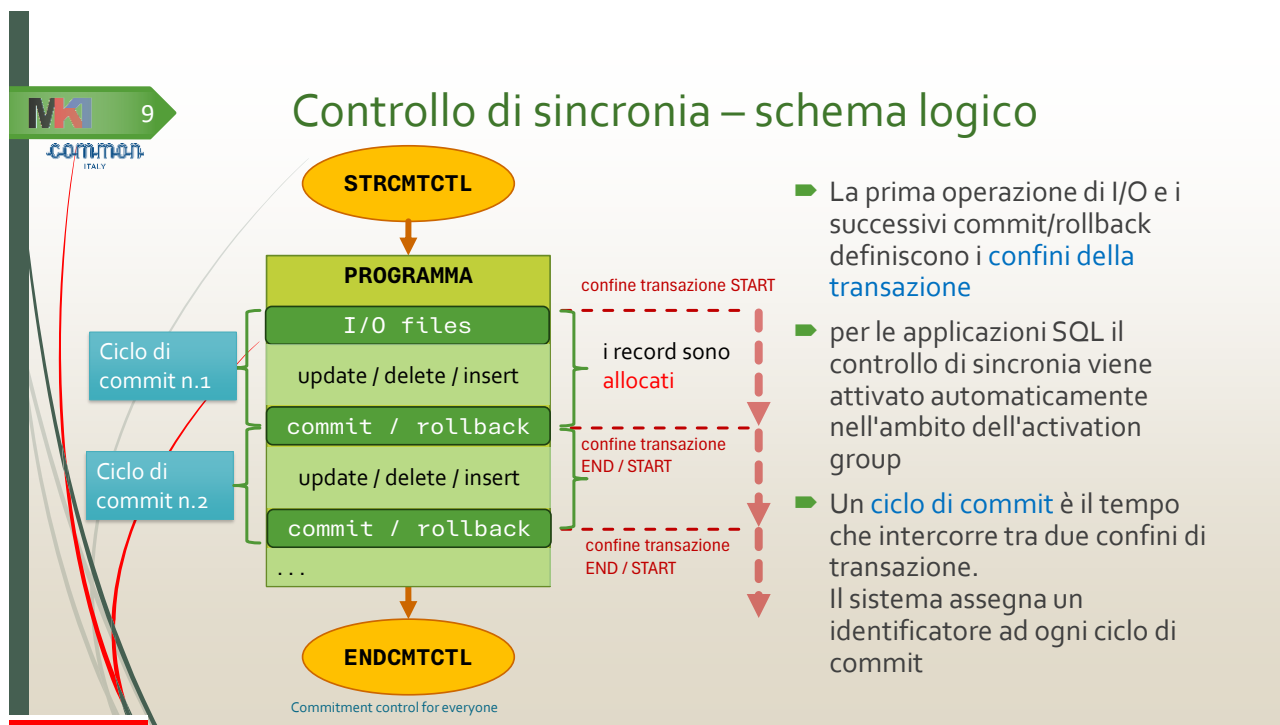
6



7



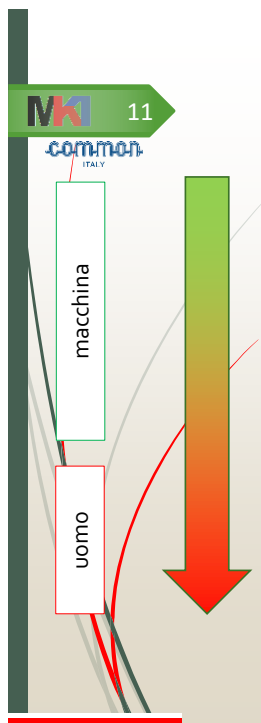
8



9



10



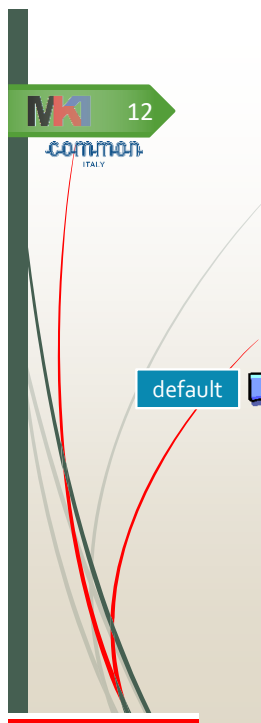
11

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

Ambito delle transazioni

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

Commitment control for everyone

12

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

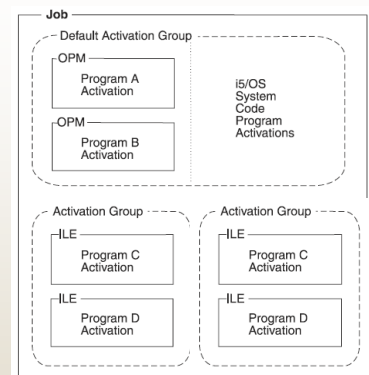
- L'**activation group** contiene le **risorse** necessarie ad eseguire il programma.

Le risorse sono:

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



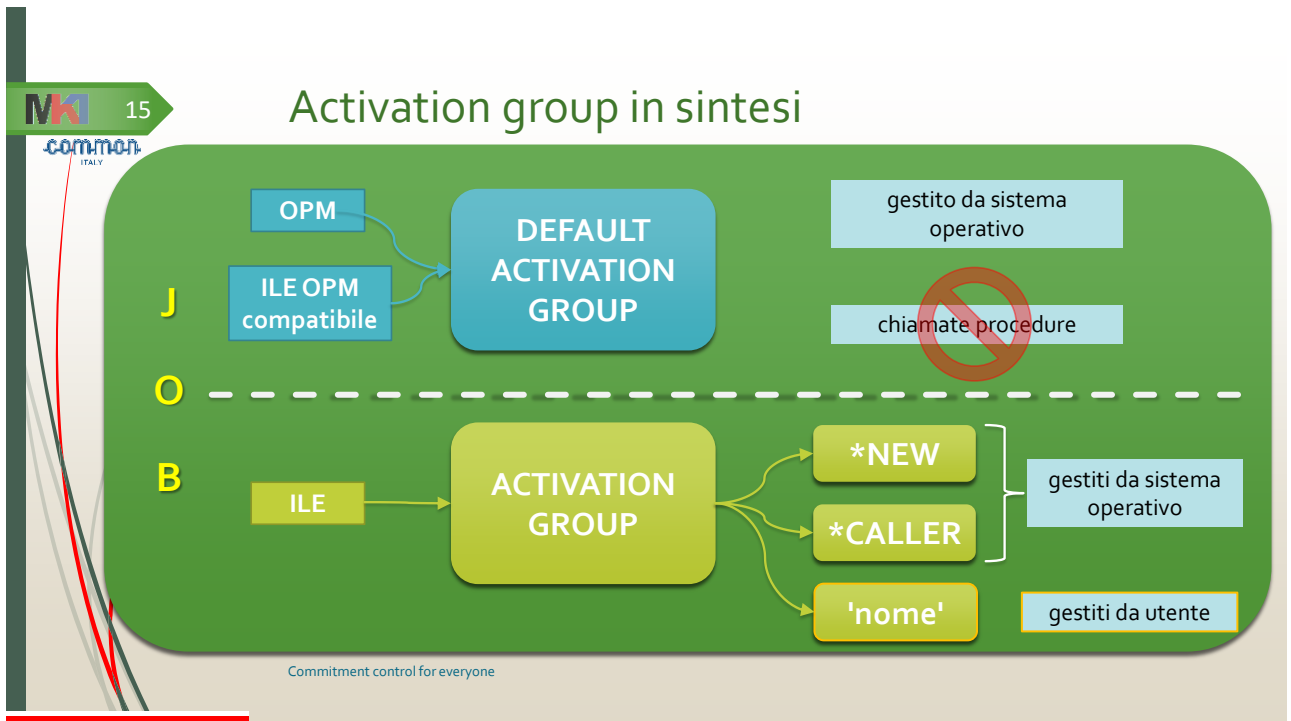
- All'avvio di un job, *automaticamente* vengono creati 2 activation group di **default** usati da tutti i programmi OPM e ILE OPM compatibile



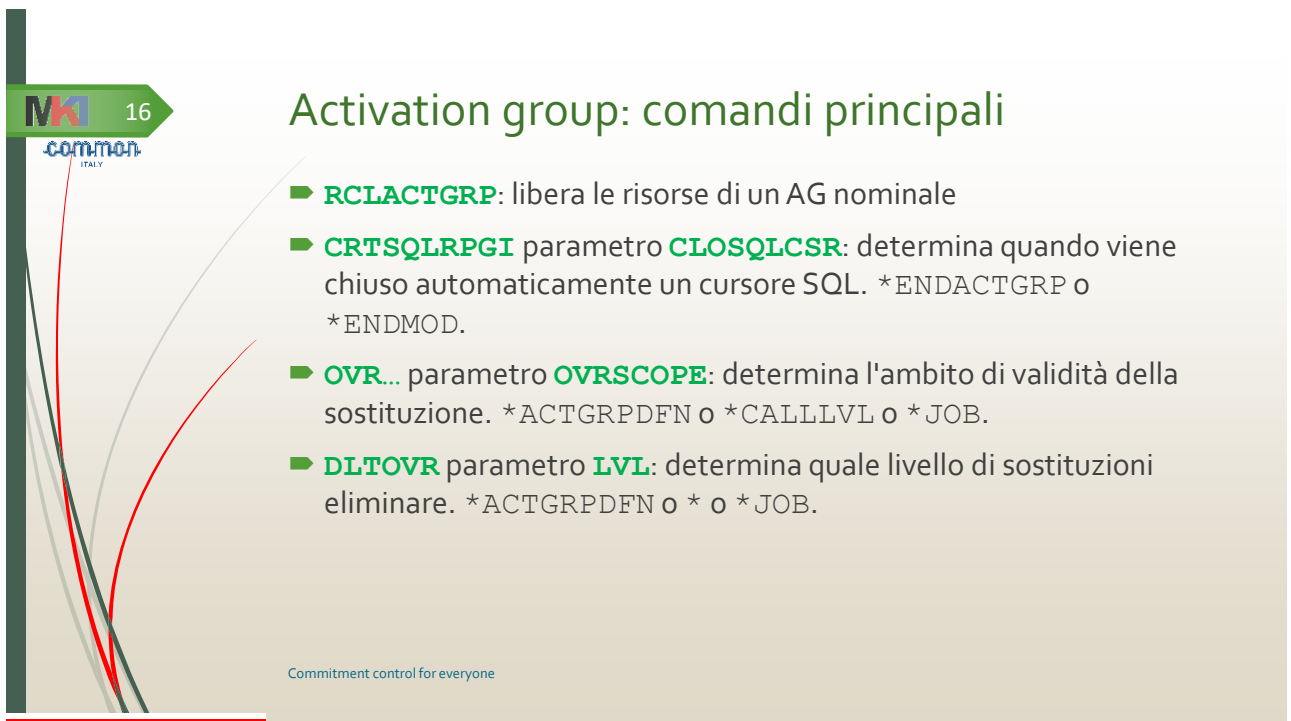
Fonte IBM

Commitment control for everyone

14



15



16

MK 17

common
ITALY

Comm.It. quiz

J
O
B

DEFAULT ACTIVATION GROUP

STRCMTCTL *JOB

PGMC

PGMD

ACTIVATION GROUP
XYZ

PGMA

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

YES

NO

17

18

common
ITALY

QUIZ TIME?

2

PGME
embedded sql
option commit = *chg

ACTIVATION GROUP 23
PGMF
embedded sql
option commit = *chg

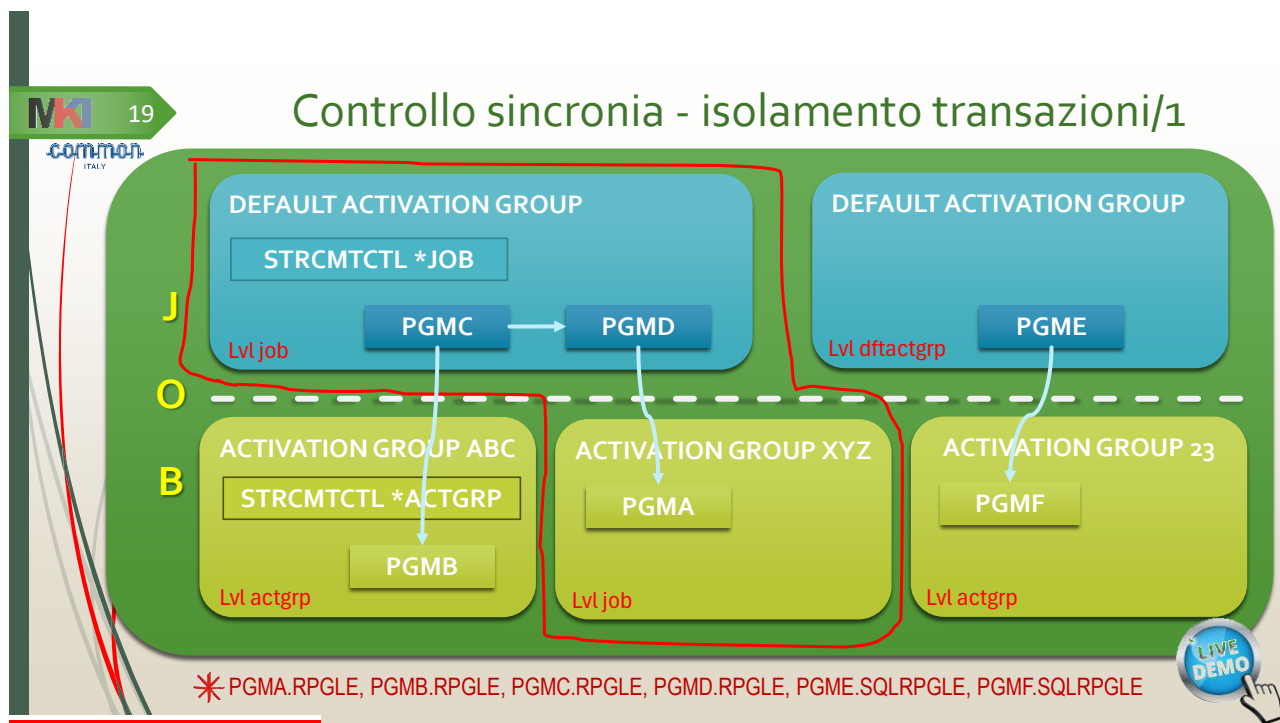
YES

NO

↑

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

18



19



20

MK 21

Controllo sincronia - isolamento transazioni/3

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

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

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

Commitment control for everyone

21

MK 22

Controllo sincronia - isolamento transazioni/4

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

Commitment control for everyone

22

MK1

23

common

ITALY

integrità e sicurezza del dato

Livelli di isolamento delle transazioni

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.

<https://www.ibm.com/docs/en/i/7.6.o?topic=concepts-isolation-level>

MK1

24


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>



25



Livello di allocazione

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

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

Commitment control for everyone

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

MKI 26

common
ITALY

Prerequisiti per attivare controllo di sincronia

Specifiche di controllo e definizione file

Opzioni precompilatore SQL

Commitment control for everyone



H specifiche di controllo

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

F specifiche definizione file

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

Commitment control for everyone

27



STRCMTCTL / ENDCMTCTL

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

Commitment control for everyone

28

MK 29

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>

29

MK 30



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

30

 31



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 6o)




Commitment control for everyone

31

 32




To be continued...



Commitment control for everyone

32