



# Central Canada Db2 User Group

## Db2 for z/OS 101: Buffer pools and group buffer pools

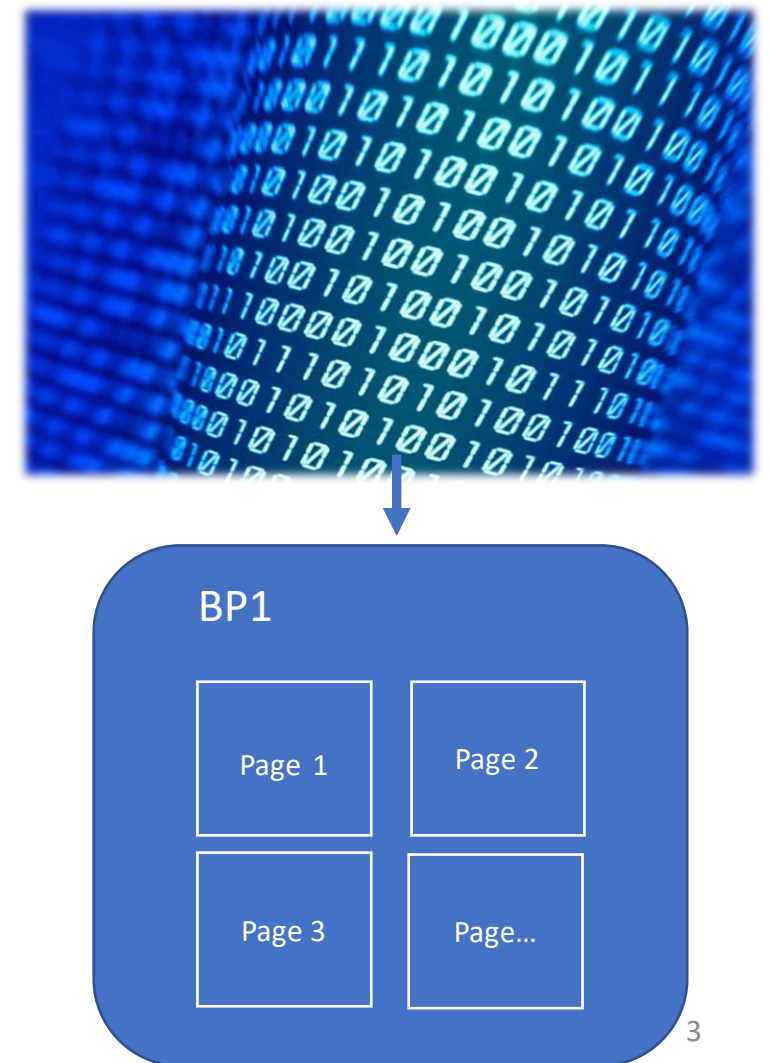
Tori Felt, Keziah Knopp  
Db2 for z/OS Specialists  
September 2023

# Agenda

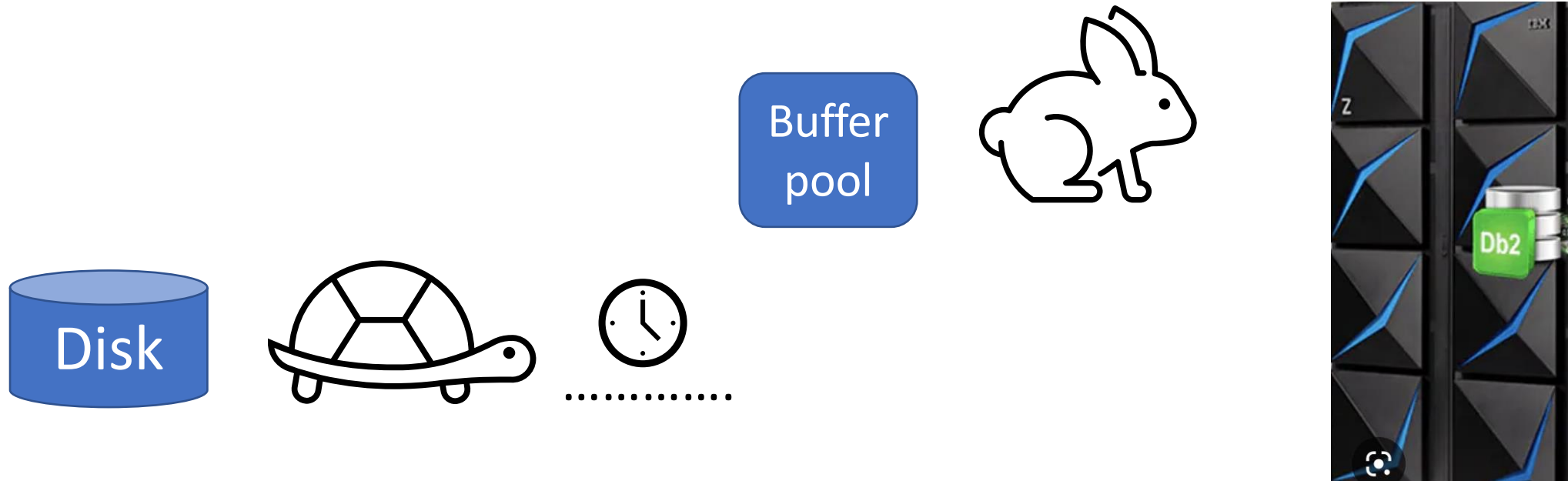
- What is a Db2 buffer pool?
  - How buffer pools are used
  - Some details of buffer pools
- What is a Db2 group buffer pool?
  - What is data sharing?
  - How group buffer pools are used
  - Some details of group buffer pools
- What's next?
- Questions

# What is a Db2 buffer pool?

- A Db2 buffer pool:
  - Contains the data with which applications interact
  - Is an area of virtual storage containing **pages** of tables or indexes
  - Caches data in memory to be reused by the same or other applications
- Every Db2 database object must be assigned to a buffer pool!



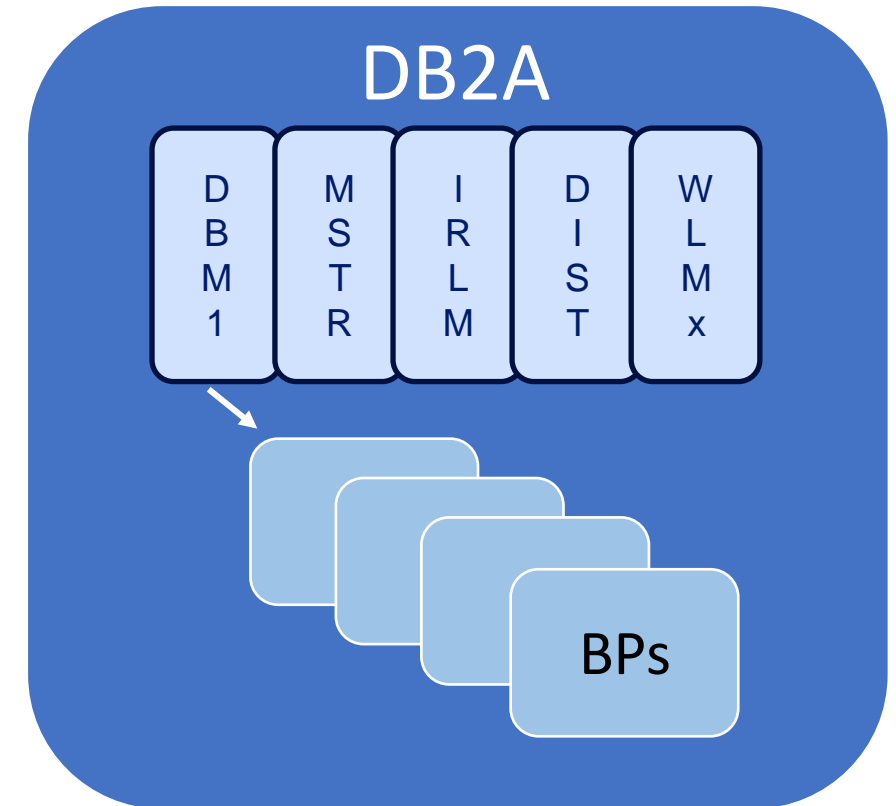
# Why do we use buffering?



- Minimize I/O activity to disk
  - When data is already in a buffer, an application program does not wait for data retrieval from disk
- Reduce overall CPU cycles and I/O wait time

# More about buffer pools

- Buffer pools:
  - Reside in the Db2 database services address space (aka DBM1)
  - Are defined during Db2 installation
    - Later you can ALTER BUFFERPOOL to change them
- Buffer manager:
  - Retrieves the **pages** for the index manager or data manager



# BPs: virtual, real and auxiliary storage

- Buffers reside in *pages* in **virtual** storage
- To read or update a buffer, the page must be in a **real** storage *frame*
  - Real storage = Central storage = Memory
- If real storage is constrained, pages may be moved to an **auxiliary** storage *slot*



**virtual** storage pages



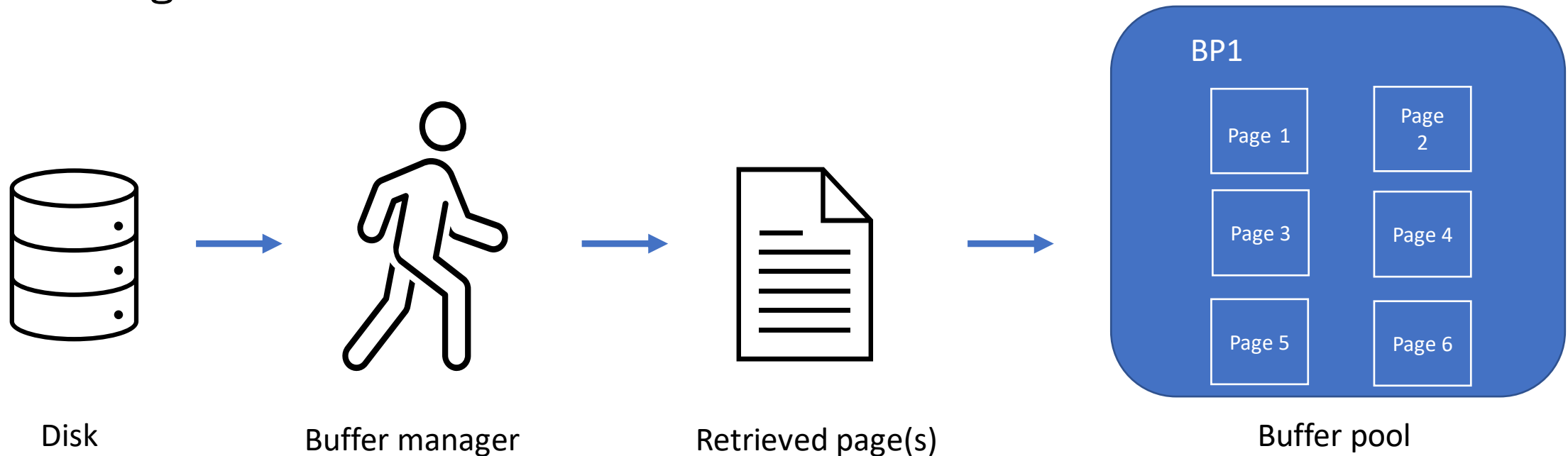
**real** storage frames



**auxiliary** storage slots

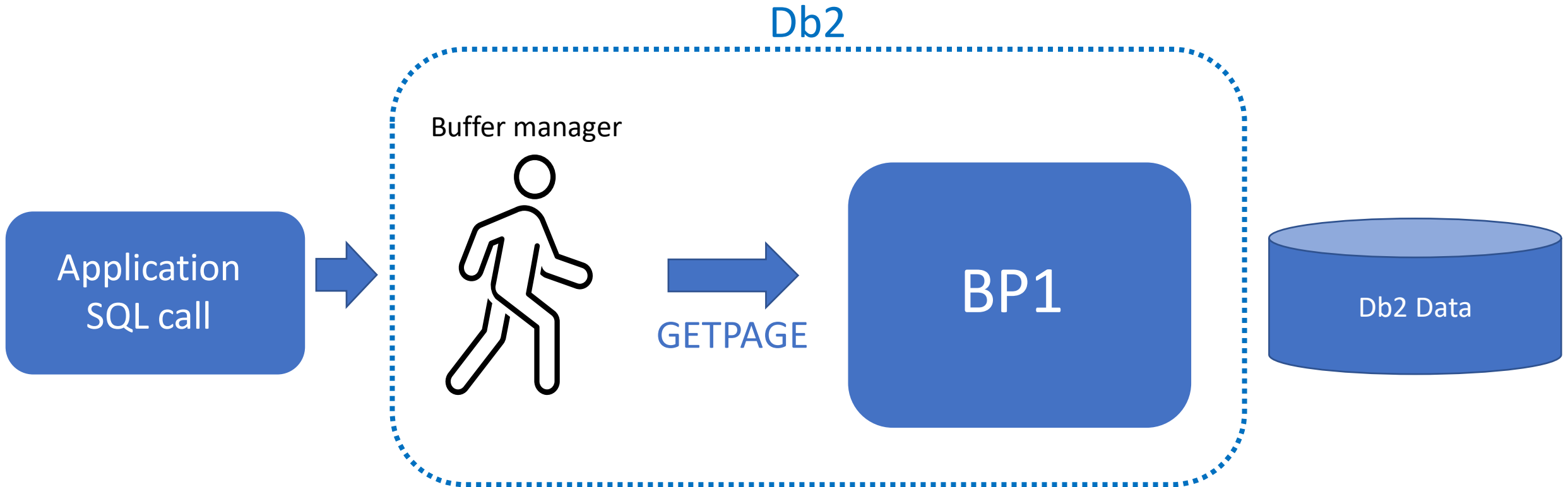
# What happens when a page is retrieved?

The retrieved page is placed in the buffer pool associated with the Db2 object being accessed



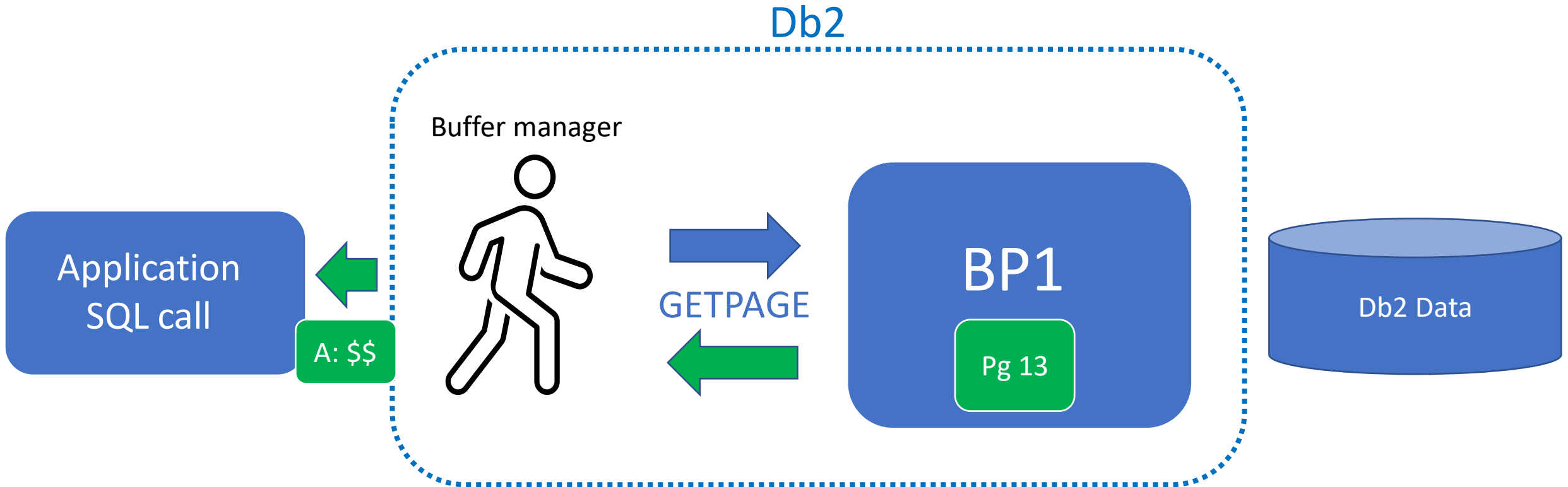


# Buffer manager and GETPAGEs

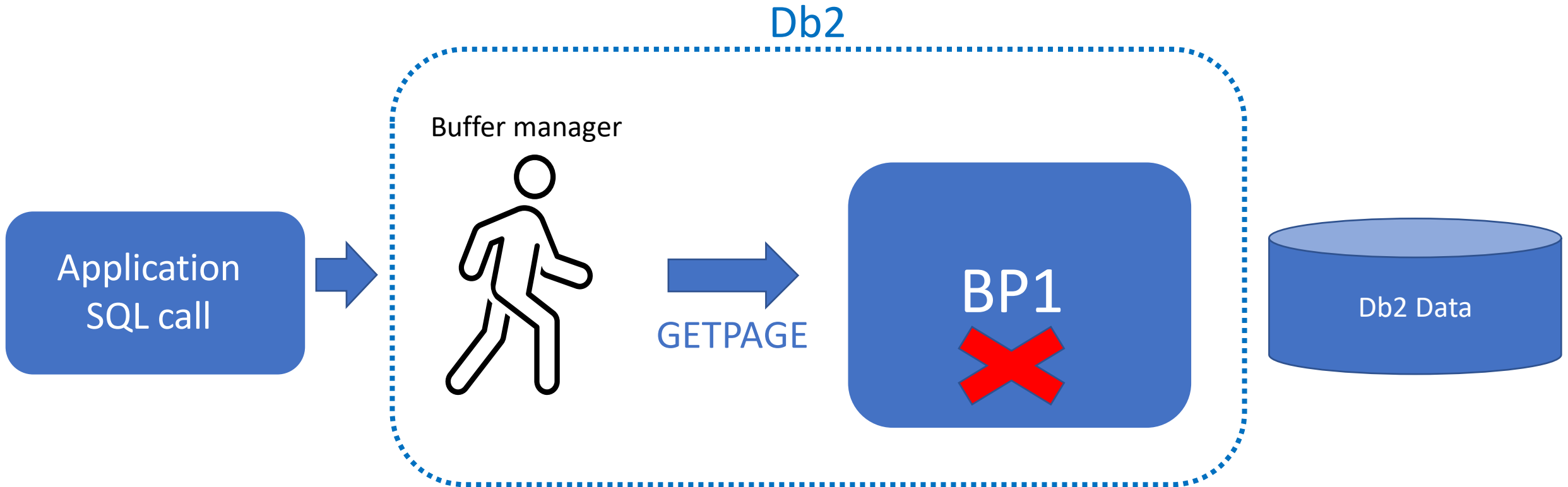




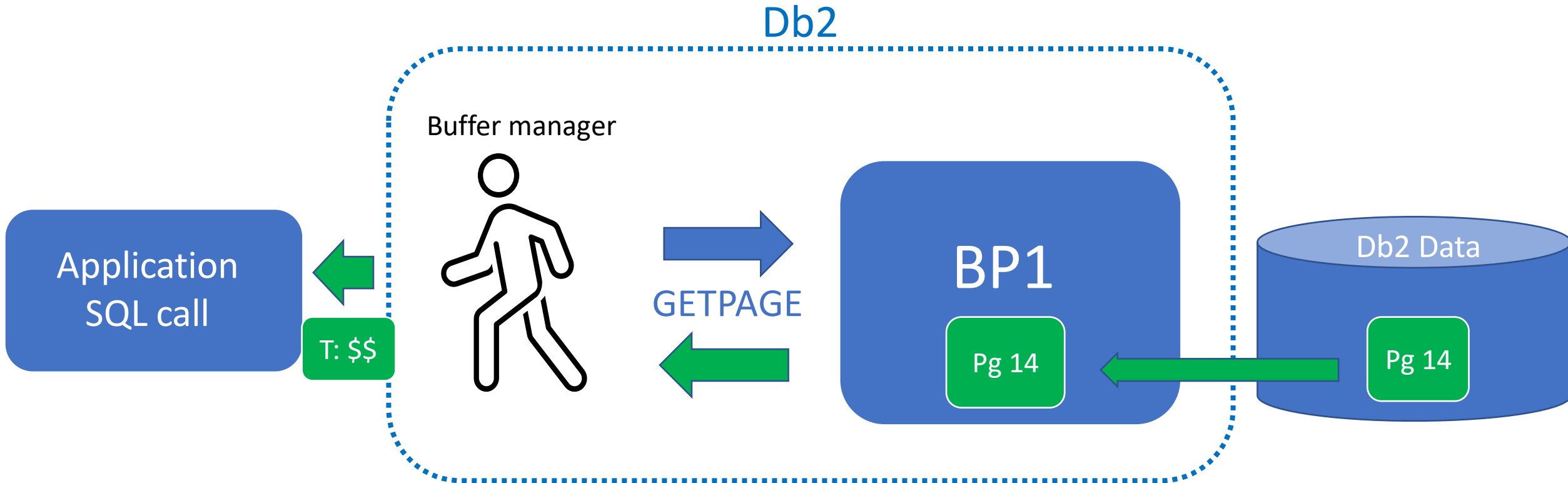
# Buffer manager and GETPAGEs



# Buffer manager and GETPAGEs



# Buffer manager and GETPAGEs



# I/O operations into a buffer pool

Buffer manager does 2 kinds of I/O to retrieve pages:

## 1. Synchronous

- Application waits for the page to be retrieved

## 2. Asynchronous – multiple pages

- Prefetch!
- Goal: the pages are in the buffer pool before the application needs them

# Prefetch

Db2 anticipates which pages an application will need

- Sequential prefetch: Db2 Optimizer chooses in advance as part of the access path
- Dynamic prefetch: Db2 identifies a sequential pattern and triggers prefetch
- List prefetch: Db2 reads a set of pages determined by a list of record identifiers (RIDs) taken from an index or from the Db2 log

# What happens after a page is retrieved?


- At any time, a page in a buffer pool can be in-use, updated or available:

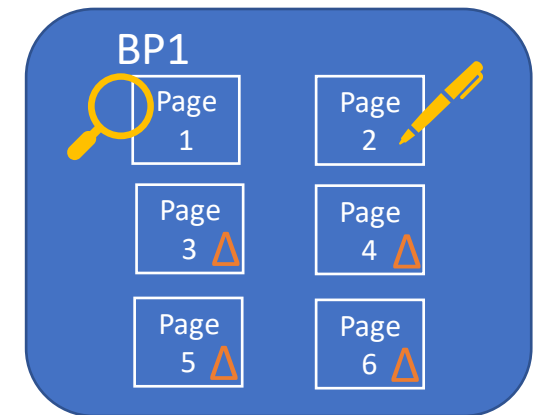
- In-use pages:

- Currently being read or updated
- If a page is being updated, it is being accessed exclusively by one agent



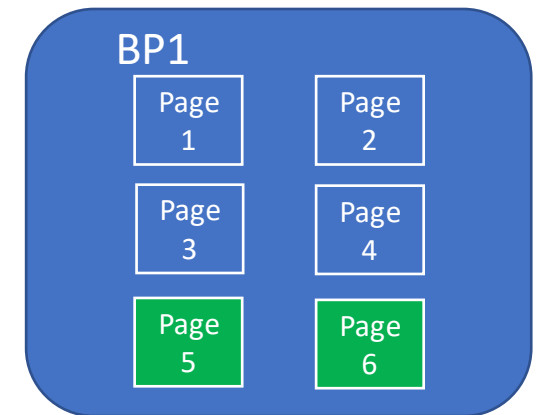
- Updated

- Pages contain changed data that has not been written to disk yet 



# What happens after a page is retrieved?

- At any time, a page in a buffer pool can be in-use, updated or available:
  - In-use pages:
    - Currently being read or updated
    - If a page is being updated, it is being accessed exclusively by one agent
  - Updated
    - Pages contain changed data that hasn't been written to disk yet
  - Available
    - Ready for use or stealing (Page 5 and Page 6 in the example)
    - Changed data has been written to disk





# Buffer pool page updates

SQL statements can change buffer pool data for tables and indexes:

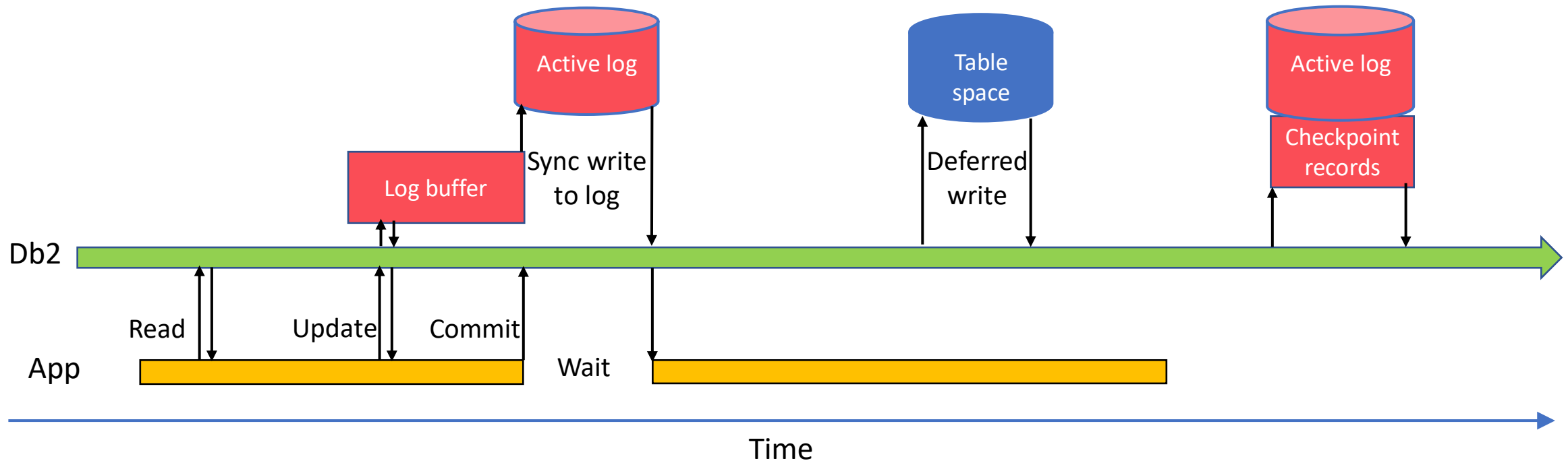
- INSERT
- UPDATE
- DELETE
- MERGE
- ....

\* This data must eventually be written back to disk \*

Utilities can also update buffer pool pages

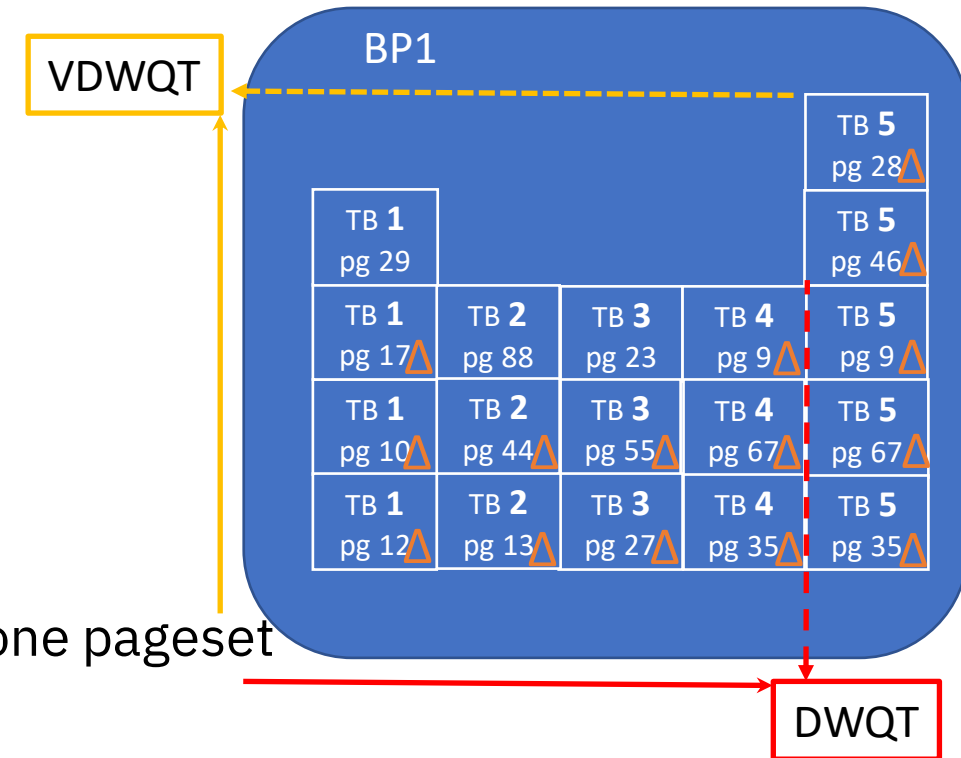
# Db2 buffer pool write activity

- Db2 uses deferred writes to optimize I/O processing
- Deferred writes can write multiple pages in one I/O operation

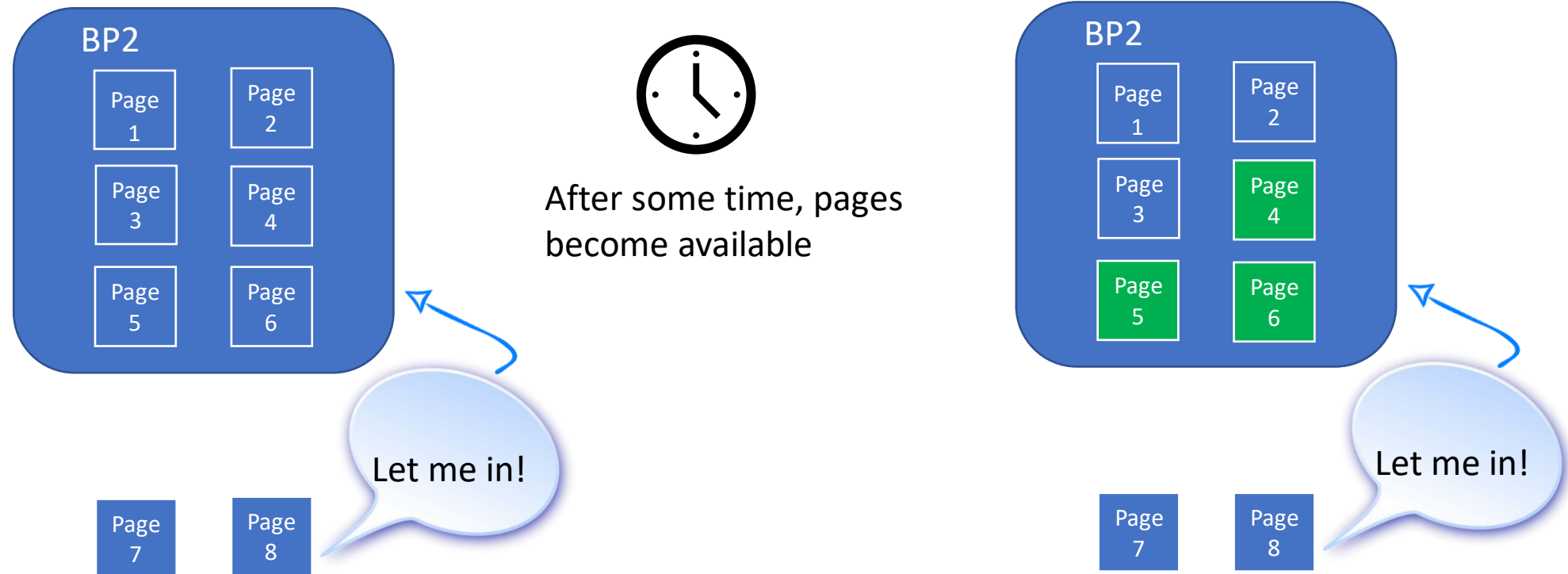


# Db2 deferred writes

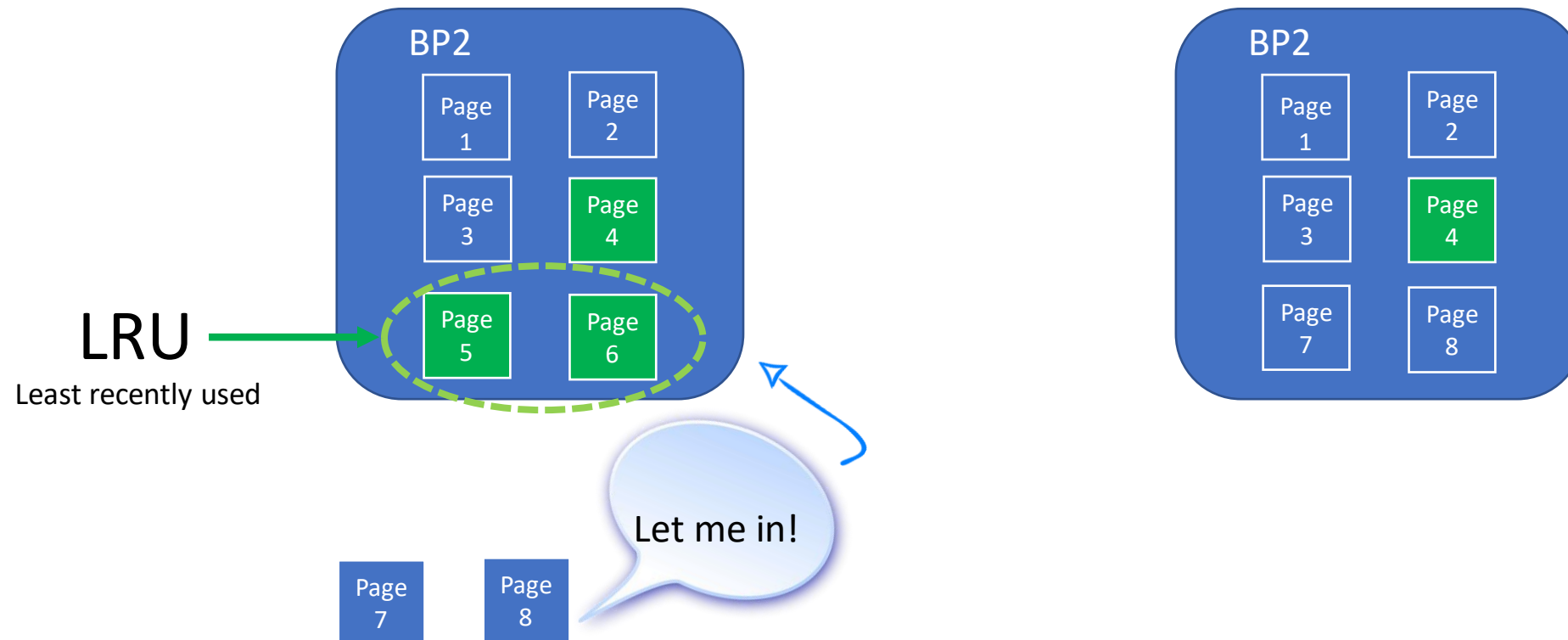
- Deferred writes are triggered by the following:
  - Checkpoint processing
  - DWQT – horizontal deferred write queue threshold
    - How full the buffer pool is of unavailable pages
  - VDWQT – vertical deferred write queue threshold
    - How full the buffer pool is of unavailable pages in one pageset
- Db2 checkpoint can trigger synchronous writes



# Page stealing [1|2]



# Page stealing [2|2]



# Page stealing algorithms

- If a page is available, it can be stolen by a new page:
  - LRU – least recently used
    - Keeps pages in the buffer pool that are being used frequently and replaces the others
  - FIFO – first in first out
    - Removes the oldest pages in the buffer pool, no matter how frequently they are being used
  - NONE
    - Db2 pre-loads the pages into the buffer pool when an object is opened
    - Keeps all pages for an object in the buffer pool

# Buffer pool configuration

- Db2 provides up to 80 buffer pools with pages sizes shown below

| BP Name         | # of pools available | Size of the buffer page |
|-----------------|----------------------|-------------------------|
| BP0 – BP49      | 50                   | 4 KB                    |
| BP8K0 – BP8K9   | 10                   | 8 KB                    |
| BP16K0 – BP16K9 | 10                   | 16 KB                   |
| BP32K – BP32K9  | 10                   | 32 KB                   |

- Maximum amount of aggregate buffer pool space is 16 TB
  - z/OS 2.5 allows up to 16 TB of real storage (storage > 4 TB in 2 GB frames)
- Page size is determined by the buffer pool to which the object is assigned
  - Long rows fit more efficiently in larger buffer sizes
    - E.g. if row size = 2200 bytes, only 1 row fits in 4 KB page, with almost 50% space wasted



# Buffer pool size

- Buffer pool size can affect application performance
  - Virtual pool size (VPSIZE) is number of buffers
  - $VPSIZE \times \text{page size} = \text{buffer pool size}$ 
    - E.g. if VPSIZE of BP1 is 100,000, then buffer pool size is 400,000 KB
- Larger buffer pools allow more pages to remain in the buffers longer



# BPs: virtual, real and auxiliary storage

***virtual*** storage pages



***real*** storage *frames*



***auxiliary*** storage slots



# Long term page fix

- For Db2 to perform I/O for a page, the page must be fixed in the frame: page fix
  - Db2 issues a PGFIX instruction before the I/O
  - Followed by PGFREE instruction when the I/O completes
- If BP is PGFIX=NO, Db2 must fix the page for every I/O operation (PGFIX + PGFREE)
- If BP is PGFIX=YES, Db2 does not issue a PGFIX instruction for every read or write
  - Pages are all fixed when BP allocated - this saves CPU!
  - 'long term page fix'
- PGFIX=YES requires sufficient real memory to avoid system paging
  - Best candidates for PGFIX=YES are BPs with highest I/O rates
    - Or highest GETPAGE rates for large frames

# Frame size

- Possible values are 4K, 1M, or 2G
  - Large frames can save CPU related to dynamic address translation (mapping virtual to real)
  - FRAMESIZE > 4K requires PGFIX=YES
  - If FRAMESIZE = 1M and PGFIX=NO, Db2 uses 4K frames
- FRAMESIZE of 1M or 2G requires z/OS team to define LFAREA (large frame area)
- A buffer pool with FRAMESIZE = 2G will only be allocated in 2G frames if 2G of LFAREA is available
  - If 2G frames cannot be allocated, Db2 allocates 1M frames – if available
- Best candidates for FRAMESIZE > 4K are those BPs with highest GETPAGE rates

# Frame size



4 KB frame

1 MB frame

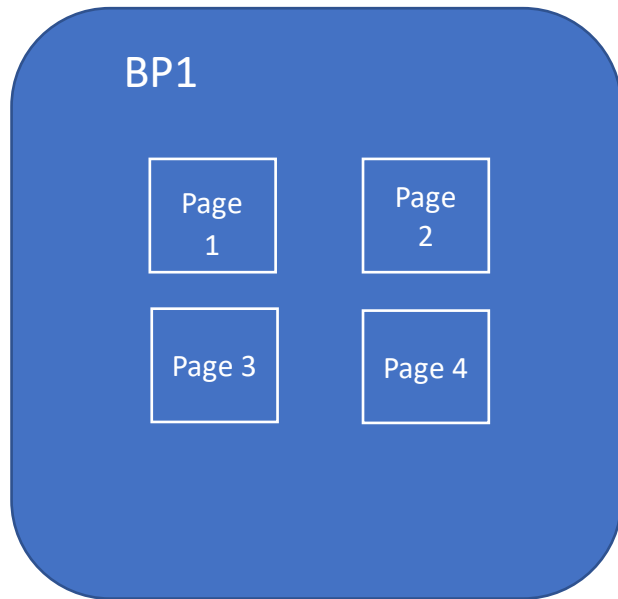
2 GB frame



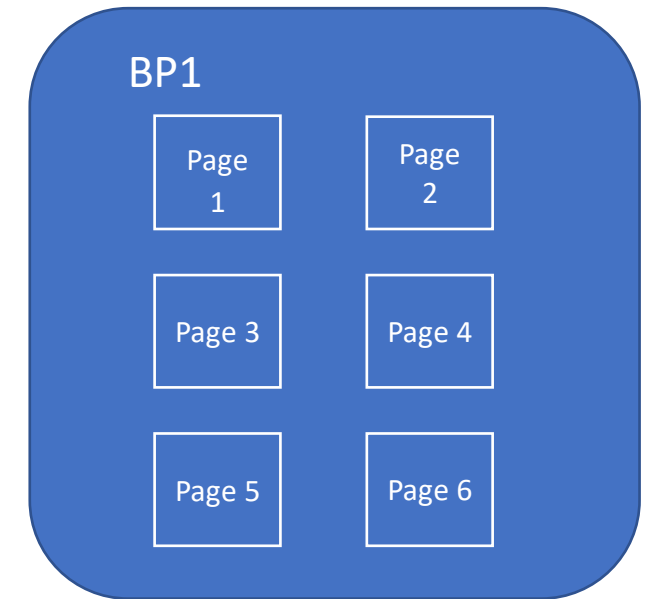
# How do you change a buffer pool?

- -ALTER BUFFERPOOL command to change:
  - VPSIZE – buffer pool size in pages
  - PGSTEAL – page steal method
    - LRU, FIFO, NONE
  - DWQT – horizontal deferred write queue threshold
  - VDWQT – vertical deferred write queue threshold
  - PGFIX – indicates if pages are fixed in real storage when initially used
  - FRAMESIZE – sets the frame size for the buffer pool
  - ....

# -ALTER BUFFERPOOL example



`-ALTER BUFFERPOOL(BP1) VPSIZE(6)`





# Buffer pools summary

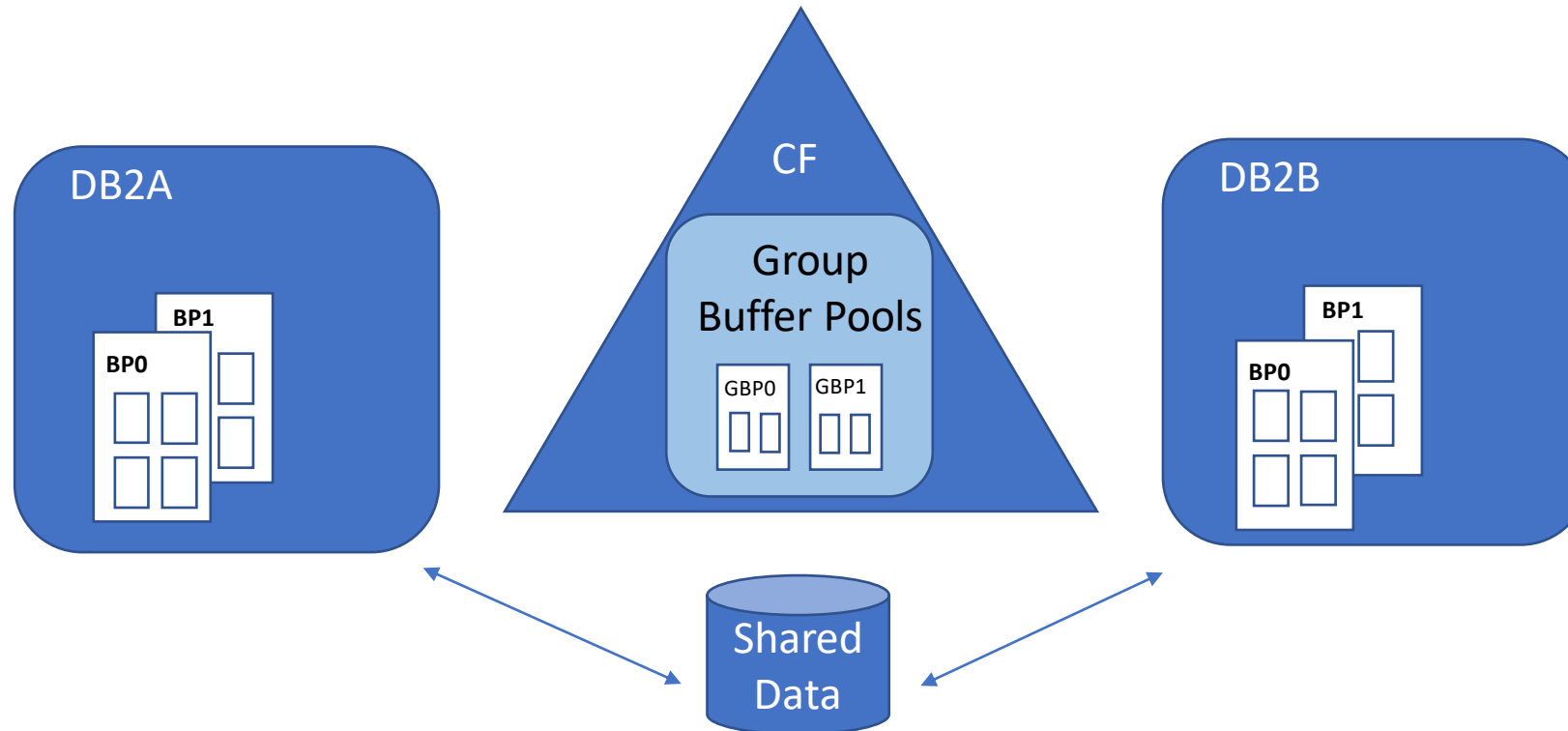
# Buffer pools

- Virtual storage pages in DBM1
- Buffer manager
- Cache data for application use
- In-use, updated, or available
- Deferred write thresholds: DWQT, VDWQT
- Page stealing: LRU, FIFO, NONE
- Long term page fix – reduced CPU
- Large frames – reduced CPU
- -ALTER BUFFERPOOL

# Group buffer pools

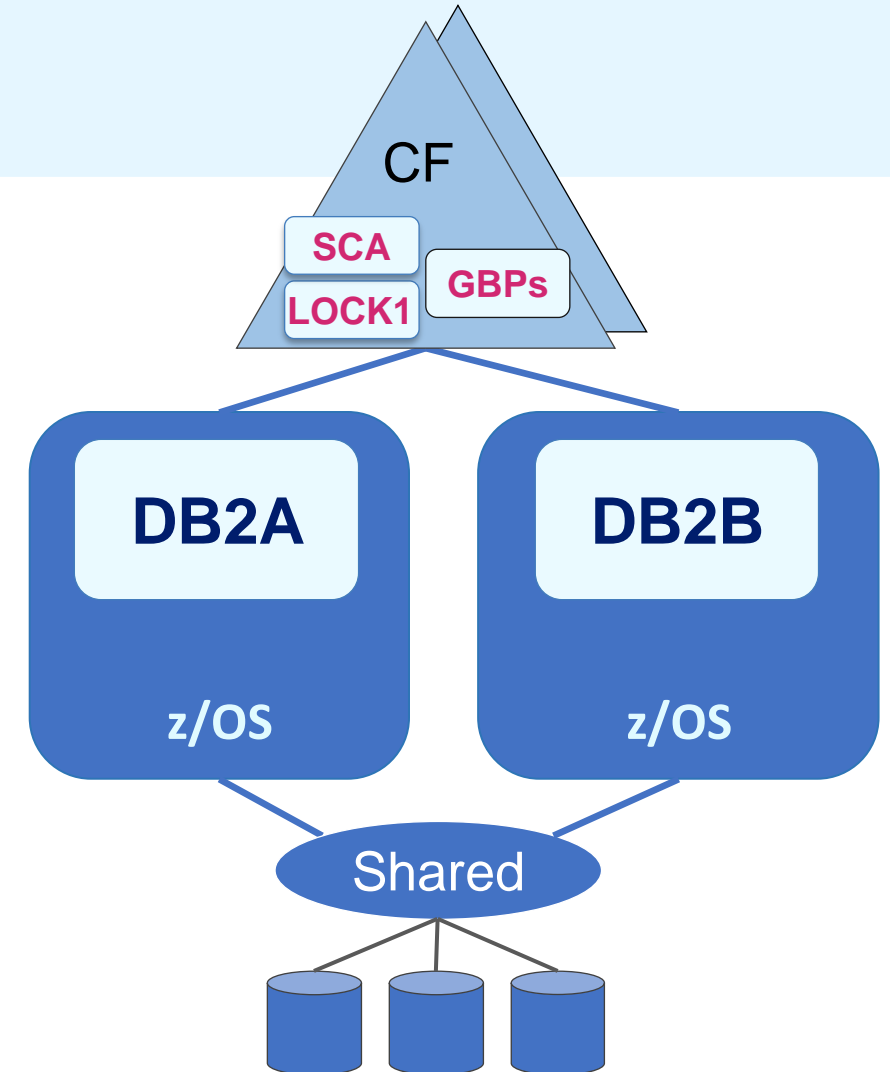
# Db2 group buffer pools (GBPs)

- Used in Parallel Sysplex and Db2 data sharing
- Reside in the coupling facility (CF) to share buffer information between Db2s



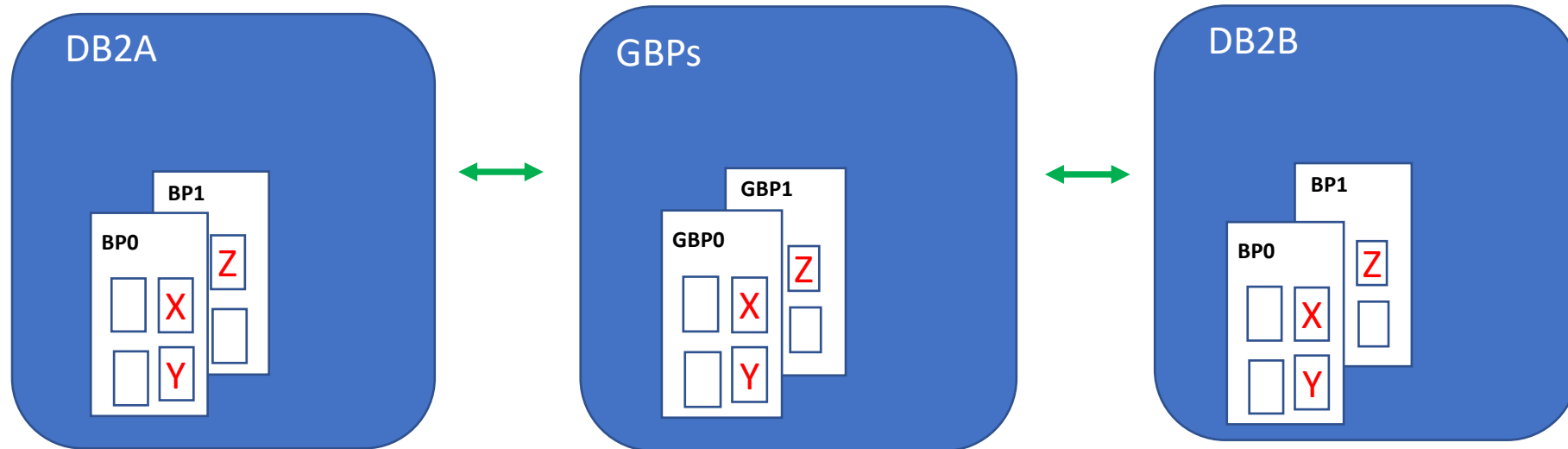
# What is Db2 data sharing?

- A collection of one or more Db2 subsystems that share Db2 data is called a data sharing group
- A Db2 subsystem that belongs to a data sharing group is a member of that group
  - Each member can belong to one, and only one, data sharing group



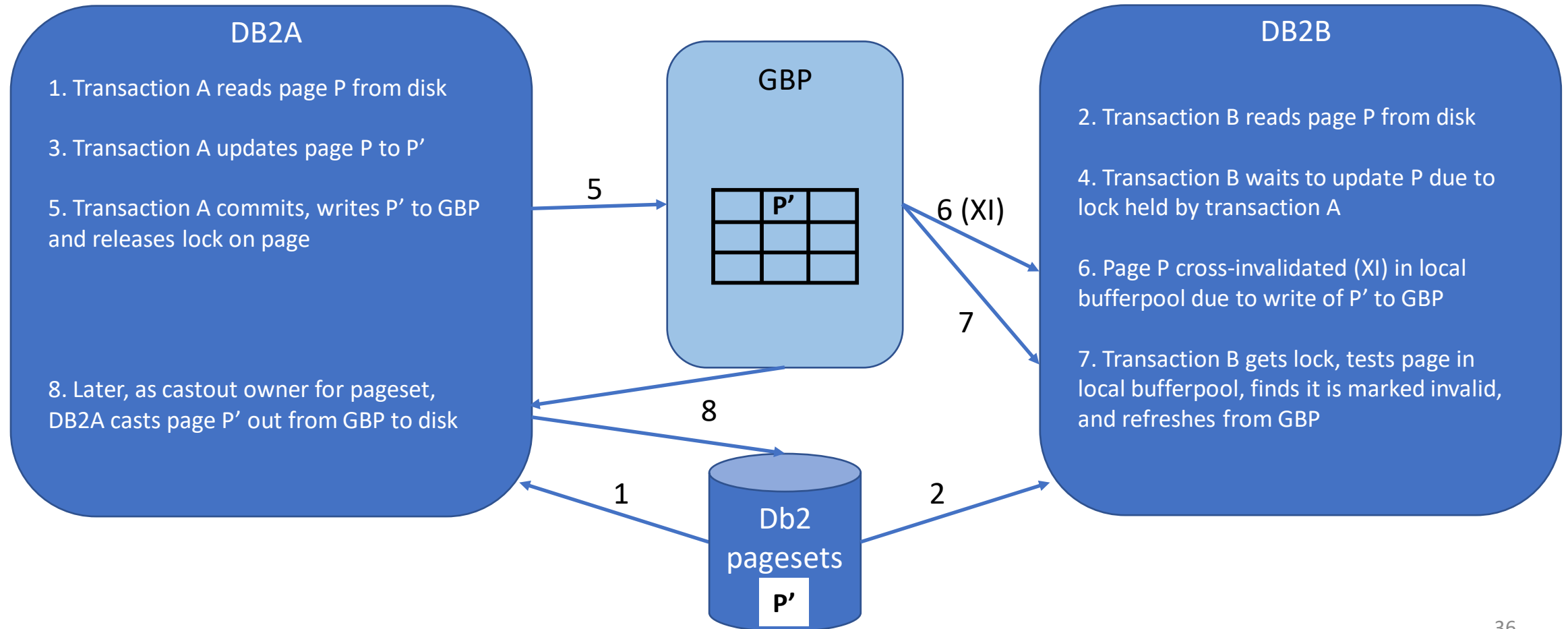
# Why does Db2 use group buffer pools?

- What if both Db2 members have the same data in their buffer pools?
  - GBP0 and GBP1 to the rescue!



There is one GBP for all buffer pools of the same name that are actively sharing data

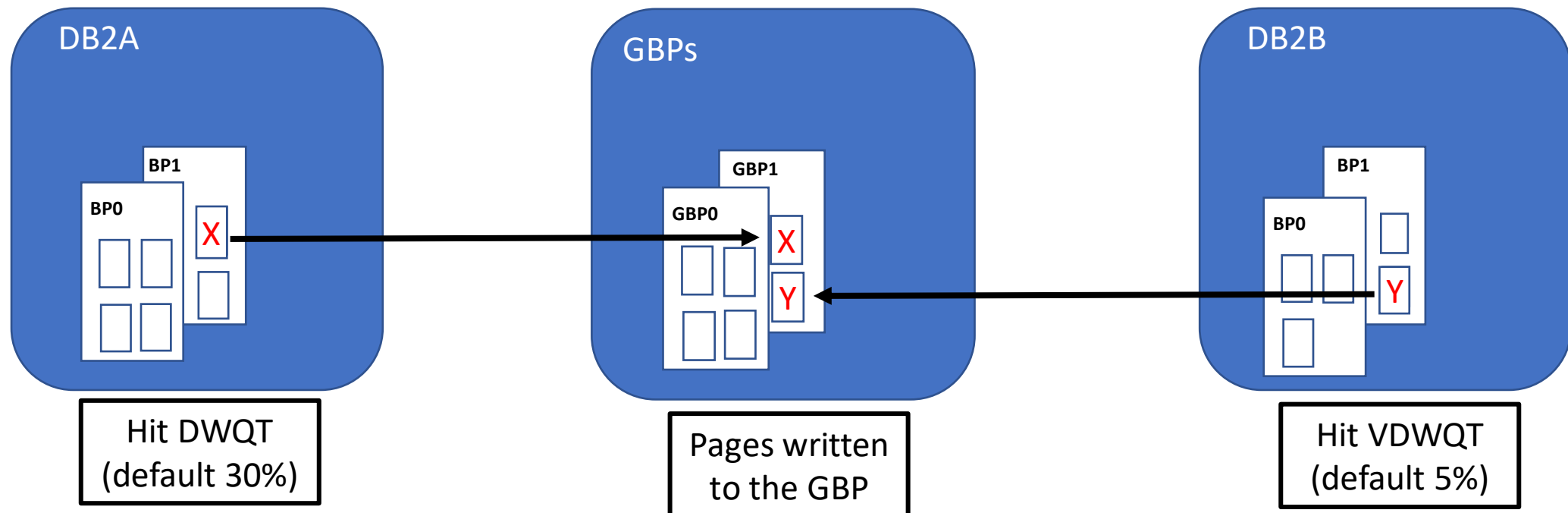
# Db2 GBPs in action





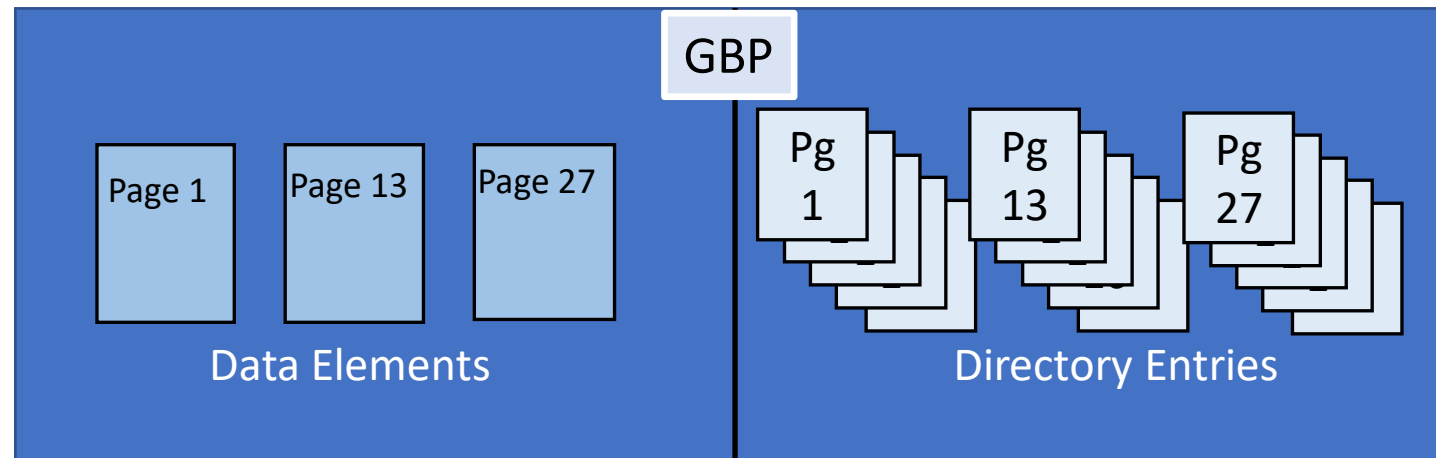
# Writes and data sharing

- At the latest, changed pages are written to GBP at commit - synchronously
- Changed pages may be written to GBP before commit



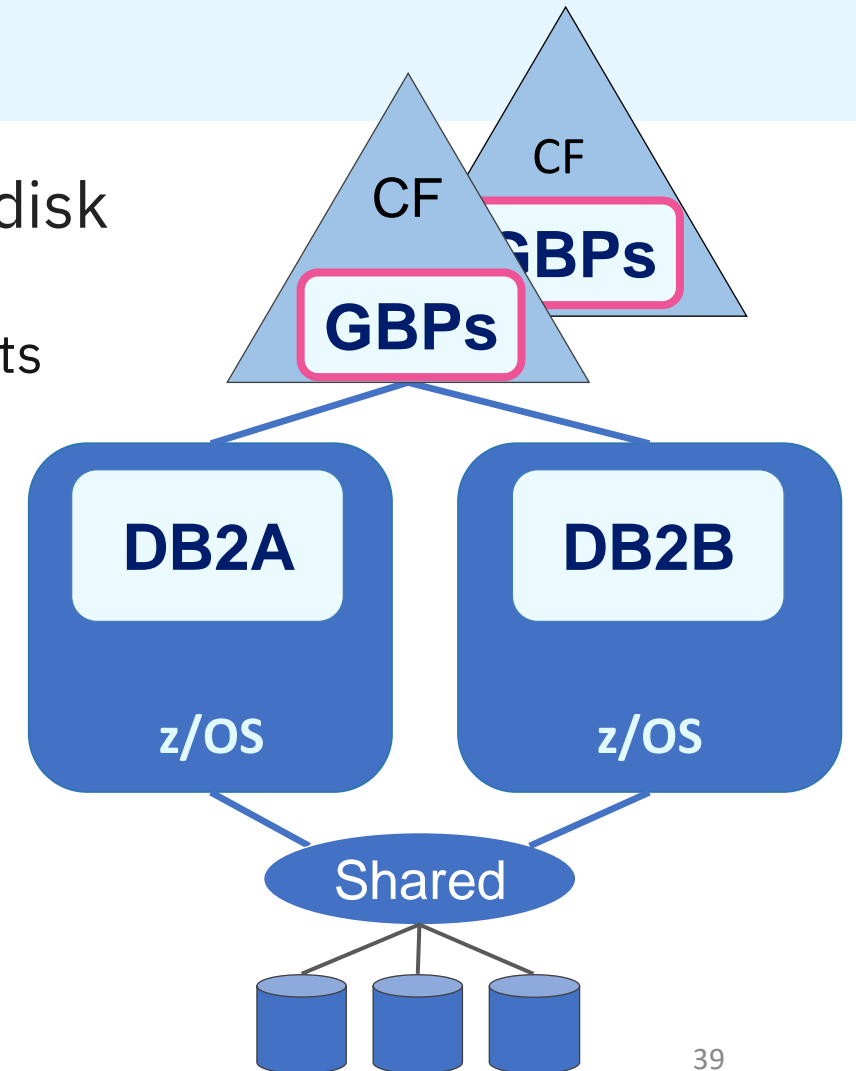
# Db2 group buffer pool details

- Directory entries:
  - Manage coherency by tracking any Db2 member's interest in table or index pages
    - Pages in local buffer pools and in GBPs
  - Drive cross-invalidation (XI) - when a member needs a page of data, it tests to see if the buffer contents are still valid
- Data elements:
  - Cache pages that a Db2 member changed

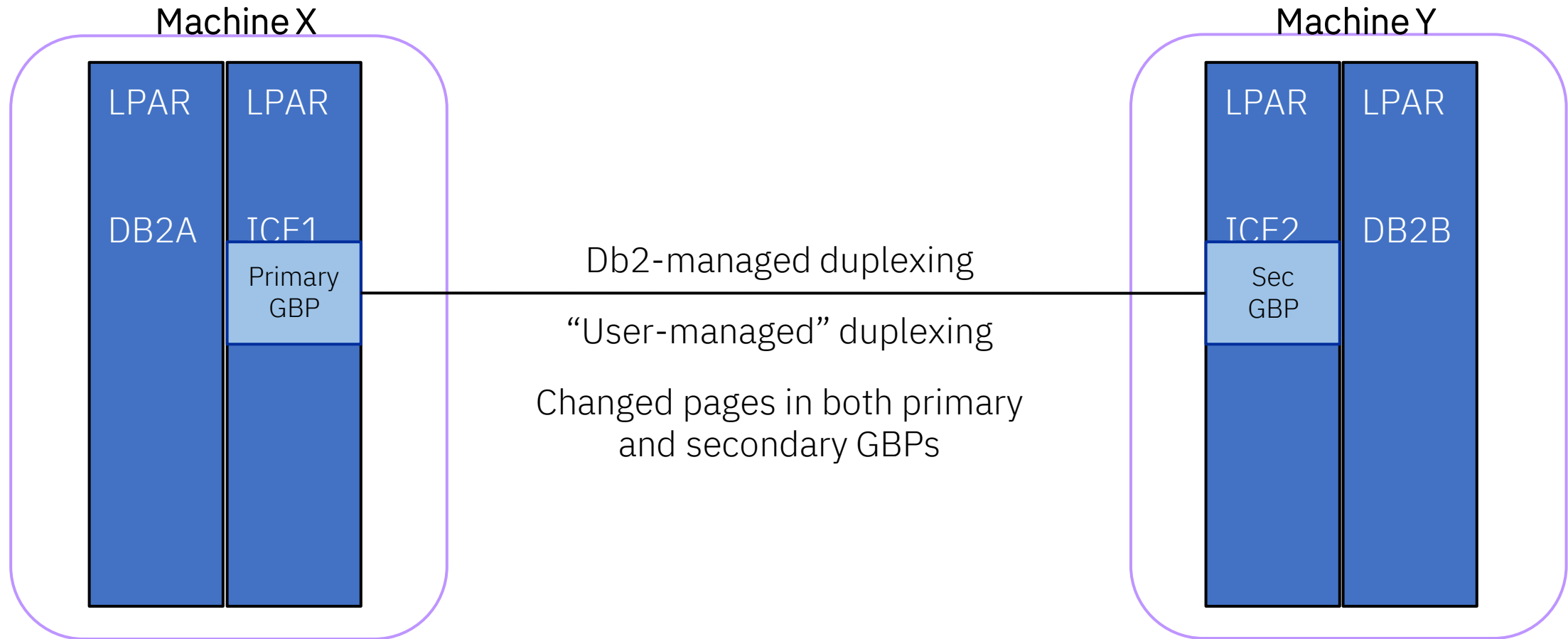


# More Db2 group buffer pool details

- Changed pages written to a GBP are externalized to disk
  - Castout process
  - GBP may be the only place the latest copy of the page exists before castout
- Duplexing GBPs – strongly recommended
  - Keeps GBP from being a single point of failure



# Duplexing Db2 group buffer pools



# GBP characteristics you control

## -ALTER GBPOOL(GBP $n$ )

- GBPCACHE \* – group buffer pool cache
  - YES – manage cross-invalidation (XI) and cache changed pages
  - NO – only manage cross-invalidation (XI)
- CLASST – class castout threshold (default 5%)
  - Trigger castout when changed pages in a class of objects exceeds this percentage
- GBPOOLT – group buffer pool castout threshold (default 30%)
  - Trigger castout when changed pages across any classes exceeds this percentage
- GBPCHKPT – GBP checkpoint interval (default 4 minutes)
- RATIO \* - ratio of directory entries to data elements (default 10:1)

# Group buffer pools - summary

# Group buffer pools (GBPs)

- Parallel Sysplex and Db2 data sharing
- Structures in coupling facility LPARs (CFs)
- Directory entries: track inter-Db2 interest
  - Cross-invalidation
- Data elements: changed pages
  - Pages written to GBP at commit...or before
- Castout to disk
- User-managed duplexing

# What's next?

1

Configuring

2

Monitoring

3

Tuning



# Thank you!

Victoria Felt  
[Victoria.felt@ibm.com](mailto:Victoria.felt@ibm.com)

Keziah Knopp  
[Keziah.knopp@ibm.com](mailto:Keziah.knopp@ibm.com)

Anna McKee  
[Anna.mckee@ibm.com](mailto:Anna.mckee@ibm.com)

# IBM