Einsatz und Realisierung von Datenbanksystemen

ERDB Übungsleitung

Alice Rey, Maximilian Bandle, Michael Jungmair

i3erdb@in.tum.de

Folien erstellt von Maximilian Bandle & Alexander Beischl

Organisatorisches
Disclaimer

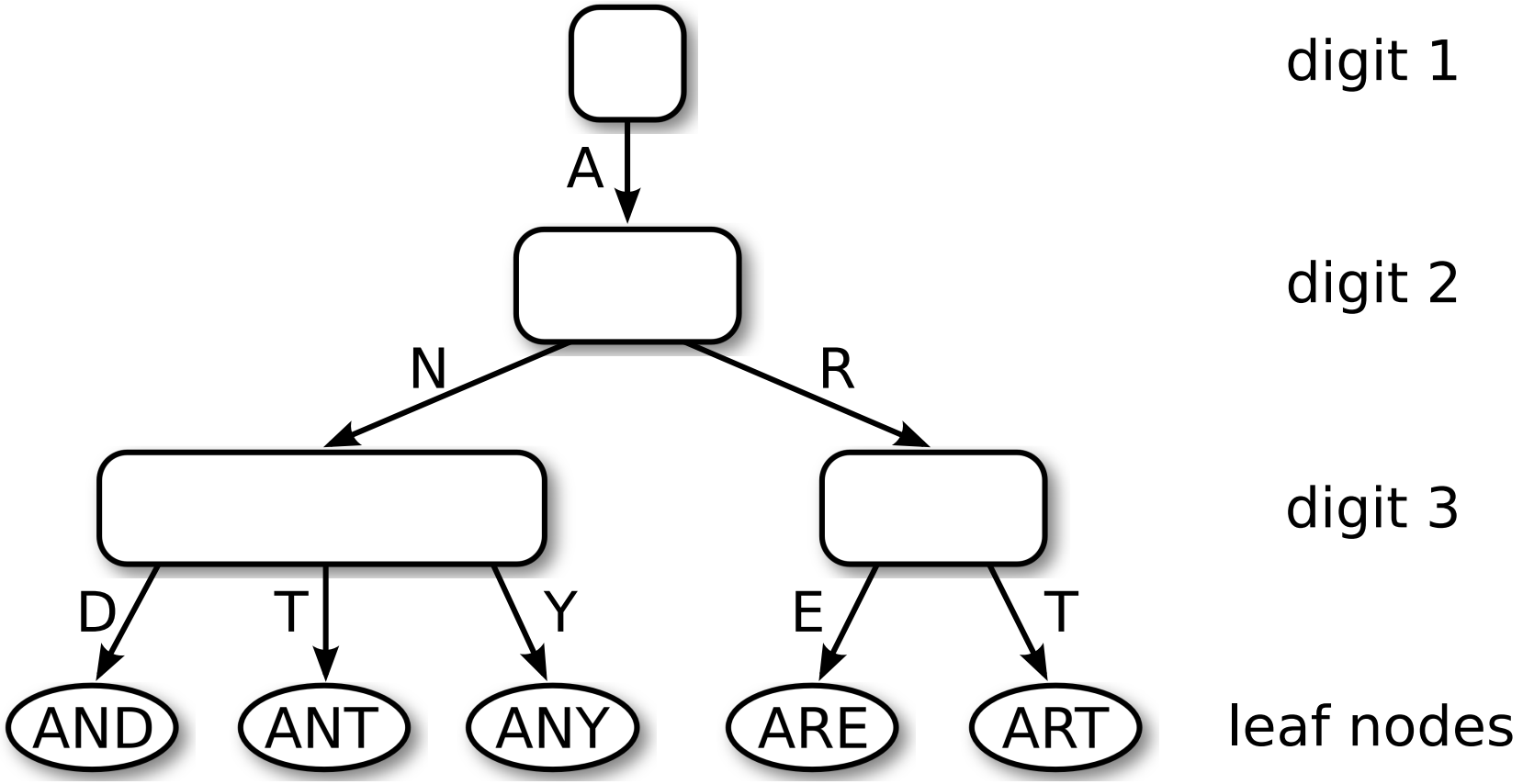
Die Folien werden von der Übungsleitung allen Tutoren zur Verfügung gestellt.

Sollte es Unstimmigkeiten zu den Vorlesungsfolien von Prof. Kemper geben, so sind die Folien aus der Vorlesung ausschlaggebend.

Falls Ihr einen Fehler oder eine Unstimmigkeit findet, schreibt an i3erdb@in.tum.de mit Angabe der Foliennummer.

Hauptspeicher-Datenbanken

Hauptspeicher-Datenbanken
ART-Tree (Adaptiver Radix-Baum)



Hauptspeicher-Datenbanken

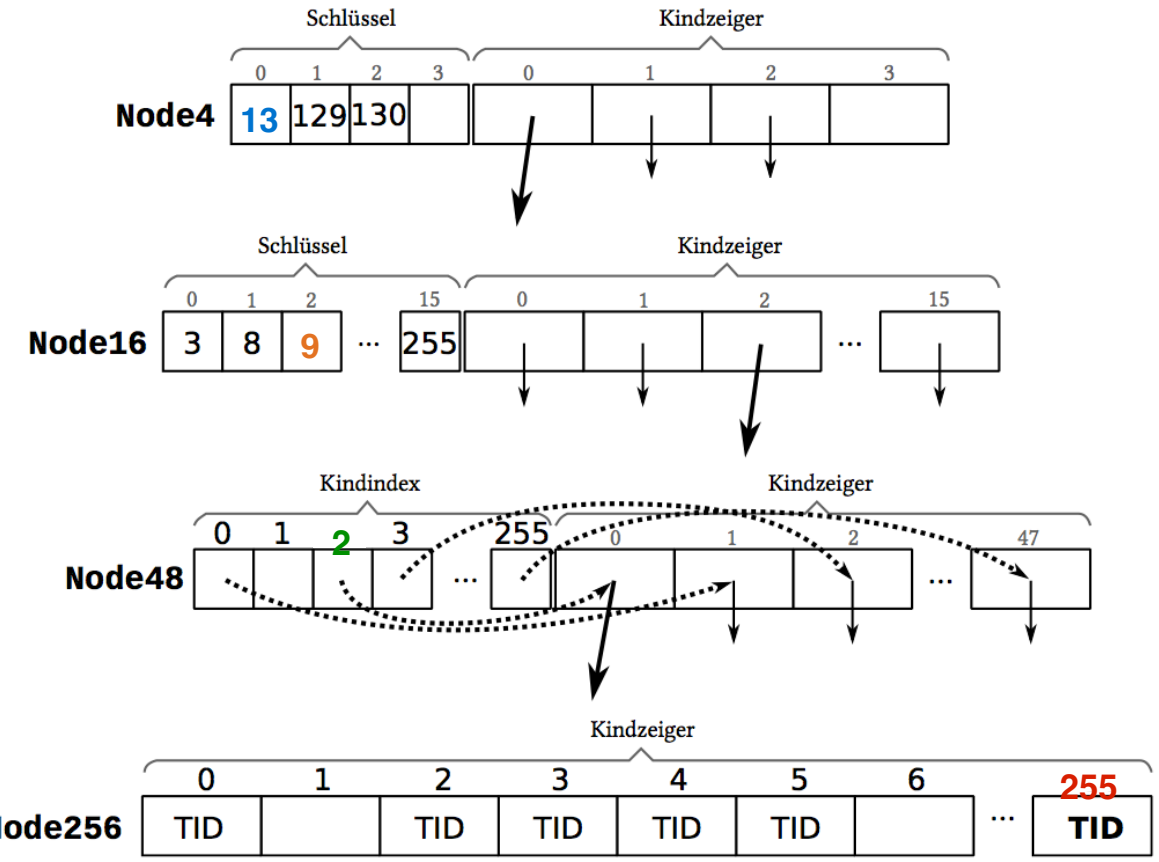
ART-Knotentypen

Integer Schlüssel
+218694399

Bit Repräsentation (32 bit, ohne Vorzeichen)
00001101 00001001 00000010 11111111

Byte-Repräsentation

13	9	2	255



Aufgabe 1

In (pseudo) C++ kann eine 'Row-Store-artige' Datenstruktur wie folgt angelegt werden:

```
struct Tuple {
  int MatrNr;
  RuntimeString Name;
  int Semester;
}
Tuple data[10000] = {};
```

Notieren Sie, wie die Daten in Form eines Column Stores gehalten werden können in (pseudo) C++.

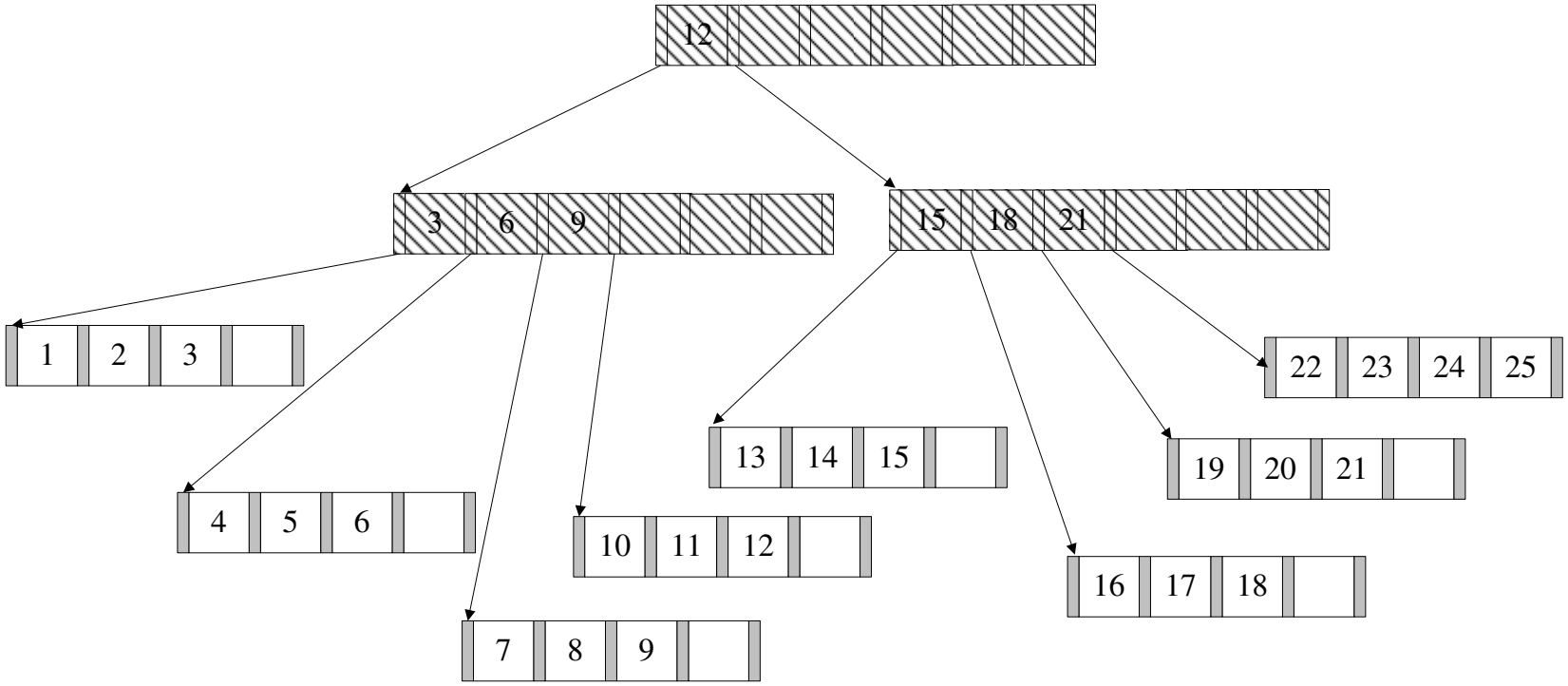
Erklären Sie Ihrem Tutor, welche Vor- und Nachteile Row- und Column Stores jeweils haben. Was würden Sie für Amazons Webseite verwenden? Was verwenden Sie für die Controlling Datenbank?

Aufgabe 2

Schätzen sie die Anzahl der Cache Misses die entstehen, wenn man 1001 32-bit Integer Werte (0-1000) in aufeinanderfolgender Reihenfolge in einen ART Baum einfügt. Wäre ein B+ Baum besser oder schlechter? Bei den Baumknoten müssen die Header nicht berücksichtigt werden, Pointer habe eine Größe von 64 bit.

Aufgabe 2

B+ Baum Beispiel



Aufgabe 3

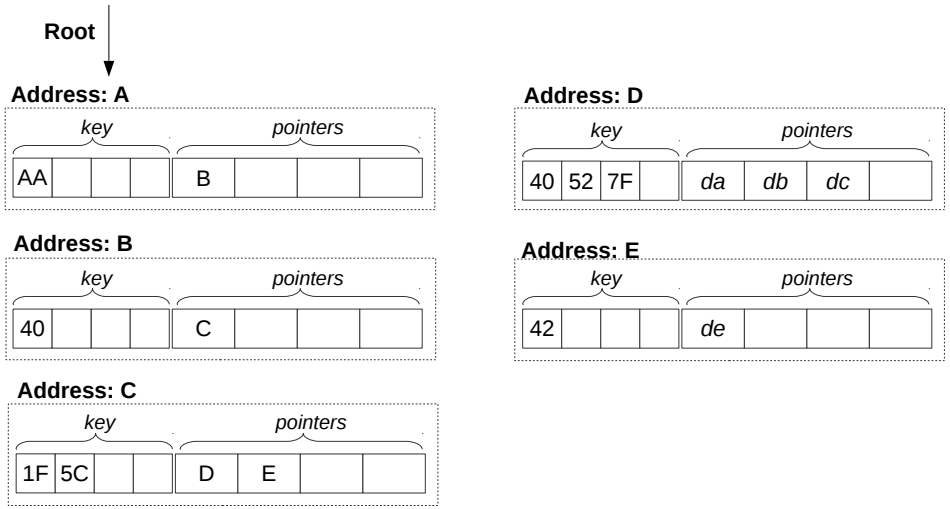


Abbildung 1: Knoten des ART (jeweils Node4)

In Abbildung 1 sehen Sie die Knoten eines ART Baums. Der Wurzelknoten liegt an Adresse `A`. Zeiger die mit `d` anfangen (z.B. da, db, ...) zeigen auf Daten. Suchschlüssel sind in den Aufgaben jeweils sowohl als Zahl z.B. 99, als auch hexadezimal codiert angegeben, z.B. der Wert 99 als 32 Bit Integer (0x00 0x00 0x00 0x63).

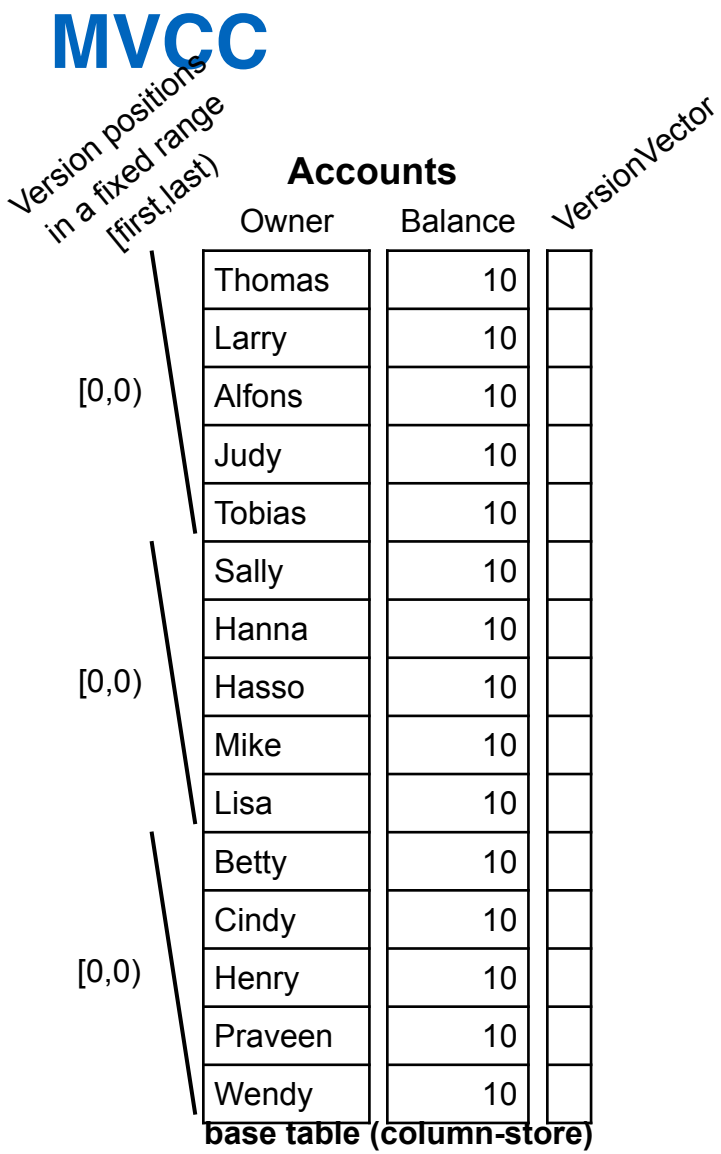
1) Beschreiben Sie kurz den Pfad durch den Baum für den 32-bit Suchschlüssel 2856344642 (0xAA 0x40 0x5C 0x42).
2) Welche dieser Suchschlüssel sind im Baum enthalten? 291 (0x00 0x00 0x01 0x23), 2856329024 (0xAA 0x40 0x1F 0x40), 2856329026 (0xAA 0x40 0x1F 0x42)
3) Beschreiben Sie kurz wie sich der Baum beim Einfügen des Schlüssels 2856352578 (0xAA 0x40 0x7B 0x42) verändert. Der Schlüssel soll auf den Wert an der Adresse `df` zeigen.

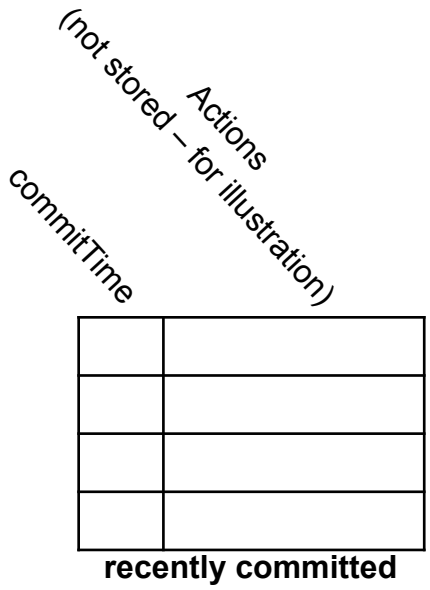
MVCC

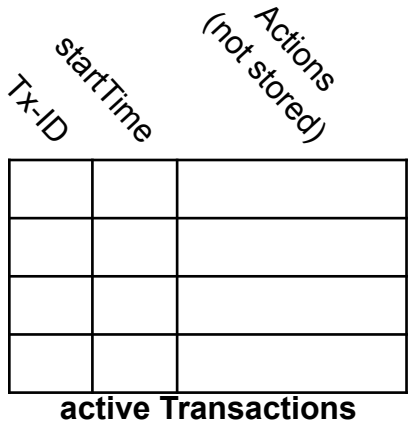
- **M**ulti**v**ersion **C**oncurrency **C**ontrol
- fein granulares Mehrversionsverfahren für die Synchronisation mehrerer paralleler TA
- sehr gut für unterbrechbare TA
- klassische DB nutzen das 2-Phasen-Sperrprotokoll

➡ zu schwerfällig für Hauptspeicher DBs

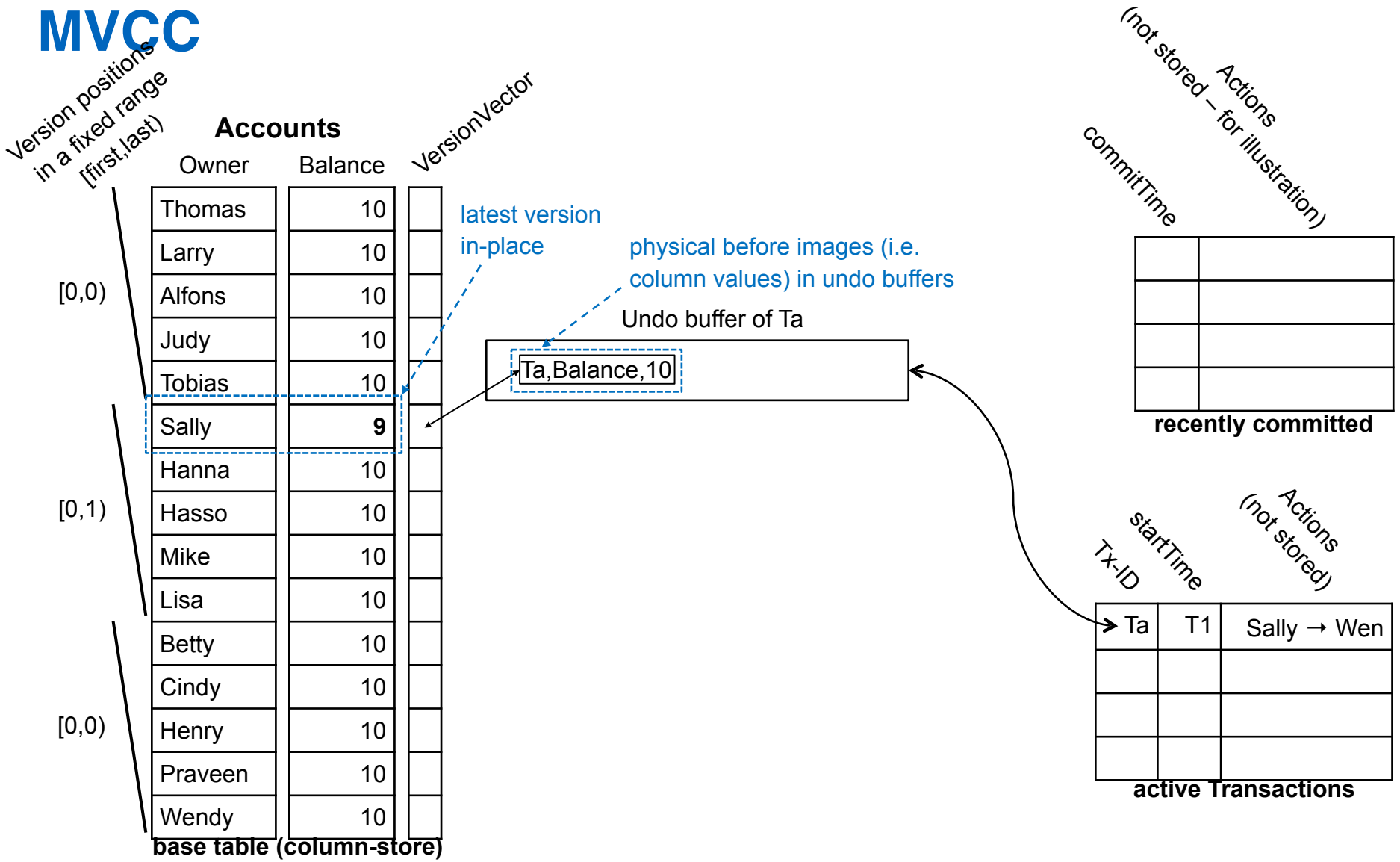
MVCC


Version positions in a fixed range [first,last)
Accounts
Owner
Balance
VersionVector
[0,0)
[0,0)
[0,0)
Thomas 10
Larry 10
Alfons 10
Judy 10
Tobias 10
Sally 10
Hanna 10
Hasso 10
Mike 10
Lisa 10
Betty 10
Cindy 10
Henry 10
Praveen 10
Wendy 10
base table (column-store)

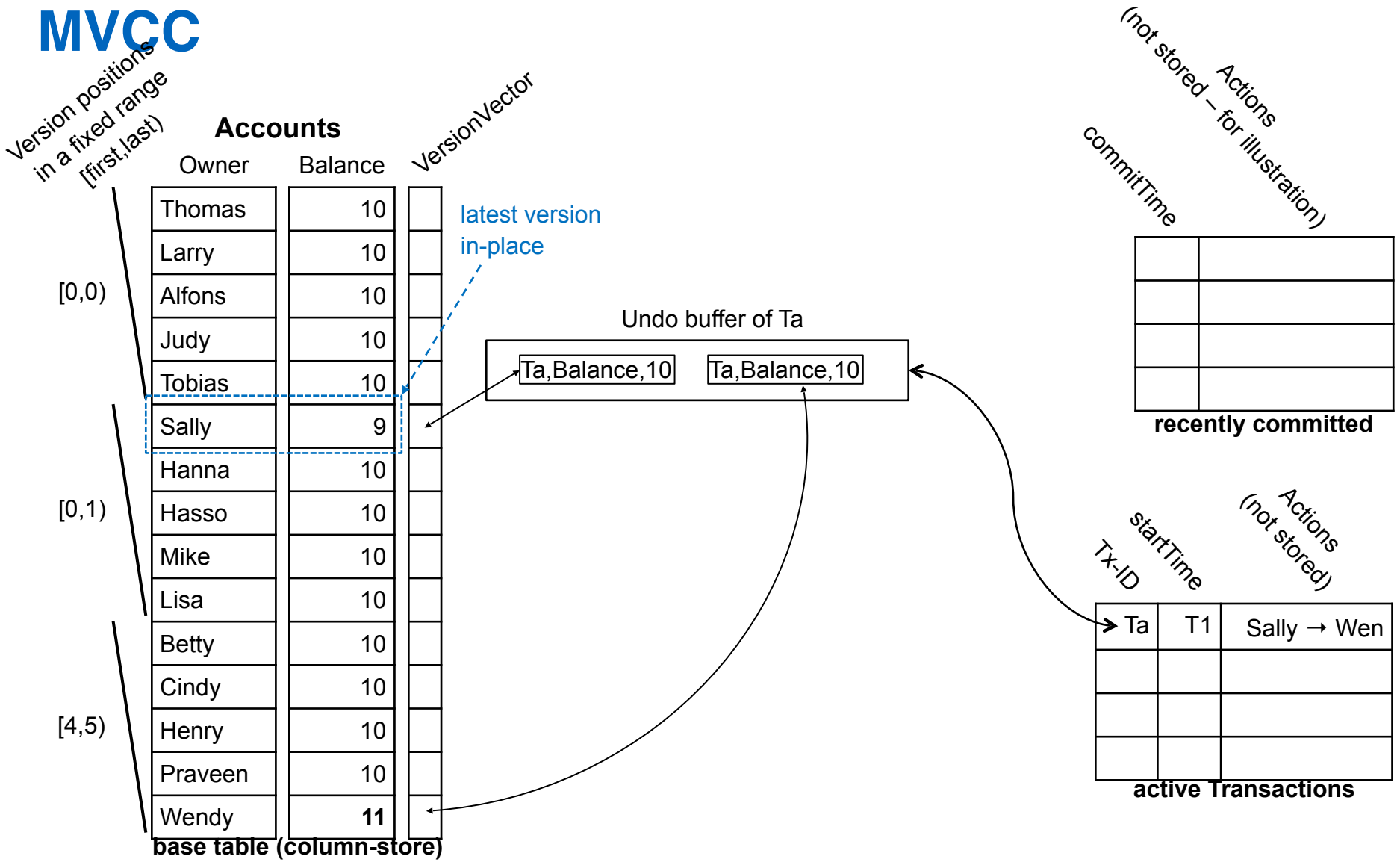

(not stored – for illustration)
Actions
commitTime
recently committed


Actions
(not stored)
startTime
Tx-ID
active Transactions

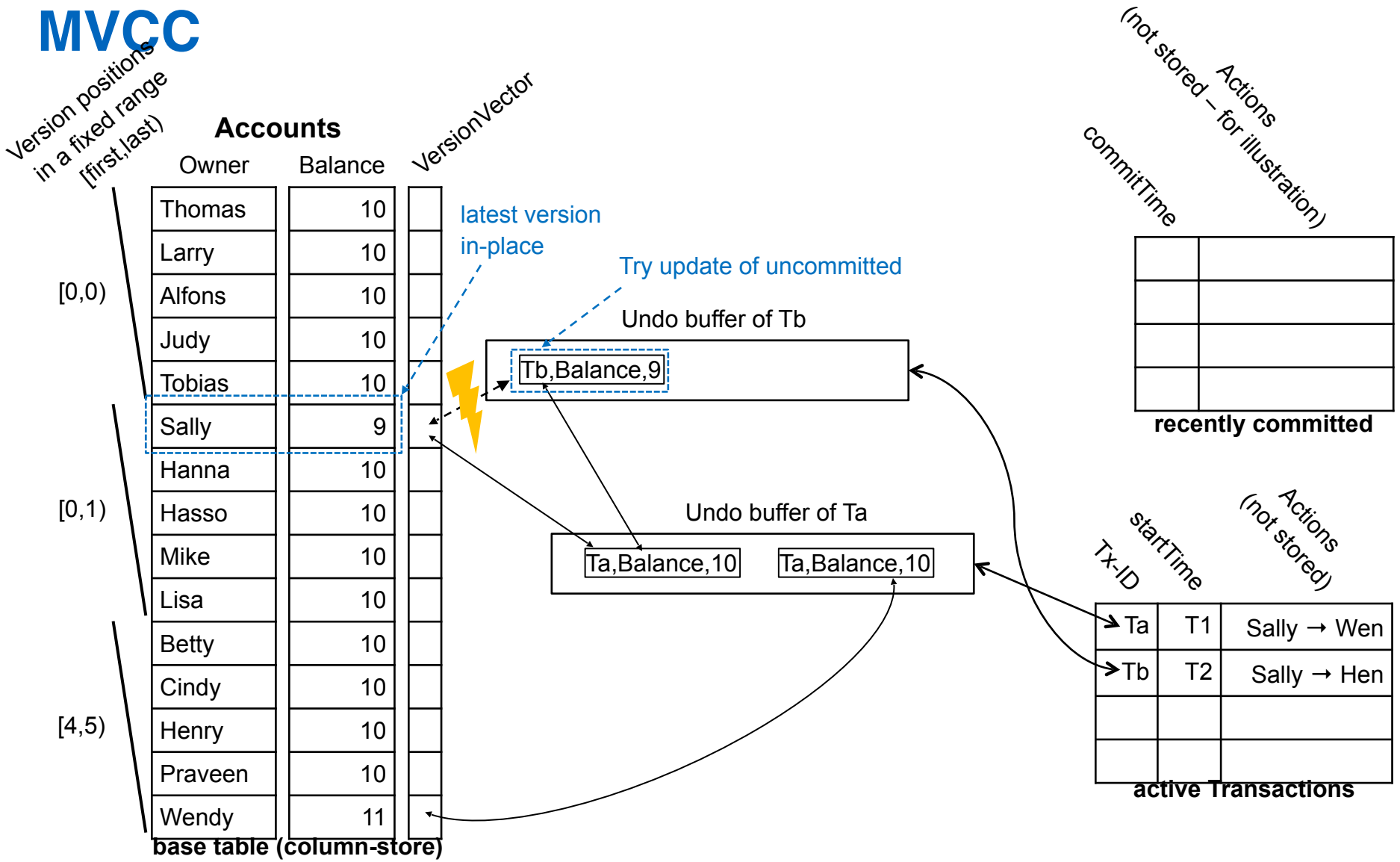
MVCC


Version positions in a fixed range [first,last)
Accounts
Owner
Balance
VersionVector
Thomas 10
Larry 10
Alfons 10
Judy 10
Tobias 10
Sally 9
Hanna 10
Hasso 10
Mike 10
Lisa 10
Betty 10
Cindy 10
Henry 10
Praveen 10
Wendy 10
[0,0)
[0,1)
[0,0)
base table (column-store)
latest version in-place
physical before images (i.e. column values) in undo buffers
Undo buffer of Ta
Ta,Balance,10
(not stored – for illustration)
Actions
commitTime
recently committed
Tx-ID
startTime
Actions (not stored)
Ta
T1
Sally → Wen
active Transactions

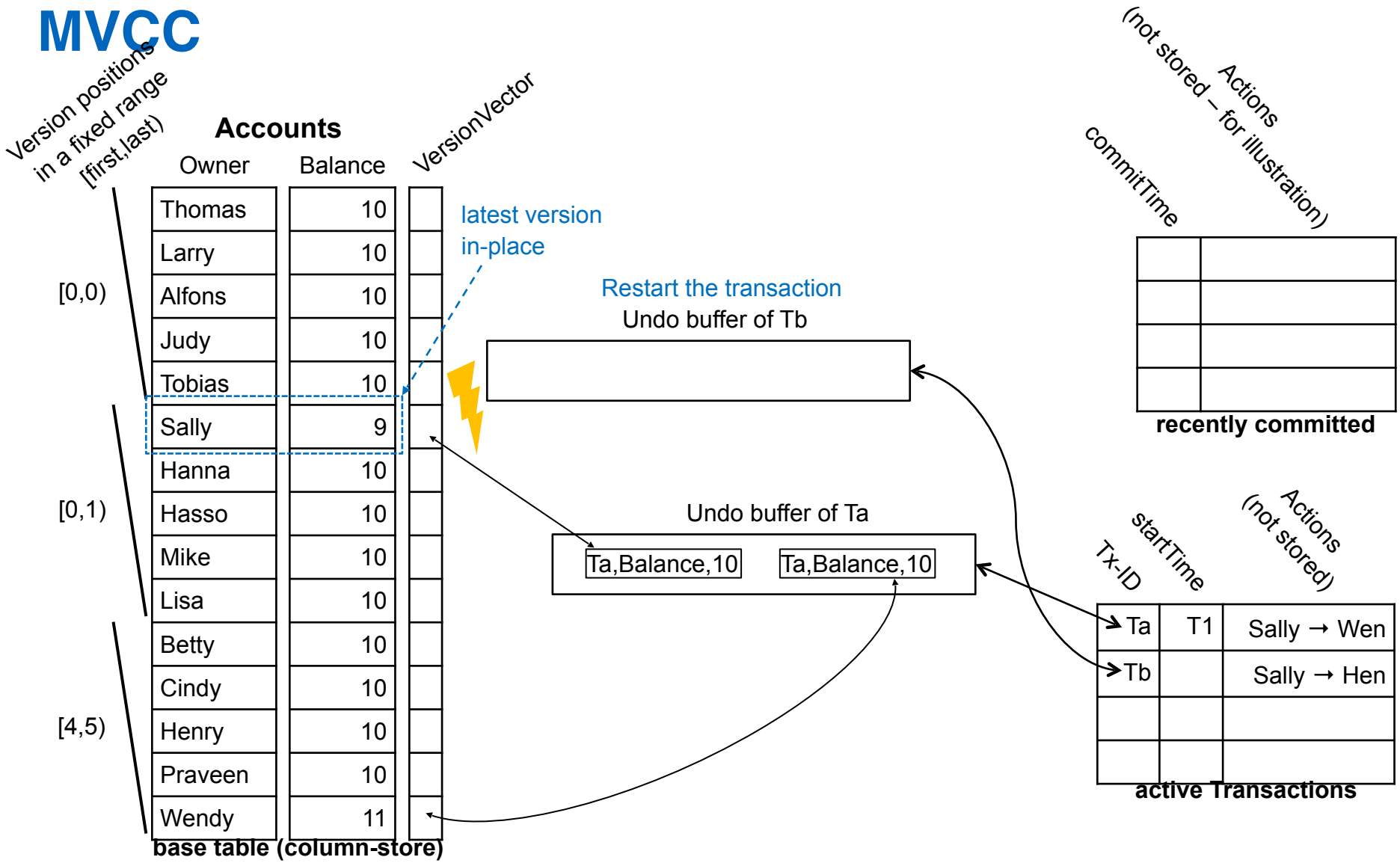
MVCC


Version positions in a fixed range [first,last)
Accounts
Owner
Balance
VersionVector
Thomas 10
Larry 10
Alfons 10
Judy 10
Tobias 10
Sally 9
Hanna 10
Hasso 10
Mike 10
Lisa 10
Betty 10
Cindy 10
Henry 10
Praveen 10
Wendy 11
[0,0)
[0,1)
[4,5)
base table (column-store)
latest version in-place
Undo buffer of Ta
Ta,Balance,10
Ta,Balance,10
(not stored – for illustration)
Actions
commitTime
recently committed
Actions (not stored)
startTime
Tx-ID
Ta
T1
Sally → Wen
active Transactions

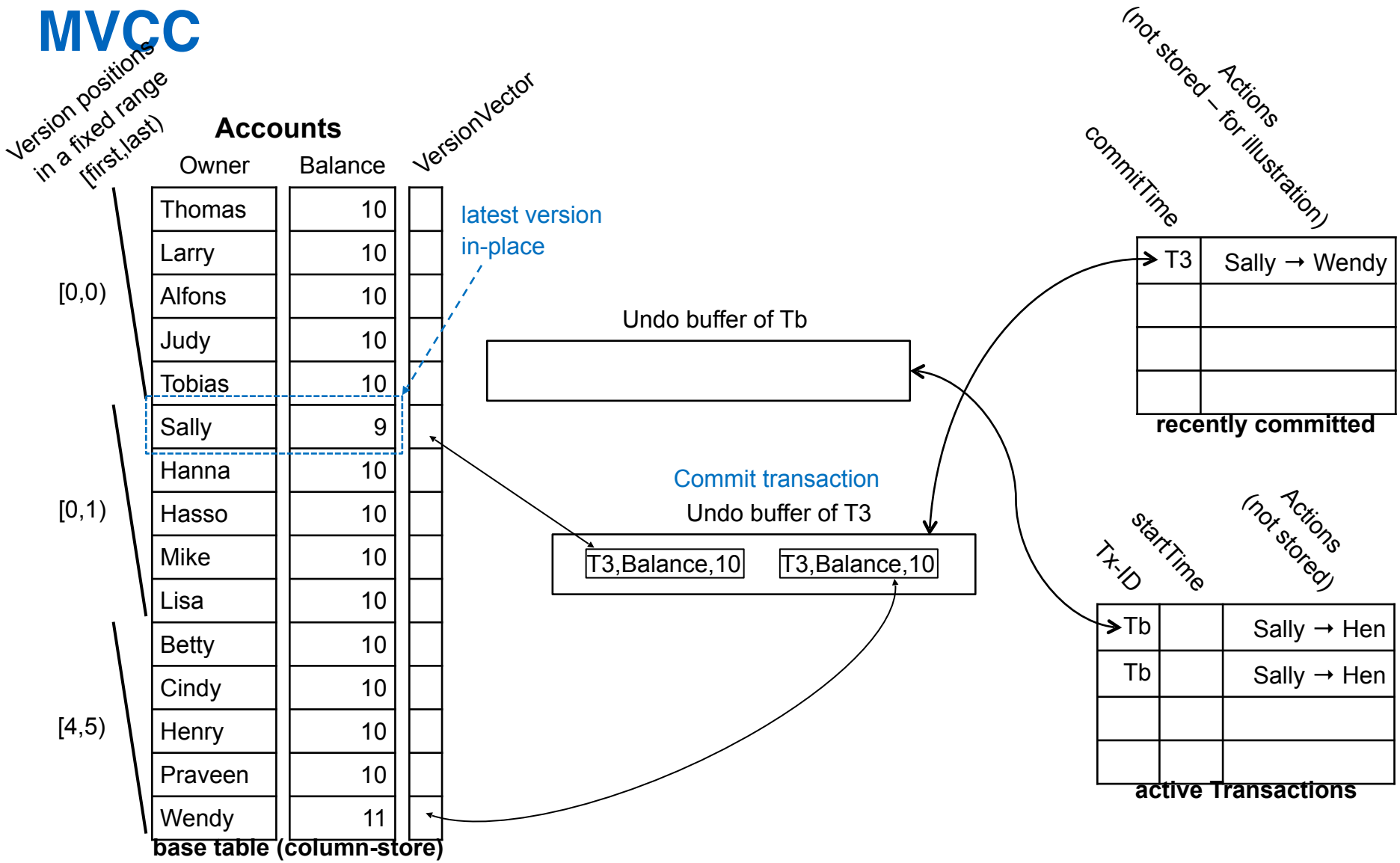
MVCC


Version positions in a fixed range [first,last)
Accounts
Owner
Balance
VersionVector
[0,0)
[0,1)
[4,5)
Thomas 10
Larry 10
Alfons 10
Judy 10
Tobias 10
Sally 9
Hanna 10
Hasso 10
Mike 10
Lisa 10
Betty 10
Cindy 10
Henry 10
Praveen 10
Wendy 11
base table (column-store)
latest version in-place
Try update of uncommitted
Undo buffer of Tb
Tb,Balance,9
Undo buffer of Ta
Ta,Balance,10
Ta,Balance,10
commitTime
Actions (not stored – for illustration)
recently committed
Tx-ID
startTime
Actions (not stored)
Ta T1 Sally → Wen
Tb T2 Sally → Hen
active Transactions

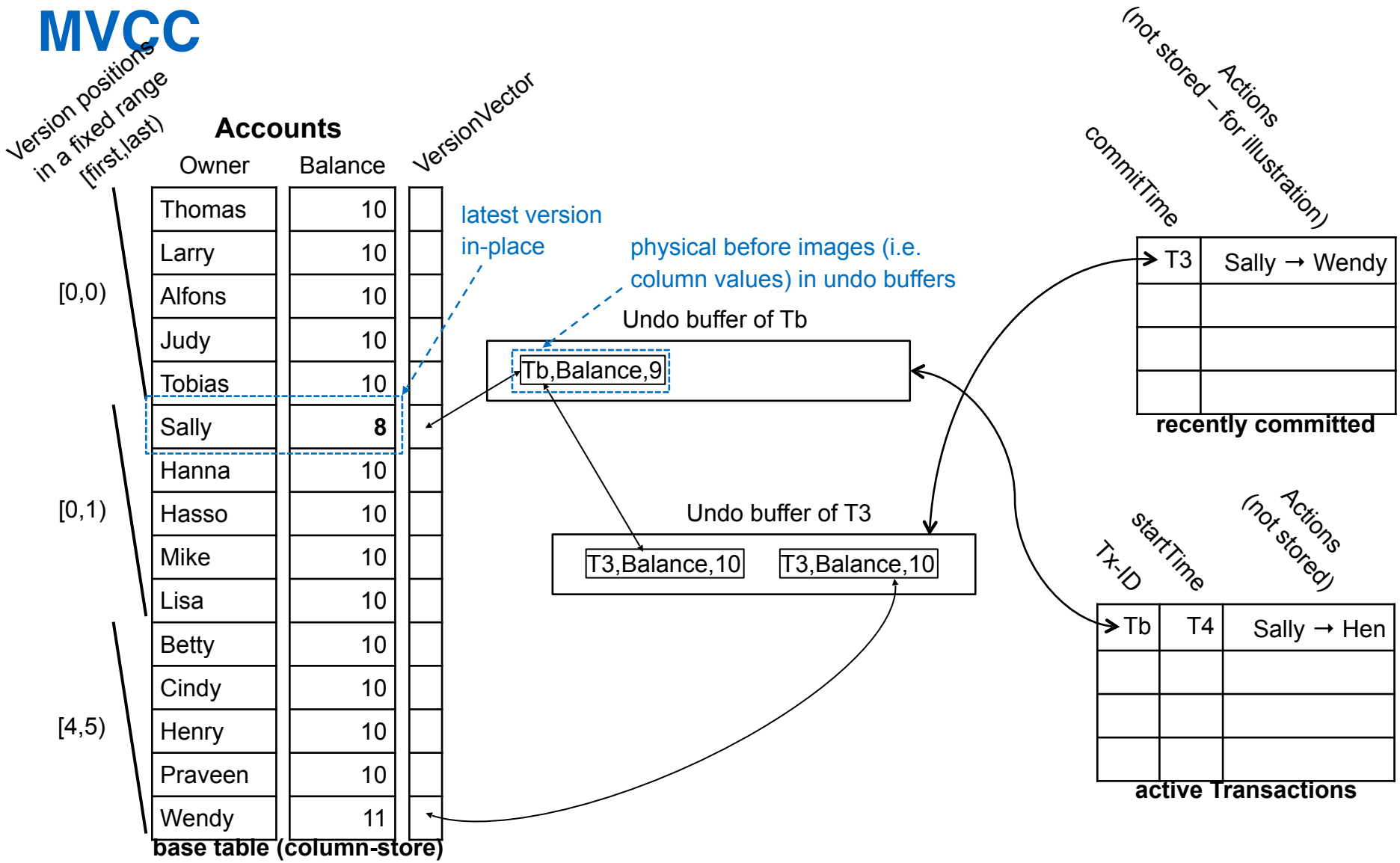
MVCC


Version positions in a fixed range [first,last)
Accounts
Owner
Balance
VersionVector
Thomas 10
Larry 10
Alfons 10
Judy 10
Tobias 10
Sally 9
Hanna 10
Hasso 10
Mike 10
Lisa 10
Betty 10
Cindy 10
Henry 10
Praveen 10
Wendy 11
[0,0)
[0,1)
[4,5)
base table (column-store)
latest version in-place
Restart the transaction
Undo buffer of Tb
Undo buffer of Ta
Ta,Balance,10
Ta,Balance,10
(not stored – for illustration)
Actions
commitTime
recently committed
Tx-ID
startTime
Actions (not stored)
Ta T1 Sally → Wen
Tb Sally → Hen
active Transactions

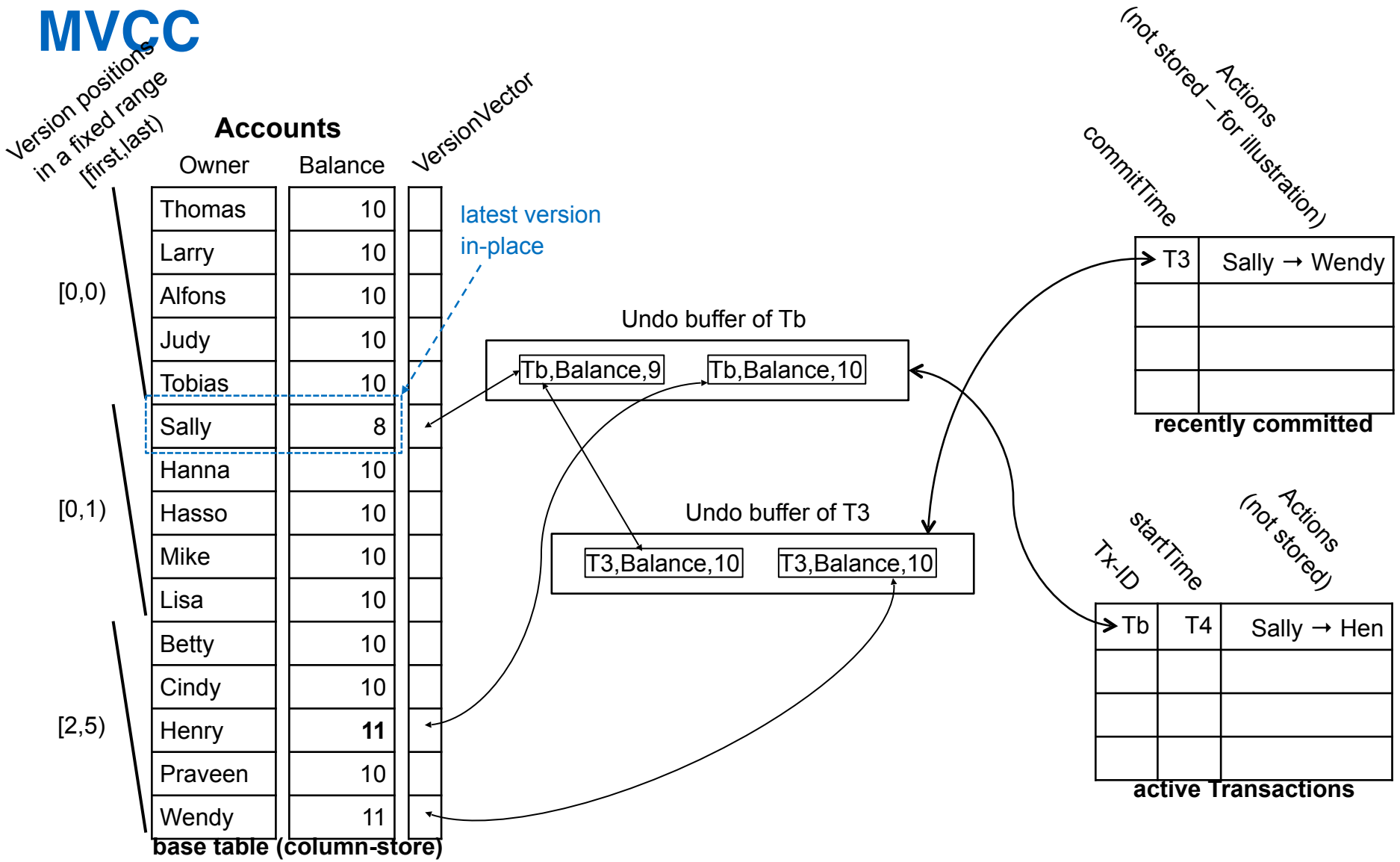
MVCC


Version positions in a fixed range [first,last)
Accounts
Owner
Balance
VersionVector
Thomas 10
Larry 10
Alfons 10
Judy 10
Tobias 10
Sally 9
Hanna 10
Hasso 10
Mike 10
Lisa 10
Betty 10
Cindy 10
Henry 10
Praveen 10
Wendy 11
[0,0)
[0,1)
[4,5)
base table (column-store)
latest version in-place
Undo buffer of Tb
Commit transaction
Undo buffer of T3
T3,Balance,10
T3,Balance,10
commitTime
Actions (not stored – for illustration)
T3
Sally → Wendy
recently committed
Tx-ID
startTime
Actions (not stored)
Tb
Sally → Hen
Tb
Sally → Hen
active Transactions

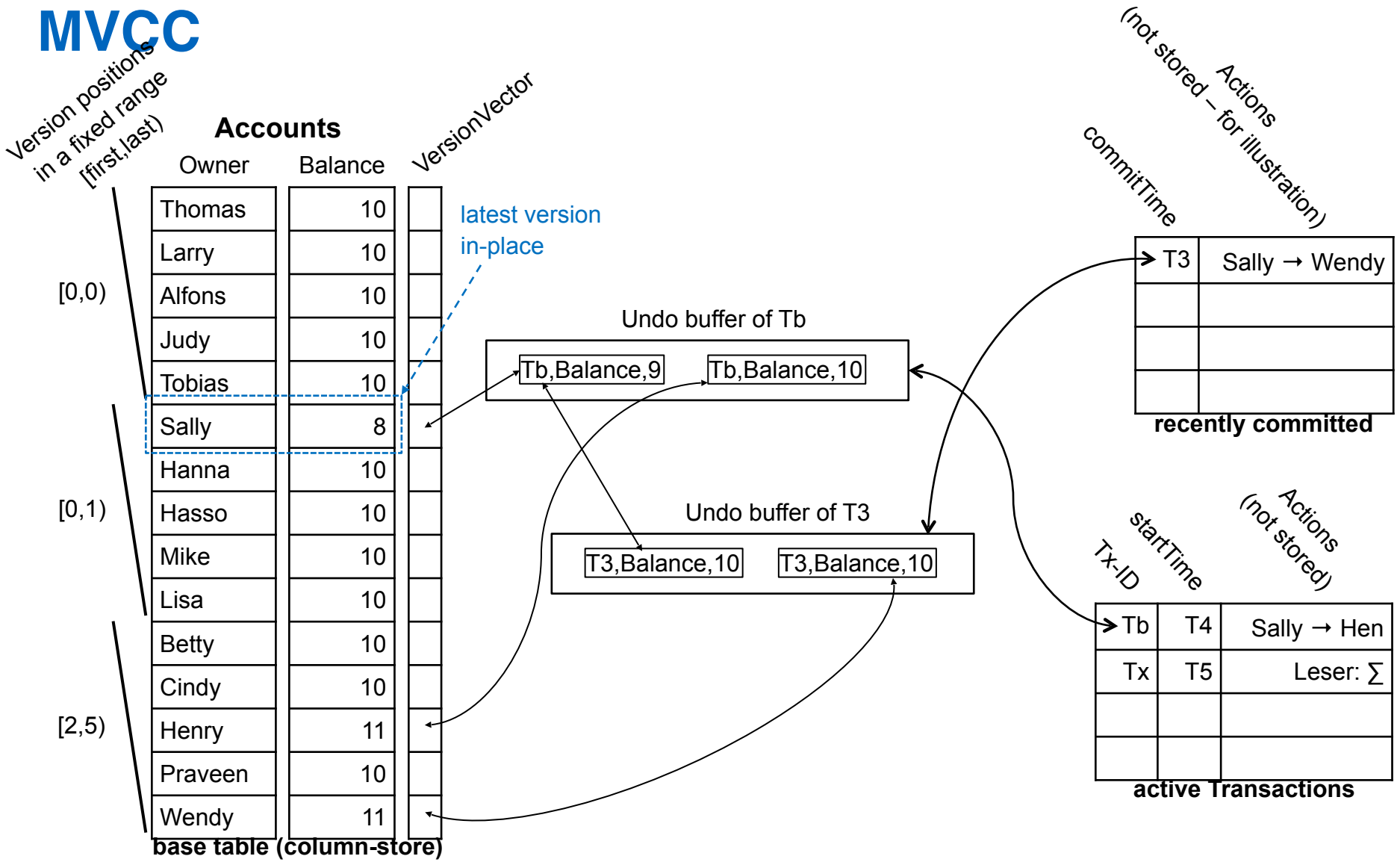
MVCC


Version positions in a fixed range [first,last)
Accounts
Owner
Balance
VersionVector
Thomas 10
Larry 10
Alfons 10
Judy 10
Tobias 10
Sally 8
Hanna 10
Hasso 10
Mike 10
Lisa 10
Betty 10
Cindy 10
Henry 10
Praveen 10
Wendy 11
[0,0)
[0,1)
[4,5)
base table (column-store)
latest version in-place
physical before images (i.e. column values) in undo buffers
Undo buffer of Tb
Tb,Balance,9
Undo buffer of T3
T3,Balance,10
T3,Balance,10
commitTime
Actions (not stored – for illustration)
T3
Sally → Wendy
recently committed
Tx-ID
startTime
Actions (not stored)
Tb
T4
Sally → Hen
active Transactions

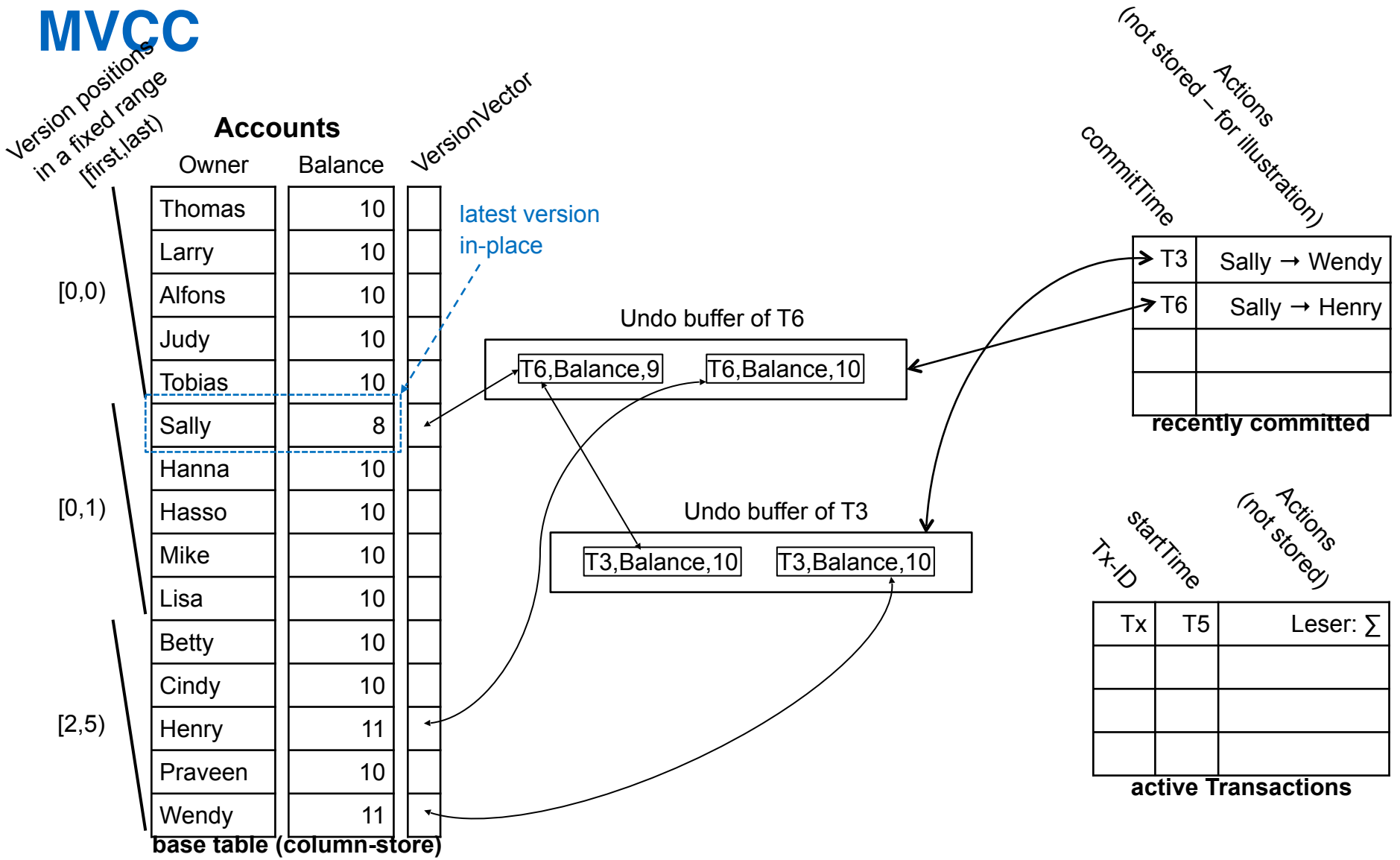
MVCC


Version positions in a fixed range [first,last)
Accounts
Owner
Balance
VersionVector
latest version in-place
[0,0)
[0,1)
[2,5)
Thomas 10
Larry 10
Alfons 10
Judy 10
Tobias 10
Sally 8
Hanna 10
Hasso 10
Mike 10
Lisa 10
Betty 10
Cindy 10
Henry 11
Praveen 10
Wendy 11
base table (column-store)
Undo buffer of Tb
Tb,Balance,9
Tb,Balance,10
Undo buffer of T3
T3,Balance,10
T3,Balance,10
commitTime
Actions (not stored – for illustration)
T3
Sally → Wendy
recently committed
Tx-ID
startTime
Actions (not stored)
Tb
T4
Sally → Hen
active Transactions

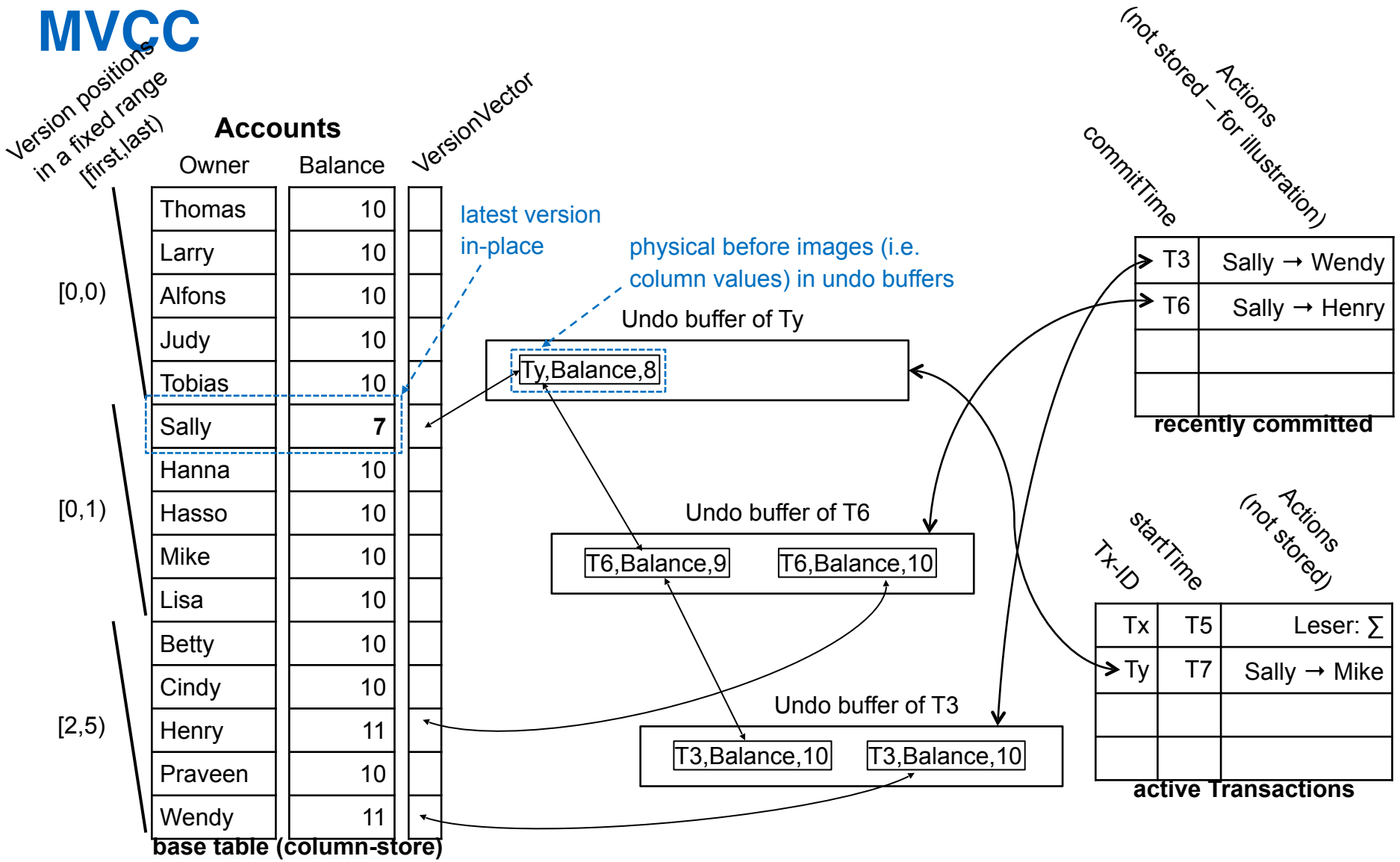
MVCC


Version positions in a fixed range [first,last)
Accounts
Owner
Balance
VersionVector
latest version in-place
[0,0)
[0,1)
[2,5)
Thomas 10
Larry 10
Alfons 10
Judy 10
Tobias 10
Sally 8
Hanna 10
Hasso 10
Mike 10
Lisa 10
Betty 10
Cindy 10
Henry 11
Praveen 10
Wendy 11
base table (column-store)
Undo buffer of Tb
Tb,Balance,9
Tb,Balance,10
Undo buffer of T3
T3,Balance,10
T3,Balance,10
commitTime
Actions (not stored – for illustration)
T3
Sally → Wendy
recently committed
Tx-ID
startTime
Actions (not stored)
Tb
T4
Sally → Hen
Tx
T5
Leser: ∑
active Transactions

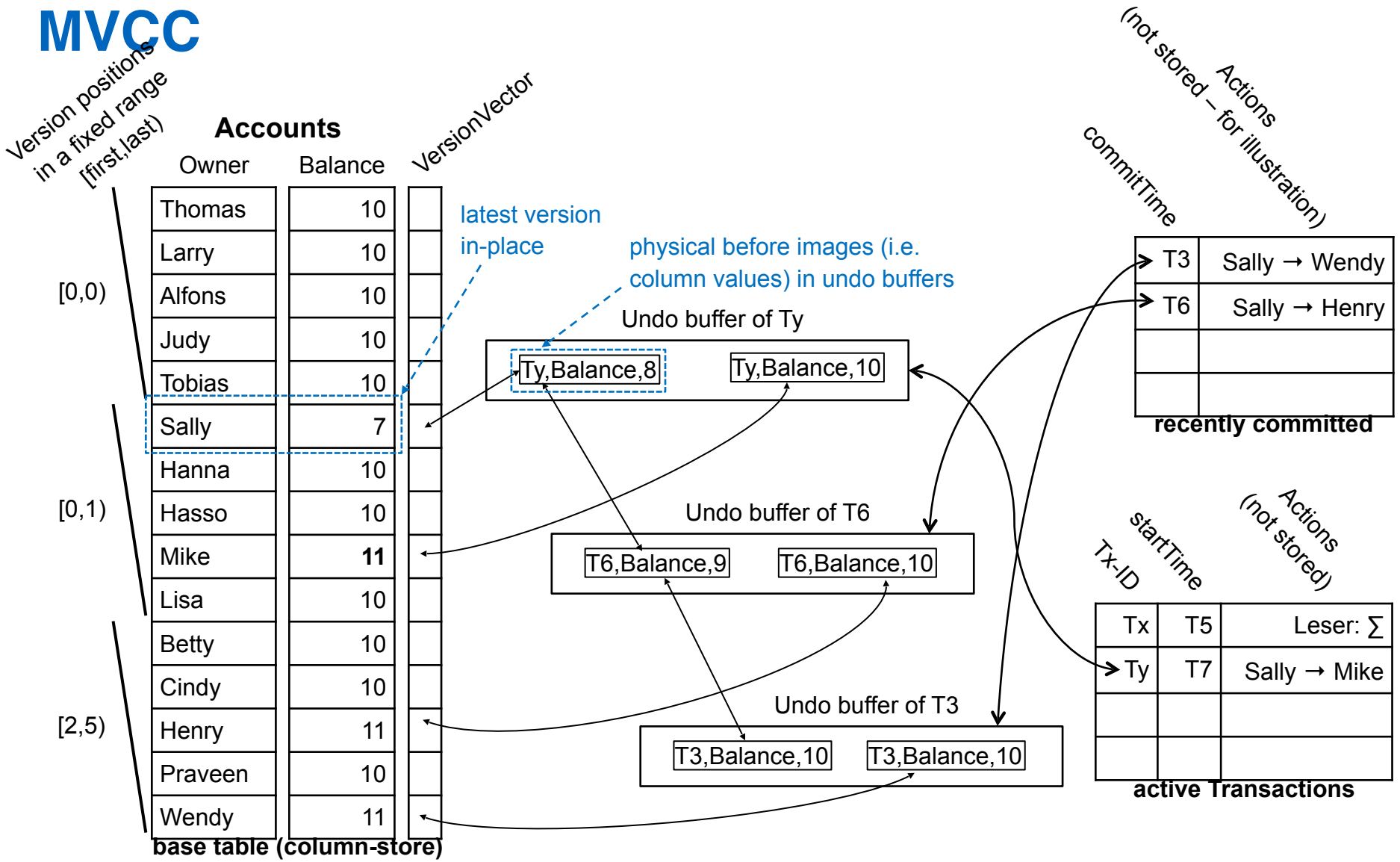
MVCC


Version positions in a fixed range [first,last)
Accounts
Owner
Balance
VersionVector
latest version in-place
[0,0)
[0,1)
[2,5)
Thomas 10
Larry 10
Alfons 10
Judy 10
Tobias 10
Sally 8
Hanna 10
Hasso 10
Mike 10
Lisa 10
Betty 10
Cindy 10
Henry 11
Praveen 10
Wendy 11
base table (column-store)
Undo buffer of T6
T6,Balance,9
T6,Balance,10
Undo buffer of T3
T3,Balance,10
T3,Balance,10
commitTime
Actions (not stored – for illustration)
T3 Sally → Wendy
T6 Sally → Henry
recently committed
Tx-ID
startTime
Actions (not stored)
Tx T5 Leser: ∑
active Transactions

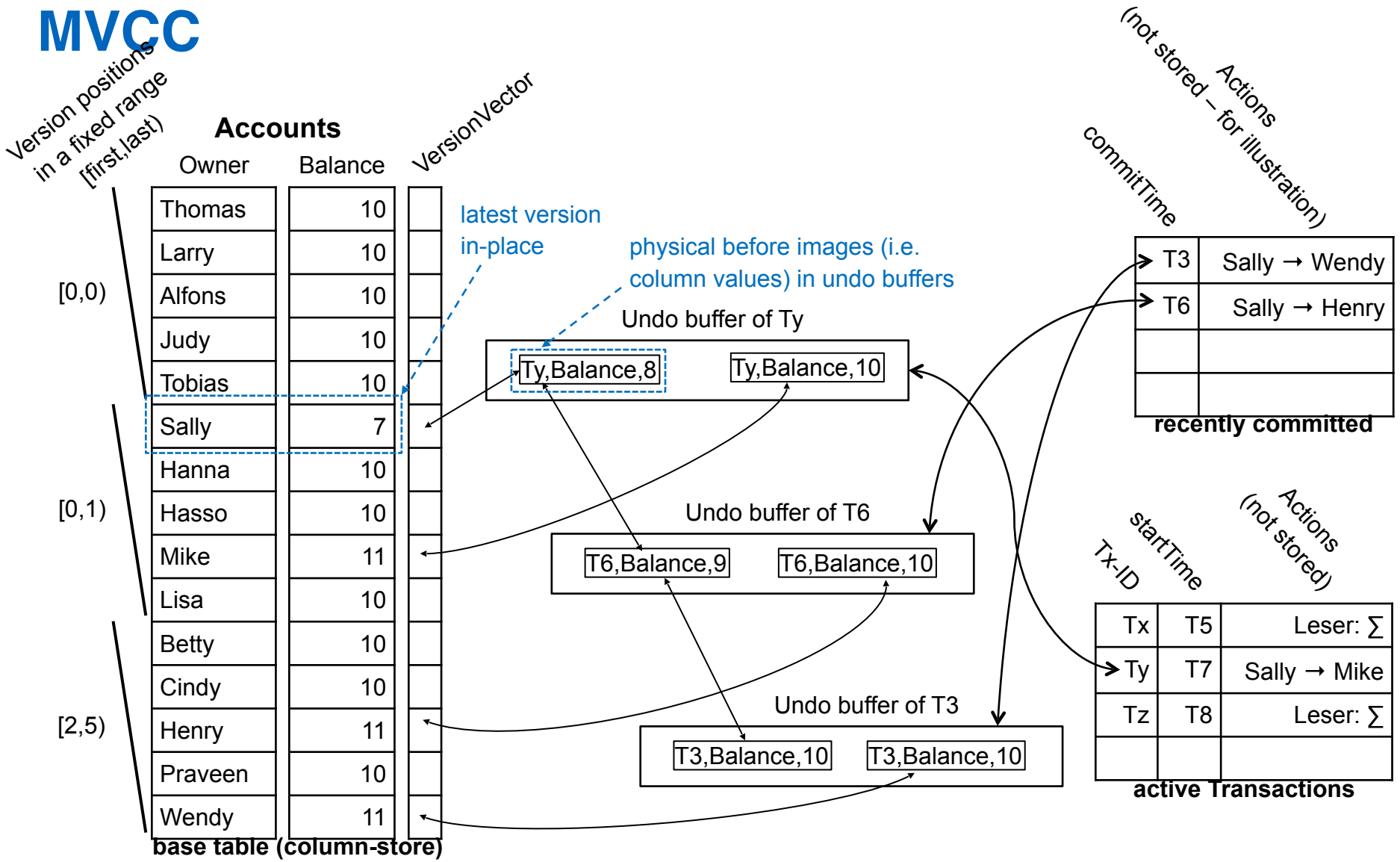
MVCC


Version positions in a fixed range [first,last)
Accounts
Owner
Balance
VersionVector
[0,0)
[0,1)
[2,5)
Thomas 10
Larry 10
Alfons 10
Judy 10
Tobias 10
Sally 7
Hanna 10
Hasso 10
Mike 10
Lisa 10
Betty 10
Cindy 10
Henry 11
Praveen 10
Wendy 11
base table (column-store)
latest version in-place
physical before images (i.e. column values) in undo buffers
Undo buffer of Ty
Ty,Balance,8
Undo buffer of T6
T6,Balance,9
T6,Balance,10
Undo buffer of T3
T3,Balance,10
T3,Balance,10
commitTime
Actions (not stored – for illustration)
T3 Sally → Wendy
T6 Sally → Henry
recently committed
Tx-ID
startTime
Actions (not stored)
Tx T5 Leser: ∑
Ty T7 Sally → Mike
active Transactions

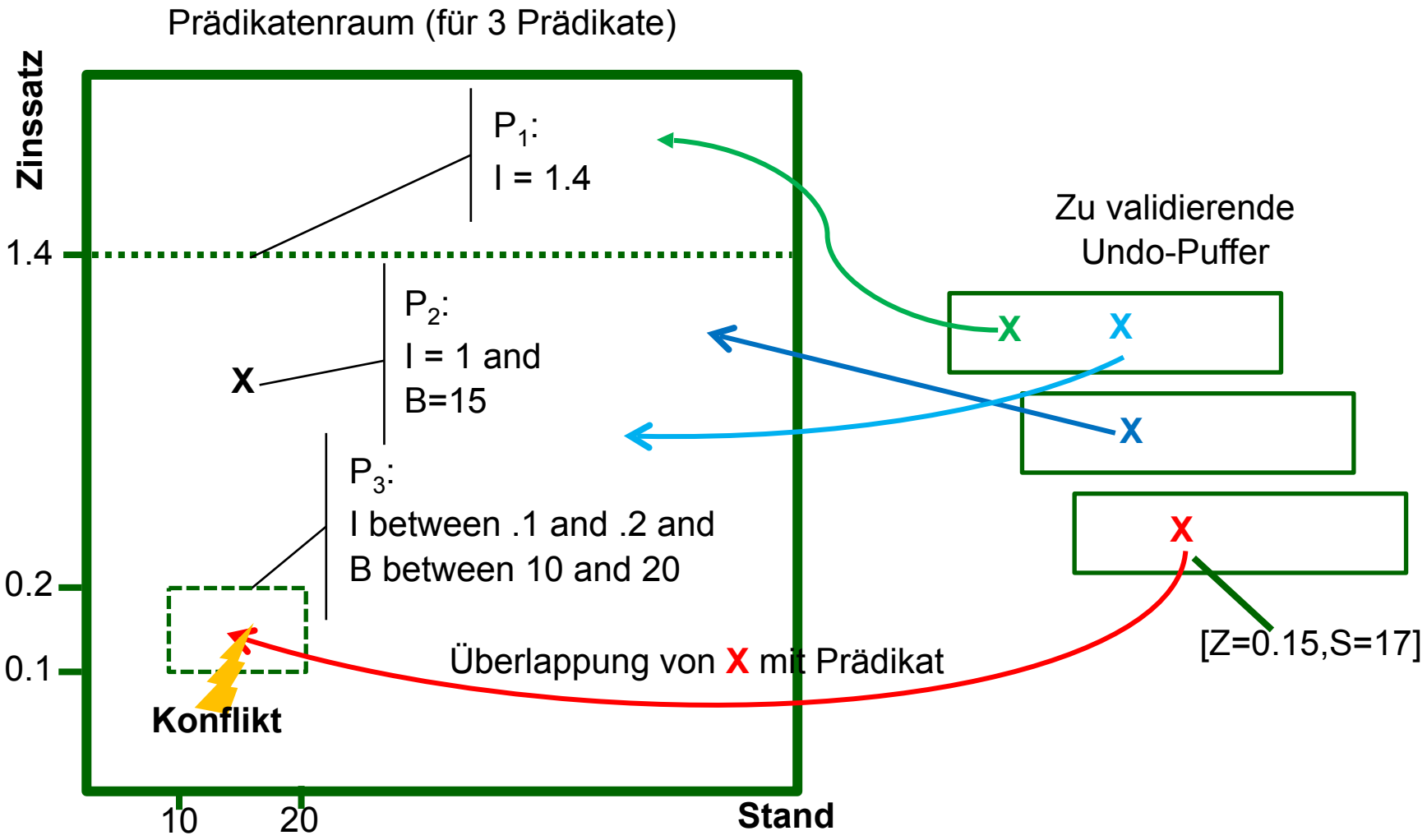
MVCC


Version positions in a fixed range [first,last)
Accounts
Owner
Balance
VersionVector
[0,0)
[0,1)
[2,5)
Thomas 10
Larry 10
Alfons 10
Judy 10
Tobias 10
Sally 7
Hanna 10
Hasso 10
Mike 11
Lisa 10
Betty 10
Cindy 10
Henry 11
Praveen 10
Wendy 11
base table (column-store)
latest version in-place
physical before images (i.e. column values) in undo buffers
Undo buffer of Ty
Ty,Balance,8
Ty,Balance,10
Undo buffer of T6
T6,Balance,9
T6,Balance,10
Undo buffer of T3
T3,Balance,10
T3,Balance,10
commitTime
Actions (not stored – for illustration)
T3 Sally → Wendy
T6 Sally → Henry
recently committed
Tx-ID
startTime
Actions (not stored)
Tx T5 Leser: ∑
Ty T7 Sally → Mike
active Transactions

MVCC


Version positions in a fixed range [first,last)
Accounts
Owner
Balance
VersionVector
Thomas 10
Larry 10
Alfons 10
Judy 10
Tobias 10
Sally 7
Hanna 10
Hasso 10
Mike 11
Lisa 10
Betty 10
Cindy 10
Henry 11
Praveen 10
Wendy 11
[0,0)
[0,1)
[2,5)
base table (column-store)
latest version in-place
physical before images (i.e. column values) in undo buffers
Undo buffer of Ty
Ty,Balance,8
Ty,Balance,10
Undo buffer of T6
T6,Balance,9
T6,Balance,10
Undo buffer of T3
T3,Balance,10
T3,Balance,10
commitTime
Actions (not stored – for illustration)
T3 Sally → Wendy
T6 Sally → Henry
recently committed
Tx-ID
startTime
Actions (not stored)
Tx T5 Leser: ∑
Ty T7 Sally → Mike
Tz T8 Leser: ∑
active Transactions

Precision Locking



MVCC mit Precision Locking

- Lesende Anfragen sind **immer erlaubt**: kein Precision Locking
- Falls schreibende Anfrage: **Überlappender Prädikatbereich**?
- Falls ja, dann **BOT** und **commit-Reihenfolge** beachten

Aufgabe 4

T1: insert into foo (select Note from Noten where MatrNr=12345)

T2: insert into bar (select count(*) from Noten where Note<1.5)

T3: insert into Noten(MatrNr,Note) values (54321, 3.0)

T4: update Noten set Note=1.4 where MatrNr=32154

T5: insert into Noten(MatrNr,Note) values (54321, 1.3)

T6: update Noten set Note=1.6 where MatrNr=12345

Analysieren Sie, ob die folgenden Historien unter dem MVCC Model, wie in der Vorlesung vorgestellt, auftreten können. Jede Historie steht für sich selbst und startet jeweils von einem ursprünglichen Datenzustand. Die Buchstaben innerhalb der Klammer entsprechen dabei jeweils den Tupeln auf die zugegriffen wird. Wenn in T2 z.B. drei Werte das 'Prädikat Note<1.5' erfüllen, gäbe es entsprechend drei $r(...)$ Einträge auf die jeweiligen Tupel.

H1 (T1 und T3): $bot_1, r_1(A), bot_3, w_3(B), w_1(C), commit_1, commit_3$

H2 (T2 und T3): $bot_2, r_2(A), bot_3, w_3(B), r_2(C), w_2(D), commit_2, commit_3$

H8 (T2 und T3): $bot_2, r_2(A), bot_3, w_3(B), r_2(C), commit_3, w_2(D), commit_2$

H3 (T2 und T4): $bot_2, r_2(A), r_2(B), bot_4, r_4(B), w_4(B), r_2(C), w_2(D), commit_2, commit_4$

H5 (T2 und T4): $bot_2, r_2(A), bot_4, r_4(B), w_4(B), r_2(C), commit_4, w_2(D), commit_2$

H4 (T1 und T6): $bot_1, r_1(B), bot_6, r_6(B), w_6(B), w_1(C), commit_1, commit_6$

H6 (T1 und T6): $bot_1, r_1(B), bot_6, r_6(B), w_6(B), commit_6, w_1(C), commit_1$

H7 (T2 und T5): $bot_2, r_2(A), bot_5, w_5(D), commit_5, r_2(D), w_2(E), commit_2$

H9 (T2 und T5): $bot_2, r_2(A), bot_5, w_5(B), r_2(C), commit_5, w_2(D), commit_2$

Gruppenaufgabe 5

Name	verfügbar	Versionsvektor
House	ja	-
Green	ja	-
Brinkmann	ja	-

TID	Startzeit	Commitzeit	Aktion
Ta	$T0$	-	$\sum$
Tb	$T2$	-	$(Green)--$
Tc	$T3$	-	$\sum$
Td	$T5$	-	$\sum$

Beschäftigen wir uns mit *Multi-Version Concurrency Control* am Beispiel unserer verfügbaren Ärzte („Doctors on call/duty“), in dem wir sicherstellen wollen, dass immer mindestens ein Arzt verfügbar ist.

Uns stehen drei Operationen zur Verfügung, $\sum$ zählt alle verfügbaren Ärzte, $(X)++$ ändert Xs Status in verfügbar, $(X)--$ zählt alle verfügbaren Ärzte und ändert Xs Status auf nicht verfügbar, wenn mindestens ein Arzt noch anwesend ist.

1. Welche Bedingungen gelten für die Zeitstempel?
2. Green möchte zum Zeitpunkt $T2$ seinen Feierabend antreten. Vervollständigen Sie Tabelle 3 und legen Sie einen geeigneten Undo-Puffer (Zeitstempel, Attribut, Undo-Image) an. Wann muss Tb committen, damit Td bereits die Änderung von Tb liest? Was lesen Ta und Tb?
3. Brinkmann und House wollen zeitgleich den Feierabend antreten. House startet bei $T8$, Brinkmann bei $T9$. Wer darf gehen? Wie sorgt *Precision Locking* dafür, dass nur ein Arzt das Krankenhaus verlässt? Vervollständigen Sie die Einträge.

Gruppenaufgabe 6

In traditionellen Datenbanksystemen sind die Festplatte und der Buffermanager oft der Hauptgrund für Performanceengpässe. Wie ändert sich dies in Hauptspeicherdatenbanken, wo sind die neuen Flaschenhälse? Unterscheiden Sie auch zwischen Analytischen und Transaktionalen Workloads.

Fragen?