ניהול תנועות
חלק 5

Transaction Management
Part 5

1

בקרת מקביליות מרובת גרסאות
על ידי חותמות זמן

Multiversion Timestamps

2

בקרת מקביליות מרובת גרסאות
Multiversion Concurrency Control

- לכל פריט במסד שומרים גרסאות רבות
  - כשעיסקה T כותבת פריט A היא למעשה מייצרת גירסה חדשה 'A של A, כך ש- WTS('A) שווה ל- TS(T)
    - ערכו של 'A וגם WTS('A) אף פעם לא משתנים
  - לכל אחת מהגרסאות 'A של פריט A יש RTS('A), ששווה ל- TS של העיסקה הכי מאוחרת שקראה גירסה זו

3

תמיד אפשר לקרוא

- עיסקה T שרוצה לקרוא פריט A, קוראת את הגירסה 'A שיש לה את ה- WTS('A) הכי גדול, מבין כל אלה שקטנים מ- או שווים ל- TS(T)
  - מתי יכול להיות שיוויון?
- העיסקה מעדכנת את RTS('A) של הגירסה שקראה לפי הכלל הקודם, כלומר

RTS('A) := max{ TS(T), RTS('A) }

4

איך כותבים?

- אם עיסקה T רוצה לכתוב פריט A, אז צריך לבדוק שהמצב הבא אינו אפשרי
  - עיסקה 'T כבר קראה גירסה 'A של A, אבל אם T תכתוב גירסה חדשה של A, אז המשמעות תהיה ש- 'T קראה את הגירסה הלא המתאימה
- דוגמה: $A_{80,50}$ היא הגירסה של A שנכתבה בזמן 50 והעיסקה הכי צעירה שקראה אותה הייתה עם חותמת זמן 80
- האם מותר לעיסקה עם חותמת זמן 60 לייצר גירסה חדשה?

5

הכלל הפורמאלי

- הכלל המדויק הוא שלעיסקה T מותר לכתוב גירסה חדשה של A אם אין אף גירסה קיימת 'A כך ש- RTS('A) > TS(T) > WTS('A)
- אם הכלל מתקיים אז T מייצרת גירסה חדשה שה- WTS וגם ה- RTS שלה שווים ל- TS(T)
- אם התנאי אינו מתקיים, אז צריך לבטל את T ולהתחיל אותה מחדש
- מה בדיוק הכלל אם T רוצה לכתוב את A פעם שנייה?

6

מתי אפשר למחוק גרסאות

- אם לפריט A יש גירסה שזמן הכתיבה שלה הוא t וכל העסקאות שרצות במערכת התחילו אחרי זמן t, אז אפשר למחוק את כל הגרסאות של A שזמן הכתיבה שלהן קודם ל- t

7

סיכום

8

פרוטוקולים ללא נעילות לעומת 2PL

- פרוטוקולים ללא נעילות (שמבוססים על חותמות זמן) טובים יותר כאשר צפויים מעט קונפליקטים בין עסקאות
  - מכיוון שעסקאות אינן צריכות לחכות למנעולים
- פרוטוקול 2PL טוב יותר כאשר צפויים קונפליקטים רבים בין עסקאות
  - מכיוון שבפרוטוקולים ללא נעילות, קונפליקטים גורמים לביטול עסקאות ולהתחלתן מחדש, שזה מעכב עוד יותר משימוש במנעולים

9

השוואה בין הפרוטוקולים ללא נעילות

- הפרוטוקול שמבוסס על חותמות זמן מגלה מוקדם יותר מצבים שמחייבים ביטול עסקאות, בהשוואה לפרוטוקול האימות או לפרוטוקול האופטימי, אבל מצריך
  - בדיקה לפני כל כתיבה או קריאה,
  - חותמות זמן לכל פריט, וגם
  - מנגנון מתאים כדי לייצר תזמון שמאפשר התאוששות
- פרוטוקול האימות והפרוטוקול האופטימי מצריכים כתיבה כפולה (אם אין מספיק זיכרון פנימי)
- פרוטוקול האימות והפרוטוקול האופטימי טובים יותר כשאין כמעט קונפליקטים ויש מספיק זיכרון פנימי

10

אף פרוטוקול אינו משיג מקביליות מקסימלית

- נעילות חוסמות (כלומר, משהות) עסקאות הזקוקות לפריטים שכרגע נעולים על ידי עסקאות אחרות
  - פרוטוקול 2PL מאלץ עיסקה להחזיק מנעולים זמן רב יותר ממה שצריך רק כדי לקרוא או לכתוב
  - ←בעיה זו חריפה יותר בגירסה המחמירה של 2PL
  - נעילות גם גורמות לביטול עסקאות כתוצאה ממצבי קיפאון
- כאשר אין נעילות, חייבים לבטל ולהתחיל מחדש כל עיסקה שאינה עוברת את הבדיקה
  - הצורך להתחיל עסקאות מחדש מונע מקביליות מקסימאלית
  - גורם נוסף הפוגע במקביליות הוא הצורך לוודא שהתזמונים הנוצרים מאפשרים התאוששות (ראה שקפים 13 – 10 של CC2.3_TS.ppt)

11

שילוב

- רצוי שעסקה שרק קוראת לא תתעכב או תתבטל
  - לפרוטוקול רב הגרסאות תכונה חשובה זו (אבל הצורך למנוע קריאות מלוכלכות עלול לעכב גם עסקאות שרק קוראות)
- לדוגמה, שאילתת הקבצה מצריכה קריאת מספר רב של פריטים ויכולה לקחת זמן רב
  - לפיכך רצוי ששאילתה כזאת לא תחסם או תתבטל
- לעיתים משיגים תכונה זו ע"י שילוב, כלומר שימוש גם במנעולים וגם בפרוטוקול רב גרסאות
  - עסקאות שקוראות וכותבות מנוהלות ע"י 2PL וגם מייצרות גרסאות חדשות בזמן כתיבה
  - עסקאות שרק קוראות מנוהלות ע"י פרוטוקול רב גרסאות
  - קל יותר למימוש כשהפריטים הם בלוקים, כי אז מנהל החוצצים קובע איזה גרסאות של בלוקים צריך להשאיר בזיכרון לשימוש אפשרי של עסקאות פעילות

12