ניהול תנועות
חלק 3

Transaction Management
Part 3

1

בקרת מקביליות ללא נעילות

2

בקרת מקביליות ללא נעילות

- העסקאות מתבצעות ללא הגבלה, אבל יש צורך בבדיקות כדי לוודא שלא נוצר תזמון שאינו בר סדרתיות
- שני סוגים:
  - בקרת מקביליות אופטימית (optimistic concurrency control)
  - בקרת מקביליות המבוססת על חותמות זמן (Timestamp-Based Concurrency Control)

3

אם עסקה אינה עוברת את הבדיקה, צריך לבטל אותה

- כל עסקה נבדקת
  - אם הבדיקה עוברת בהצלחה, העסקה רשאית להתחייב
  - אם עסקה אינה עוברת את הבדיקה, המשמעות היא שהעסקה עלולה לגרום לתזמון שאינו בר סדרתיות, ולכן צריך לבטל אותה ולהתחיל להריץ אותה מחדש

4

בקרת מקביליות ללא נעילות לעומת בקרת מקביליות עם נעילות

- המחיר של בקרת מקביליות המבוססת על נעילות מורכב מהמחיר של
  - ניהול המנעולים, כולל הצורך לגלות או למנוע מצבי קיפאון
- המחיר של בקרת מקביליות ללא נעילות מורכב מהמחיר של
  - הבדיקות שצריך לבצע (כולל המחיר של שמירת מידע מסוים, שנחוץ לצורך ביצוע הבדיקות)
  - כתיבה כפולה (בגרסה האופטימית)

5

כל אחת משתי הגישות אינה משיגה מקביליות מקסימלית

- נעילות חוסמות (כלומר, משהות) עסקאות הזקוקות לפריטים, שכרגע נעולים על ידי עסקאות אחרות
- כאשר אין נעילות, חייבים לבטל ולהתחיל מחדש כל עסקה שאינה עוברת את הבדיקה
  - הצורך להתחיל עסקאות מחדש מונע מקביליות מקסימלית (בדומה לחסימות כאשר משתמשים בנעילות)

6

בקרת מקביליות אופטימית

Optimistic Concurrency Control

7

בקרת מקביליות אופטימית
Optimistic Concurrency Control

- בגישה האופטימית דוחים את הבדיקה עד לשלב הסיום של ביצוע העסקה, מתוך הנחה שיש רק סיכוי קטן שיהיה צורך לבטל את העסקה

8

השלבים העיקריים של ביצוע עסקה בגישה האופטימית

- שלב הקריאה
  - לכל עסקה מוקצה שטח עבודה פרטי בזיכרון הפנימי של המחשב, שלתוכו קוראת העסקה את הנתונים מהמסד
  - העסקה מבצעת את כל החישובים והכתיבות בשטח העבודה הפרטי
- שלב הבדיקה
- שלב הכתיבה
  - אם הבדיקה עברה בהצלחה, העסקה מעתיקה את התוצאות הסופיות משטח העבודה הפרטי למסד

9

קטע קריטי

- הקטע שבו עסקה מבצעת את שני השלבים האחרונים (בדיקה וכתיבה) הוא קטע קריטי
  - בכל נקודת זמן, המערכת מאפשרת רק לעסקה אחת (לכל היותר) לבצע את השלבים הללו
  - וכל עסקה מבצעת את שני השלבים הללו ברצף
- כלומר, העסקאות מבצעות את השלבים של בדיקה וכתיבה בצורה סדרתית לחלוטין
  - בפרט, רק עסקה אחת לכל היותר כותבת למסד הנתונים בכל רגע נתון
  - בזמן שעסקה כותבת למסד, אף עסקה אחרת אינה יכולה לקרוא מהמסד

10

אז איפה המקביליות?

- הבדיקה לוקחת מעט זמן באופן יחסי לזמן הריצה של העסקה
- גם הכתיבה לוקחת מעט זמן, כי כל החישובים כבר בוצעו ורק נשאר להעתיק את התוצאות הסופיות משטח העבודה הפרטי למסד
- לפיכך, רוב הזמן העסקה מתבצעת במקביל לעסקאות אחרות

11

חותמת זמן

- כל עסקה מקבלת חותמת זמן בתחילת שלב הבדיקה
- $TS(T_i)$ מסמן את חותמת הזמן של העסקה T_i
- הרעיון הבסיסי: התזמון הנוצר על ידי העסקאות שעוברות את הבדיקות שקול לתזמון הסדרתי שבו העסקאות מבוצעות לפי הסדר של חותמות הזמן
  - כלומר, עסקה שעוברת את הבדיקה "כאילו" מתבצעת בצורה אטומית ברגע של תחילת הבדיקה

12

אז מה צריך לבדוק?

- עסקה T_k צריכה להיבדק כנגד כל עסקה T_i שהתחייבה לפניה, כלומר
  - $TS(T_i) < TS(T_k)$ וגם
  - T_i עברה בהצלחה את הבדיקה
    - כמובן ש- T_i עברה את הבדיקה בהצלחה לפני ש- T_k החלה את הבדיקה
- לגבי כל פריט ש- T_k וגם T_i פועלות עליו (ולפחות אחת משתי הפעולות איננה קריאה), צריך לבדוק שהפעולה של T_k מתבצעת אחרי הפעולה של T_i

איך נבדוק?

- אם T_k התחילה אחרי ש- T_i הסתיימה, אז ברור שכל הפעולות של T_k מתבצעות אחרי הפעולות של T_i
- בכל מקרה, ברור שכל פעולות הכתיבה של T_k מתבצעות אחרי פעולות הכתיבה של T_i, כי העסקאות מבצעות את שלבי הבדיקה והכתיבה בצורה סדרתית (והנחנו ש- $TS(T_i) < TS(T_k)$)

אז מה נשאר לבדוק?

- אם T_k התחילה לפני ש- T_i הסתיימה, אז לכל פריט A שנכתב על ידי T_i ונקרא על ידי T_k, צריך לבדוק ש- T_k קוראת את הערך של A אחרי ש- T_i ביצעה את פעולת הכתיבה על A
- תנאי מספיק: T_i לא כותבת שום פריט ש- T_k קוראת

לסיכום, בגישה האופטימית, עבור עסקה T_k צריך לבדוק את הדבר הבא

- לכל עסקה T_i שהתחייבה לפני T_k, כלומר $TS(T_i) < TS(T_k)$, צריך להתקיים אחד משני התנאים הבאים:
  - T_k התחילה אחרי ש- T_i סיימה
  - T_i אינה כותבת שום פריט ש- T_k קוראת
- כדי לבצע את הבדיקות, צריך לשמור לכל עסקה את המידע הבא
  - רשימת הפריטים שהעסקה כותבת
  - רשימת הפריטים שהעסקה קוראת
  - זמני ההתחלה והסיום של כל עסקה
- עד מתי צריך לשמור את המידע על עסקה שהסתיימה בהצלחה?

שיפור לגישה האופטימית

- כאשר עסקה T_i עברה בהצלחה את שלב הבדיקה, אפשר לבדוק האם יש עסקה T_k שטרם החלה את שלב הבדיקה, אבל כבר קראה פריט A ש- T_i עומדת לכתוב
- אם יש עסקה T_k כנ"ל, אז ברור שהיא תיכשל בשלב הבדיקה
- אפשר להרוג את T_k מייד (kill policy) או לתת לה למות כשתגיע לשלב הבדיקה (die policy)

המידע שצריך לשמור

- צריך טבלה עם רשומות מהצורה
  (transaction id, item id, modified)
- כל עסקה, שקוראת פריט, מוסיפה רשומה כנ"ל לטבלה עם ה- id של העסקה וה- id של הפריט
  - השדה modified הוא בהתחלה false
- כאשר עסקה מסתיימת בהצלחה או מתבטלת, כל הרשומות שלה נמחקות מהטבלה

בשלב הכתיבה

- כאשר עסקה T_i כותבת למסד פריט A, היא שולפת מהטבלה את כל הרשומות עבור הפריט A ומשנה בכולן את השדה modified ל- true
- אפשר להרוג מייד (kill policy) את כל העסקאות, שאת הרשומות שלהן T_i שינתה, או לתת להן למות כשתגענה לשלב הבדיקה (die policy)

19

בשלב הבדיקה

- כאשר עסקה T_k נבדקת, צריך לשלוף מהטבלה את כל הרשומות עבור עסקה זו
- אם (לפחות) באחת הרשומות, עבור פריט שיסומן ע"י A, השדה modified הוא true, פירוש הדבר שעסקה שכבר התחייבה כתבה ערך עבור A, אבל T_k לא קראה אותו
- לכן צריך, לבטל את T_k

20

הפעולות על הטבלה מחייבות מנעול

- כאשר עסקה מוסיפה רשומה לטבלה או משנה את השדה modified בכל הרשומות עבור פריט נתון A, הדבר מצריך מנעול בלעדי על הטבלה

21

איזה גישה עדיפה? להרוג מייד או לתת למות בבדיקה?

- לכאורה, עדיף להרוג מיד
- אבל אם מאפשרים לעסקה להמשיך לרוץ עד שלב הבדיקה, היא תביא לזיכרון הפנימי את כל הדפים של המסד שהיא צריכה
- לכן, הריצה הבאה של העסקה תהיה הרבה יותר קצרה ולפיכך יש סיכוי גדול יותר שהיא תעבור את שלב הבדיקה בהצלחה

22

כמובן, שבריצה הבאה, כל הדפים של המסד שהעסקה צריכה כבר יהיו בזיכרון הפנימי בתנאי ש-

- יש מספיק מקום בזיכרון לשמור את כל הדפים (יחד עם דפים של עסקאות אחרות שרצות בינתיים)
- הדפים שהעסקה צריכה אינם משתנים מריצה אחת לשניה
- הערה: אם העסקה מתבטלת שוב, אז צריך להרוג אותה מיד, כי ממילא כל הדפים שלה כבר בזיכרון הפנימי

23

בקרת מקביליות המבוססת על חותמות זמן

Timestamp-Based Concurrency Control

24

בקרת מקביליות המבוססת על חותמות זמן
Timestamp-Based Concurrency Control

- במקום לבצע את הבדיקה לקראת סיום העסקה, מבצעים בדיקה בכל פעם שהעסקה קוראת או כותבת
- עסקה מקבלת חותמת זמן כאשר היא מתחילה
- מטרת הבדיקות לוודא שהתזמון הנוצר בפועל הנו שקול מבטים לתזמון הסדרתי שבו העסקאות מתבצעות לפי הסדר של חותמות הזמן
  - "כאילו" כל עסקה מתבצעת בצורה אטומית מיד כשהיא מתחילה

גם לכל פריט יש חותמות זמן

- לכל פריט A במסד יש שתי חותמות זמן
  - RTS(A) הנה חותמת זמן המציינת מתי הפריט נקרא לאחרונה (ליתר דיוק, הזמן של העסקה הצעירה ביותר שקראה את A)
  - WTS(A) הנה חותמת זמן המציינת מתי הפריט נכתב לאחרונה (ליתר דיוק, הזמן של העסקה הצעירה ביותר שכתבה את A)
- בעקרון, עסקה T שקוראת פריט A משנה את RTS(A) ל- TS(T) אם RTS(A) קטן מ- TS(T)
- באופן דומה, עסקה T שכותבת פריט A משנה את WTS(A) ל- TS(T) אם WTS(A) קטן מ- TS(T)

מה צריך לבדוק?

- עסקה T לא יכולה לקרוא פריט שכבר נכתב על ידי עסקה אחרת, שאמורה להתבצע אחרי T
- במילים אחרות, אם T רוצה לקרוא פריט A, אז
  - אם TS(T) < WTS(A), אז צריך לבטל את T ולהתחיל אותה מחדש (עם חותמת זמן חדשה)
  - אחרת T קוראת את A ו-
    RTS(A) := max{ TS(T), RTS(A) }

מה עוד צריך לבדוק?

- עסקה T לא יכולה לכתוב פריט שכבר נקרא על ידי עסקה אחרת, שאמורה להתבצע אחרי T
- במילים אחרות, אם T רוצה לכתוב פריט A, אז
  - אם TS(T) < RTS(A), אז צריך לבטל את T ולהתחיל אותה מחדש (עם חותמת זמן חדשה)
  - אם TS(T) < RTS(A) וגם TS(T) > WTS(A), אז T כותבת את A ו- WTS(A) := TS(T)

ומה קורה אם TS(T) < RTS(A) וגם TS(T) < WTS(A)?

- לא עושים כלום והעסקה יכולה להמשיך, מבלי לכתוב ערך חדש עבור A או לשנות את WTS(A)
- כלומר, השינוי ש- T הייתה אמורה לבצע בערכו של הפריט A הנו חסר משמעות, כתוצאה מכתיבה עיוורת של A על ידי עסקה אחרת T', שכבר כתבה את A ובתזמון הסדרתי השקול היא מופיעה אחרי T

איך יודעים שהכתיבה של A על ידי T' היא כתיבה עיוורת?

- TS(T') שווה ל- WTS(A), כי T' היא האחרונה שכתבה את A
- אילו T' גם קראה את A, אז היה מתקיים TS(T) < RTS(A), כי TS(T) < WTS(A)
- אבל נתון ש- TS(T) > RTS(A)

צריך למנוע קריאה מלוכלכת

- עסקה T שכותבת פריט A צריכה למעשה לחסום עסקאות אחרות שרוצות לקרוא את A עד שברור ש- T מתחייבת
  - שקול למעשה לאחזקת מנעול בלעדי על A עד ההתחייבות
- אם הדבר לא נעשה, אז יכולות להיות קריאות מלוכלכות, שעלולות לגרום למפל הפלות ולתזמון שאינו מאפשר התאוששות

סיכום

- פרוטוקול חותמות הזמן מאפשר תזמונים שאינם אפשריים על ידי 2PL (גם כשמונעים קריאות מלוכלכות)
  - וגם ההפך נכון
- בגלל המחיר של שמירת חותמות זמן לכל הפריטים במסד והצורך למנוע קריאות מלוכלכות, פרוטוקול חותמות הזמן פחות טוב מפרוטוקולים של נעילות במערכות מרכזיות
  - משתמשים בו בעיקר במערכות מסדי נתונים מבוזרים

בקרת מקביליות מרובת גרסאות

Multiversion Concurrency Control

בקרת מקביליות מרובת גרסאות Multiversion Concurrency Control

- לכל פריט במסד שומרים גרסאות רבות
  - כשעסקה T כותבת פריט A היא למעשה מייצרת גרסה חדשה של A, שה- WST(A) שלה שווה ל- TS(T)
  - לכל הגרסאות של פריט A אותו ה- RTS(A), ששווה ל- TS של העסקה הכי מאוחרת שקראה את A

תמיד אפשר לקרוא

- עסקה T שרוצה לקרוא פריט A, קוראת את הגרסה שיש לה את ה- WST(A) הכי גדול, מבין כל אלה שקטנים מ- TS(T)
- העסקה מעדכנת את RST(A):

RTS(A) := max{ TS(T), RTS(A) }

איך כותבים?

- אם עסקה T רוצה לכתוב פריט A, אז צריך לבדוק (כמו בפרוטוקול חותמות הזמן הרגיל) ש- TS(T) > RTS(A)
  - אם תנאי זה מתקיים, אז מייצרים גרסה חדשה של A, שיש לה WTS(A) ששווה ל- TS(T)
    - ה- RST(A) של הגרסה החדשה הוא כמו ה- RST(A) של כל הגרסאות הקיימות (לכולן אותו ה- RST(A))
  - אם התנאי אינו מתקיים, אז צריך לבטל את T ולהתחיל אותה מחדש

לסיכום

- העובדה שתמיד אפשר לקרוא היא היתרון הגדול של הפרוטוקול רב הגרסאות
  - פרוטוקול זה קיים באורקל
- לדוגמה שאילתת הקבצה מצריכה קריאת מספר רב של פריטים ויכולה לקחת זמן רב – לפיכך רצוי ששאילתה כזאת לא תחסם או תתבטל
- צריך למחוק גרסאות כאשר ברור שכבר לא יהיה בהן צורך

37

נושאים נוספים הקשורים לנעילות

38

מנעולים עם גרעיניות מרובה

Multiple-Granularity Locks

39

מנעולים עם גרעיניות מרובה Multiple-Granularity Locks

- אם עסקה צריכה לנעול את מרבית הדפים (בלוקים) של טבלה (יחס), אז עדיף שהעסקה תנעל את כל היחס, כדי להקטין את התקורה של הטיפול במנעולים
- לעומת זאת, אם עסקה צריכה לנעול רק חלק קטן מהדפים של טבלה, אז מוטב שהעסקה תנעל רק דפים אלה, כדי לא למנוע מעסקאות אחרות לפעול במקביל על דפים אחרים של אותה טבלה

40

באופן דומה

- אם עסקה צריכה לנעול את מרבית הרשומות הנמצאות בדף נתון, אז מוטב לנעול את כל הדף
- לעומת זאת, אם עסקה צריכה לנעול רק חלק קטן מהרשומות, אז מוטב לנעול רק את הרשומות האלה ולא את כל הדף

41

לפיכך, הבעיה היא כיצד לנהל נעילות ברמות שונות של גרעיניות

- איך מוודאים שמנעול בלעדי שניתן לעסקה אחת על דף מסוים אינו סותר מנעול בלעדי (או משותף) שניתן לעסקה אחרת על כל הטבלה?

42

אפשר לנעול לפי ההיראכיה הטבעית של מסד הנתונים

- המבנה ההיראכי
  - מסד הנתונים מכיל טבלאות
  - כל טבלה (יחס) מכילה דפים
  - כל דף (בלוק) מכיל רשומות
- ניתן לתאר מבנה זה כעץ, כאשר
  - השורש הוא כל מסד הנתונים
  - הילדים של צומת בעץ הם כל האיברים מהרמה הנמוכה יותר המוכלים בו

←לדוגמה, הילדים של טבלה הם כל הדפים של הטבלה

נעילת צומת ע"י מנעול S או X פירושה נעילת כל תת-העץ

- אם מנעול מסוג S או X מוחזק על צומת בעץ, אז המשמעות היא שמנעול זה מוחזק על על כל הצאצאים של הצומת
- לכן, עסקה אינה צריכה לנעול צומת אם היא מחזיקה מנעול מאותו הסוג על הורה או הורה קדמון (ancestor) של הצומת

צריך גם מנעולי כוונה Intention Locks

- מנעול מסוג intention exclusive מסומן ע"י IX
- אם עסקה נועלת צומת ע"י מנעול IX, אז פירוש הדבר שיש לה כוונה לנעול צאצאים של הצומת ע"י מנעול X
- מנעול מסוג intention shared מסומן ע"י IS
- אם עסקה נועלת צומת ע"י מנעול IS, אז פירוש הדבר שיש לה כוונה לנעול צאצאים של הצומת ע"י מנעול S

תאימות בין סוגי המנעולים

- כמו קודם, על צומת יכולים להיות מספר מנעולי S כל עוד אין מנעול מסוג X (של עסקה אחרת)
- על צומת יכולים להיות מספר מנעולי IS כל עוד אין מנעול מסוג X (של עסקה אחרת)
- על צומת יכולים להיות מספר מנעולי IX כל עוד אין מנעולים מסוג S או X (של עסקאות אחרות)
- לדוגמה, על צומת יכולים להיות מספר מנעולי IS ומספר מנעולי IX בו-זמנית
- אפשר גם מספר מנעולי IS ומספר מנעולי S בו-זמנית
- אם יש מנעול X, אז אין אפשרות לאף מנעול נוסף

הפרוטוקול לנעילות רב-גרעיניות

- עסקה חייבת להתחיל בנעילת השורש ע"י S, X, IX או IS
- עסקה יכולה לנעול צומת ע"י IS או S אם היא מחזיקה IS על ההורה של הצומת
- עסקה יכולה לנעול צומת ע"י IX או X אם היא מחזיקה IX על ההורה של הצומת
- *הערה:* כשעסקה נועלת צומת ע"י S או X, אז כל הצאצאים של הצומת נעולים גם הם ואין צורך להמשיך לרדת בעץ כדי לנעול

המשך הפרוטוקול

- אי אפשר להסיר מנעול מכל סוג שהוא (IS, IX, S או X) מצומת נתון, אלא אם כן אין שום מנעולים על הילדים של הצומת
- כל הנעילות (מכל ארבעת הסוגים) מתנהלות לפי הפרוטוקול של נעילות בשתי פאזות:
  - תחילה, עסקה מבצעת רק נעילות
  - לאחר השחרור הראשון של מנעול (מכל סוג שהוא), עסקה יכולה רק לשחרר מנעולים

מדוע נדרש התנאי ש

- עסקה יכולה להסיר מנעול מכל סוג שהוא (IS, IX, S או X) מצומת נתון רק אם אין לה שום מנעולים על הילדים של הצומת?
- אם עסקה מסירה את המנעול שלה מהשורש (ואין מנעולים אחרים על השורש), אז כל עסקה אחרת יכולה לנעול את השורש ע"י X ואז למעשה כל העץ נעול ע"י X
  - אם העסקה הראשונה עדיין מחזיקה מנעול S או X על צומת כלשהי בעץ זה סותר את הדרישה שהעסקה האחרת מחזיקה מנעול X בלעדי על כל העץ

מקרה נפוץ

- עסקה רוצה לקרוא יחס במלואו ולשנות מספר קטן של רשומות השייכות ליחס
- העסקה תנעל את הצומת המתאימה לכל היחס ע"י IX וע"י S
- בהמשך העסקה תנעל צאצאים נוספים ע"י IX או X

הסלמה של גירעון
Granularity Escalation

- איך מחליטים מהי הגרעיניות המתאימה לעסקה נתונה?
- אפשר להתחיל עם גרעיניות עדינה (למשל, נעילה ברמה של רשומות) ולאחר שמספר המנעולים שהעסקה מבקשת עובר סף מסוים, אז מגדילים את הגרעיניות לרמה הבאה (למשל, נעילות ברמה של דפים)

נכונות

- הפרוטוקול של נעילות רב גרעיניות מייצר רק תזמונים ברי סדרתיות קונפליקטית
- ההוכחה בספר של אולמן, עמוד 506
  - בספר של אולמן intention locks קרויים warnings

עוד נושאים נוספים
הקשורים לנעילות

בריחים Latches

- בעקרון, שתי עסקאות יכולות להחזיק בו-זמנית מנעולים על שתי רשומות שונות שנמצאות באותו בלוק
- מכיוון שאי אפשר לפעול בו-זמנית על חלקים שונים של אותו בלוק, עסקה חייבת להחזיק בריח על הבלוק לפרק הזמן שבו היא מבצעת קריאה או כתיבה של רשומה השייכת לבלוק

פעולות מקביליות על עצי B+

- כאשר ניגשים ליחס דרך עץ B+ (העץ יכול להיות אינדקס ראשי או משני), צריך לנעול גם את הצמתים של האינדקס דרכם עוברים ולא רק את היחס
- הרמות העליונות של העץ נחוצות למספר רב של עסקאות בו-זמנית
  - פרוטוקול 2PL ייצור עיכובים לא רצויים וצווארי בקבוק

55

אפשר לשפר

- ניתן לנעול צמתים של עץ B+ לפי פרוטוקול שמאפשר יותר מקביליות מאשר 2PL
- במיוחד, אפשר לשחרר את הנעילות על הרמות העליונות של העץ מוקדם יותר, כלומר אין צורך לחכות לפאזה של שחרור המנעולים
- פרטים בספר

56