

## תכונות ה- ACID The ACID Properties

- Atomicity: עסקה מתבצעת כפעולה אטומית – העסקה מתבצעת בשלמותה או בכלל לא
- Consistency: אם כל עסקה הנה עקבית (נכונה) ובתחילה המסד עקבי, אז גם בסוף המסד עקבי
- Isolation: ביצוע של עסקה מבודד מהשפעות של עסקאות אחרות
- Durability: אם עסקה מתחייבת אז ההשפעה שלה נשמרת

2

## התאוששות Crash Recovery

1

## הבעיות שיש להתמודד איתן

- עסקאות יכולות להתבטל (או ליפול – abort)
  - ▶ המערכת מבטלת עסקה (למשל בגלל קיפאון)
  - ▶ המשתמש מבטל את העסקה תוך כדי ריצתה
  - ▶ שגיאת חישוב (למשל חלוקה באפס)
- המחשב יכול ליפול
  - ▶ תקלת תוכנה או חומרה
  - ▶ הפסקת חשמל
  - ▶ הדיסק נהרס

4

## תכונות ה- ACID (המשך) The ACID Properties

- Atomicity: עסקה העסקה מתבצעת
- Consistency: אם ובתחילה המסד עקבי
- Isolation: ביצוע עסקאות אחרות
- Durability: אם עסקה נשמרת

3

## התאוששות מנפילה

- אם יש נפילה כמתואר בציור, אז כשהמחשב חוזר לפעול, צריך לשמר את השינויים שעשו T1, T2 ו-T3, ולבטל את השינויים שעשו T4 ו-T5

6

## ביטול עסקה

- כאשר עסקה מתבטלת, אבל המחשב ממשיך לפעול, צריך לבטל את השינויים שהעסקה ביצעה במסד הנתונים ("Rollback")
- אם משתמשים ב- 2PL מחמיר (Strict 2PL), אז אין צורך לבטל שינויים של עסקאות אחרות (קרי, אין מפל הפלות)

5

## אטומיות של עסקה

- עסקה כותבת לדיסק את השינויים שביצעה
- איך נדע האם העסקה התחייבה?
  - ▶ נכתוב לדיסק (במקום מתאים) שהעסקה התחייבה
  - ▶ העסקה חייבת לכתוב לדיסק את כל השינויים שביצעה לפני שהיא כותבת לדיסק שהיא התחייבה
- אם המחשב נפל ולא כתוב בדיסק שהעסקה התחייבה, איך נבטל את השינויים שביצעה?
  - ▶ מייבום לכתוב את כל השינויים לדיסק פעמיים

8

## הדיסק נהרס Disk Crash

- אם הדיסק נהרס באופן מלא או חלקי, צריך לשחזר אותו – וזו בעיה בפני עצמה
- **אנו מניחים שכתובה של בלוק בודד לדיסק היא אטומית – מתבצעת בשלמותה או לא מתבצעת בכלל**
  - ▶ זו הנחה נכונה כל זמן שהדיסק לא נהרס
  - ▶ אבל יתכן שאינה מתקיימת לגבי הבלוק האחרון שנכתב, אם המחשב נפל תוך כדי כתיבה זו
  - ▶ יש לשחזר את הבלוק האחרון שנכתב במקרה הצורך

7

## ביצוע עסקה כפעולה אטומית

- עם סיומה, עסקה כותבת "commit" ללוג
  - ▶ כתיבה זו מצריכה, למעשה, כתיבה של בלוק אחד לדיסק ולכן מתבצעת כפעולה אטומית
- אם בלוג כתוב "commit" אז ניתן לשחזר את השינויים שהעסקה ביצעה לפי המידע בלוג
- אם בלוג לא כתוב "commit" אז ניתן לבטל את השינויים שהעסקה ביצעה לפי המידע בלוג

10

## כתיבה מקדימה ל-LOG Write-Ahead Log (WAL)

- הלוג (LOG) הוא מקום מיוחד בדיסק שבו עסקה רושמת את
  - ▶ כל השינויים שהיא מבצעת במסד
  - ▶ מידע חיוני נוסף, כגון העובדה שהעסקה התחייבה
- **שינויים נכתבים למסד על הדיסק רק אחרי שהם נכתבו ללוג על הדיסק**
- **עסקה חייבת לכתוב את כל השינויים שעשתה ללוג על הדיסק לפני שהיא מתחייבת**

9

## מאגר חוצצים Buffer Pool

- בלוקים (דפים) נקראים מהדיסקים לחוצצים בזיכרון הפנימי
- עסקה קוראת מהחוצצים וכותבת לחוצצים
- בעקרון, עם סיומה, העסקה צריכה לכתוב לדיסקים את הבלוקים ששינתה בחוצצים
- אבל למעשה עדיף להשאיר בלוקים אלה בזיכרון, כי אולי עסקה אחרת תזדקק להם

12

## הכתיבה הכפולה – ללוג ואח"כ למסד – מייעלת

- הלוג נמצא על דיסק מיוחד
- הכתיבה בלוג היא סדרתית – תמיד מוסיפים רשומות חדשות לסוף הלוג
- כל המידע שיש לרשום בלוג עבור עסקה נתונה מצריך בלוק אחד או (מספר קטן של בלוקים), לפיכך:
  - ▶ הלוג מצריך, עבור כל עסקה, כתיבה של מספר קטן של בלוקים
  - ▶ הלוג מאפשר אי-כפייה וגניבה ולכן מייעל

11

## מה עושים אם החוצצים מלאים?

- בחוצצים ניתן לאחסן בו-זמנית מספר מוגבל של בלוקים
- אם עסקה צריכה לקרוא מהדיסק בלוק נוסף של המסד ואין מקום בזיכרון הפנימי, אז
  - ▶ אם יש בזיכרון הפנימי בלוק של המסד שכרגע אינו נעול ע"י אף עסקה, אפשר לפנות בלוק זה, אבל תחילה יש לכתוב אותו לדיסק (כדי לשקף במסד את השינויים שבוצעו בבלוק זה)

14

## אי-כפיה No-Force

- לא כופים על עסקה שביצעה commit לכתוב לדיסק את הבלוקים של המסד שהיא שינתה
  - ▶ ניתן לשחזר את השינויים מהלוג (כזכור, כל השינויים נכתבים ללוג על הדיסק לפני ההתחייבות)
- עסקה יכולה לשנות מספר רב של בלוקים של המסד, אבל למעשה אין צורך לכתוב אותם לדיסק, תודות למספר הקטן של בלוקים שהעסקה חייבת לכתוב ללוג

13

## הדברים שהלוג מאפשר – סיכום

- הלוג מאפשר לשחזר במסד שינויים שבוצעו ע"י עסקאות שהתחייבו
- הלוג מאפשר לבטל במסד שינויים שבוצעו ע"י עסקאות שלא התחייבו
- הלוג מאפשר אי-כפיה וגניבה ולכן מקצר את זמן התגובה של המערכת

16

## גניבה Steal

- אם כל הבלוקים שבזיכרון הפנימי נעולים, אז
  - ▶ אפשר "לגנוב" מקום של בלוק שנמצא כרגע בזיכרון הפנימי, אבל תחילה צריך לכתוב אותו לדיסק, כדי לשקף את השינויים שבוצעו בו
  - ◀ יתכן שחלק מהשינויים בוצעו ע"י עסקאות שהתחייבו וסיימו, בעוד ששינויים אחרים בוצעו ע"י עסקאות שכרגע נועלות רשומות שנמצאות בבלוק ואולי לא יסיימו
  - ▶ בעזרת הלוג, תמיד ניתן לבטל את השינויים שבוצעו בבלוק זה ע"י עסקאות שלא יסיימו

15

## סוגי רשומות בלוג

- רשומה עבור ביצוע פעולת עדכון (update)
  - ▶ מחיקה והוספה הם מקרים מיוחדים של עדכון
- רשומה עבור ביצוע commit
- רשומה עבור ביצוע abort
- רשומה המציינת שהעסקה הסתיימה (end)
- סוגים נוספים יפורטו בהמשך

18

## מודל פשטני ללוג ולהתאוששות

- לכל רשומה בלוג יש שדה עם מספר סידורי, הקרוי LSN (Log Sequence Number)
  - ▶ הרשומות נכתבות ללוג בסדר עולה, לפי ה-LSN
- כמו כן יש שדות עבור:
  - ▶ מספר העסקה שאליה מתייחסת הרשומה
  - ▶ סוג הרשומה
  - ▶ ה-LSN הקודם עבור אותה עסקה (רשומות הלוג עבור עסקה נתונה משורשרות לאחור)
- יש שדות נוספים לפי סוג העסקה

17

## ביטול (abort) של עסקה בודדת (המערכת ממשיכה לרוץ)

- רוצים לבטל עסקה שטרם התחייבה
- ▶ העסקה עדיין מחזיקה מנעולים על כל הפריטים ששינתה
- כותבים רשומת abort ללוג עבור העסקה
- מבצעים Undo לעסקה

20

## רשומת לוג עבור עדכון

- לכל פריט A שמעדכנים, יש בלוג רשומה עם:
  - ▶ מספר סידורי (LSN)
  - ▶ מספר סידורי של הרשומה הקודמת עבור אותה עסקה
  - ▶ מספר סידורי של העסקה
  - ▶ סוג הרשומה (בשקף הנוכחי הסוג הוא עדכון)
  - ▶ ערך של הפריט A לפני העדכון (before-image)
  - ▶ ערך של A אחרי העדכון (after-image)

19

## התאוששות מנפילה

- כאשר המערכת נופלת, צריך לעבור על הלוג (מתחילתו) ולבצע שלושה שלבים
  - ▶ שלב האנליזה (Analysis Phase)
  - ▶ שלב העשייה מחדש (Redo Phase)
  - ▶ שלב ביטול העשייה (Undo Phase)

22

## ביצוע Undo של עסקה בודדת

- עוברים על רשומות הלוג של העסקה מהסוף להתחלה
  - ▶ לכל רשומת עדכון עבור העסקה, כותבים למסד את הערך של הפריט לפני העדכון
  - ▶ בסיום המעבר על כל רשומות הלוג של העסקה, כל הפריטים ששוננו ע"י העסקה חזרו לערכם המקורי
  - משחררים את כל המנעולים שהעסקה החזיקה

21

## שלב העשייה מחדש Redo Phase

- בשלב זה מבצעים מחדש את כל העסקאות (כולל אלה שלא התחייבו)
- עוברים על הלוג מתחילתו, לפי הסדר של הרשומות
  - ▶ לכל רשומת עדכון, כותבים למסד את הערך של הפריט אחרי העדכון

24

## שלב האנליזה Analysis Phase

- בשלב זה עוברים על הלוג (מתחילתו) ובודקים עבור איזה עסקאות יש בלוג רשומות commit ועבור איזה – אין
  - צריך לדאוג שכל השינויים שבוצעו ע"י עסקאות שהתחייבו אומנם יכתבו על הדיסק של המסד
  - צריך לבטל את כל השינויים שנכתבו על הדיסק של המסד ע"י עסקאות שלא התחייבו

23

## נקודות חשובות

- אין זה משנה האם פעולות עדכון שמופיעות בלוג אומנם בוצעו בפועל על הדיסקים של המסד לפני הנפילה
- ▶ התוצאה הסופית של שלושת השלבים תלויה רק במה שכתוב בלוג ולא במה שכתוב במסד
- פעולות עדכון שטרם נרשמו על הדיסק של הלוג (ולכן נעלמו) גם לא נרשמו על הדיסק של המסד – לכן אפשר להתעלם מהן

26

## שלב ביטול העשייה (Undo Phase)

- בשלב זה מבטלים את כל הכתיבות של עסקאות שלא התחייבו
- עוברים על הלוג מהסוף להתחלה
- ▶ עבור כל רשומת עדכון של עסקה שלא התחייבה, רושמים למסד את הערך של הפריט לפני העדכון
- בסיום שלושת השלבים, השינויים למסד משקפים בדיוק את הכתיבות שבוצעו ע"י עסקאות שהתחייבו

25

## נקודת ביקורת Checkpoint

- זה לא יעיל לבצע התאוששות מתחילת הלוג
- לפיכך מפעם לפעם מבצעים checkpoint
- ▶ מפסיקים לקבל עסקאות חדשות ומסיימים את כל העסקאות שעדיין רצות
- ▶ כותבים את כל השינויים לדיסקים של המסד
- ▶ כותבים רשומת checkpoint ללוג
- ▶ מתחילים שוב לקבל עסקאות לביצוע
- צריך לבצע התאוששות מה-checkpoint האחרון

28

## נפילה תוך כדי ביצוע התאוששות

- אם יש נפילה תוך כדי ביצוע התאוששות, אז מתחילים את תהליך ההתאוששות מחדש
- מכיוון שהתוצאה של תהליך ההתאוששות תלויה רק במה שכתוב בדיסק של הלוג (ואינה תלויה במה שכתוב בדיסק של המסד), התוצאה הסופית היא נכונה

27

## כמה הערות לגבי אלגוריתם אמיתי להתאוששות

- פריטים לוגיים לעומת פיזיים
- Fuzzy checkpoint
- Fuzzy backup
- הקטנת מספר הכתיבות לדיסקים של המסד בשלבים של redo ו-undo

30

## גיבוי

- מפעם לפעם צריך לגבות את הדיסקים של המסד, כדי שניתן יהיה לשחזר את המסד במקרה של הרס הדיסקים
- רשמים ללוג שבוצע גיבוי
- אם הדיסקים נהרסו, צריך להעתיק למסד את העותק שנשמר בגיבוי ולבצע התאוששות (בעזרת הלוג) מרשומת הגיבוי האחרונה (הכי מאוחרת) שמופיעה בלוג

29

## פריטים פיזיים

- ❁ לפיכך, בדר"כ שומרים בלוג שינויים שבוצעו על פריטים פיזיים
  - ▶ הכתובת (על הדיסק) של הבלוק ששונה
  - ▶ הכתובת (מרחק מראשית הבלוק) של ה-byte הראשון ששונה
  - ▶ אורך הקטע (מספר ה-bytes) ששונה
  - ▶ הערך של הקטע לפני השינוי (before-image)
  - ▶ הערך של הקטע אחרי השינוי (after-image)

32

## פריטים לוגיים לעומת פיזיים

- ❁ הפריטים (שאת השינויים שלהם כותבים ללוג) יכולים להיות לוגיים, למשל רשומות
  - ▶ לכן, צריך לזהותם לפי מפתח לוגי (למשל, מספר עובד)
- ❁ מקומם הפיזי (על הדיסק) של פריטים לוגיים עשוי להשתנות כשמבצעים עדכונים על המסד
  - ▶ אי אפשר לשחזר בלוק בודד של דיסק וצריך לשחזר את הדיסק כולו (יקר לביצוע)

31

## Fuzzy Checkpoint and Fuzzy Backup

- ❁ רצוי לעשות checkpoint מבלי להפסיק את פעולתן של עסקאות
  - ▶ אפשרי, אבל מסובך יותר
- ❁ באופן דומה, לגבי גיבוי

34

## יחד עם זאת...

- ❁ יש מקרים שבהם עדיף לשמור שינויים לפריטים לוגיים
  - ▶ למשל, שינויים באינדקס של B-tree
- ❁ הסיבה: הביטול של שינוי איננו בדיוק הפעולה ההפוכה של השינוי עצמו

33

## אלגוריתם ההתאוששות אריס The Aries Recovery Algorithm

- ❁ אלגוריתם ההתאוששות אריס מאפשר
  - ▶ Fuzzy checkpoint and backup
  - ▶ לשמור שינויים של פריטים פיזיים או לוגיים
  - ▶ להקטין את מספר הכתיבות לדיסקים של המסד ע"י שמירת מידע פשוט
  - ▶ קל (יחסית) להבנה ולמימוש
- ❁ האלגוריתם מתואר בפרק 20 בספר

36

## הקטנת מספר הכתיבות

- ❁ בעזרת מידע פשוט ששומרים ניתן להקטין את מספר הכתיבות לדיסקים של המסד בזמן התאוששות

35