Homework 1: Distributed RDBMS and NoSQL – Theoretical Part

שאלה 1

תיאור 4 מאפייני ה-ACID. עבור כל מנגנון יתואר גם מצב של כשל אפשרי כאשר מנגנון זה אינו חל.

1. אטומיות (Atomicity) - כל פעולת הטרנזקציה מתבצעת יחד או ששום דבר לא קורה ("הכל או כלום").

נתבונן במשיכה של כסף מכספומט. נניח כי פלוני מבקש למשוך 500 ש"ח. טרם המשיכה נכתב ל-DB כי נמשכו על ידי כרטיס האשראי שלו 500 ש"ח וכן משתנה רישום היתרה הכללית בכספומט ואז מתרחשת תקלה, החשמל בכספומט נופל.
במצב זה, במידה ולא מתקיים עקרון האטומיות, אף שפלוני לא משך את הכסף, כרטיס האשראי שלו יחויב וכן היתרה הכללית בכספומט אינה תואמת את הרישום ב-DB.

2. עקביות (Consistency) - מעבר ממצב לוגי תקין אחד למצב לוגי תקין אחר.

נשאר בדוגמת הכספומט, נניח כי פלוני מבקש למשוך סכום של 5,000 ש"ח, כאשר היתרה בכספומט הינה 4,000 ש"ח. הוצאה לפועל של טרנזקציה זו, ללא וידוא היתרה הזמינה, תביא למצב בו היתרה הרשומה בכספומט הינה מינוס 1,000 ש"ח (מצב לוגי לא תקין) ופלוני יקבל בפועל רק חלק מהסכום אותו ביקש למשוך.

3. בידוד (Isolation) - אין השפעה של טרנזקציה אחת על טרנזקציה אחרת המתבצעת באותו הזמן.

נתבונן ברכישה מקוונת של כרטיס למופע. פלוני ופלונית מעוניינים לרכוש כרטיס, כל אחד ממחשבו האישי, באותו הזמן ממש. לרוע המזל נותר רק כרטיס פנוי אחד. ללא בידוד, לשניהם יוצג כי קיים כרטיס אחרון והמערכת תאפשר לשניהם לרכוש אותו אף כי בפועל זמין רק כרטיס אחד.

4. עמידות (Durability) - השינויים כתוצאה מטרנזקציה שבוצעה (commit), ישמרו לעד (כלומר לא יאבדו).

פלוני רוכש מבעוד מועד הסדר נסיעה לרכבת. תהליך הרכישה הושלם בהצלחה ופלוני קיבל אישור עסקה. לצער כולם, בשל מגפה שפרצה בקרב עובדי ושרתי הרכבת, קרסו השרתים וכאשר עלו, רשומת הרכישה נמחקה. כאשר פלוני הגיע להטעין ולתקף את כרטיס הרב קו שלו במכונה האוטומטית, מופיעה הודעת שגיאה כי לא קיים הסדר נסיעה בתוקף.

שאלה 2

תיאור של replication and fragmentation לרבות יתרונות וחסרונות של כל אחד -

1. replication - המערכת שומרת מספר עותקים של אותו ה-data, מאוחסן במיקומים פיזיים שונים.
יתרונות - גיבוי ושרידות, מקבול, זמינות, קרבה למיקום הגיאוגרפי בו נמצאים המשתמשים (הן בימד המהירות והן בכך שבכל אזור ניגשים למידע שרלוונטי אליהם - הפחתת המידע המועבר בפועל - "read").
חסרונות - עלויות, כפילות בפעולות עדכון ותחזוקה, מורכבות.

2. fragmentation - מתבצעת הפרדה (Partition) של המידע לחלקים. כל חלק נשמר במיקום אחר.
ההפרדה יכולה להיעשות במימד השורות (Horizontal) או במימד העמודות (Vertical).
יתרונות - מקביליות, התאמת תוכן ה-data למיקום בו נדרש, צמצום נזק מטעויות (לדוג' רק חלק מה-data נמחק במקרה של טעות), זמינות חלק מה-data במקרה שאתר מסוים אינו זמין. עלויות ביחס ל-replication.
חסרונות - אין מקביליות בעבודה על כל ה-data, כאשר נדרש מידע רחב היקף (מכלל האתרים) פעולות join יקרות במקרה של vertical ו/או פעולות union במקרה של horizontal, עומס על הרשת. שרידות המידע חלקית ביחס ל-replication.

תיאור של 2 דוגמאות מהעולם האמיתי:

1. Horizontal - לחברה בינלאומית המציעה שירותים לחברות פיננסיות גדולות, יש לקוחות במספר מוקדים, לדוג' ביפן ובגרמניה. כללי הרגולציה במדינות אלו מחייבים שמידע רגיש יעובד וישמר במדינת המוצא. כפועל יוצא החברה מבצעת Horizonal Fragmentation, כאשר הסכמה זהה, אולם פרטי לקוחות יפניים נשמרים באתר ביפן ופרטי לקוחות גרמניים באתר בגרמניה.
2. Vertical - חברת טכנולוגיה שומרת מידע רב על לקוחותיה. מידע זה משרת 2 קבוצות שונות בחברה - קבוצת המחקר וקבוצת ה-support, כאשר הראשונה מתעניינת במידע אודות מאפייני המשתמשים והשנייה בפרטי ההתקשרות עמם. בעבר המידע על הלקוחות נשמר בטבלה אחת. עם מעבר קבוצת ה-Support למדינה במזרח אירופה, בוצע Vertical Fragmentation, כך שהעמודות אודות פרטי ההתקשרות הועברו לאתר אחר במזרח אירופה.

שאלה 3

יתרונות

1. ביצועים (מאחורי הקלעים - מנגנוני דחיסה, cache, ו-recovery)
2. זמינות גבוהה
3. גמישות בכל הנוגע לסקייל (scale; כיוון שהדרישות משתנות מעת לעת)

מגבלות

1. ה-UI - מקשה על משתמשים להבין איך להשתמש בפלטפורמה בצורה מיטבית
2. אינדוקס - תמיכה הבאינדקס ראשי בלבד
3. התשתית אינה מאפשר replications
4. *תחזוקה - לא קיים מוצר מדף / SaaS לניהול ותחזוקה של ה-clusters