בשביל שמדד כלשהו שאוטומטית לא נכונות תמיד. (לדוגמא Min-Max).

דוגמא נוספת בשכבת שליטה היא הפת בתיבת Vision. בשיטה זו, אנו יכולים ליצור כלפי מעלה תנאים מ משוע אחד לשליטתו ובאמצעות הוספה של התנאי גם ה משוע השני. בדוגמא שבשקפות אנו רואים שבגלל שב V1 יש תנאי שגם הרשומות יהיה גבוה מ- 20, נוכל להוסיף את התנאי הזה לשליטה (כי בכל מקרה בגלל V1 לא יהיו בה רשומות עם גיל קטן מ20). בנוסף, נוכל להוריד את התנאי שלא ל- V2 (גם תמיד נשתמש בו יחד עם V1).

בשכבת עליון - אנחנו יכולים להשתמש בכלים סטטיסטיים כדי להפריד עליון יכול למחוק גם דרך אינטואיציה לבדוק שליטה. השיטה אינה מדויקת. לדרוך שיטת בשיטה יש לשמור גם טבלה. אפשר פרטי מידע. סטטיסטי אי אפשר לבדוק עם כל שליטה. אך כן מתאדכנים מפעם לפעם. כי הרבה שומרים גם מספר הדפים ומספר הסרטת בטבלה. בנוסף שומרים מראה אינדקסים מה התפלגות הדפים באינדקס (כך נוכל להפריד כמה רשומות יקומדמו אנו בתת בדיוק שליטה). כן, שומרים את גובה האינדקס (כלומר עליון הדפים)

באמצעות הנתונים שאנו שומרים אנו יכולים לחשב את RF (פקטור הדמדום) המייצג את הדמדום בבלוק בתת שליטה גם term מסוים. חושב ה RF מבוטל באמצע לגם שליטה. מכיר שליטה "בשיטה", נבדוק את RF. לדוגמא. 1 תלוי מספר הדפים השונים באינדקס מסוים. מכיר שליטה חוסך RF יהיה 1 תלוי המקסימום של מספר הדפים השונים בכל אחד מהאינדקסים של גם טבלה. בעיקר נשתמש מבדוק כמות השינוי של טווח נבדוק את RF לדוגמא H(I)-Value / H(I)-L(I) (הסבר למה ישלם בדף).

הסטטיסטיקה - בשיטה זו אנחנו לא מנתחים "התפלגות גיאוגרפית" אלא מחלקים את הדפים לקבוצות ומכל כל טווח מחשבים כמה דפים יש בטווח המדגם. כך נוכל להגיע להערכה טובה. בתת בדיוק חוסך, נבדוק כל טווח בכל טווח מהבלוק (1) בכל הטבלה מול כל הטבלה) ומשתמשים בהערכה זו לחישוב (שקפות 37).

בתת בדיוק חוסך יש מספר דרכים לבדוק אותו (סדר שינוי). כדי להחליט על הסדר של בדיוק נוכל להשתמש בתכונות דינאמי כדי לחשב כמה כל פעולה חוסך עדיף ולבסוף לקבוע את המסלול הכולל ביותר.

סוף בשביל לא נוכנס לעומק של הנושאים הבאים אבל כן יש כמה חשובים שכדאי לדעת. ומעניינים. בגלל מבנים הנלכדים על מדמים.
הרבה פעמים נתונים נתונים על שליטה ואל נשלח כמה ועלה מבלי אינדקס הפעלה (חשיבות), הבעיה בין אחד/חוסך Hash, מה קרה בשרת יש בפעולת Select לפני חוסך וכו'.

XML

הינה דרך מוסכמת לייצוג נתונים. מבחר כל נתון אנחנו רוצים מה תוא מייצג, לה טוב כי לא נוכל בשיטת לה רק כי לא ליקח המון מקום. דוגמאות לשימוש ב- XML: RSS, SOAP וכו'.
בנוסף משתמשים ב- XML כדי לבדוק אינטראקציה בין מערכות שונות על הוספת שלב בנוסף מאבר בין המערכות.

ההבדל בין HTML לבין XML הינה שב- HTML אנו מתארים כיצד הנתונים יבואו לב- XML אנו מתארים את הנתונים עצמם.

DTD - מגדיר את השימוש בסכמה סטנדרטית בתת בתיבת XML. אם אין DTD ניתן לבשתמש בעיצה בסכמה שרוצים. כלומר משהו מגדיר את התיבה בתת בתיבת מתוך XML תת DTD מסוים. מגדיר במדגים.

XML מתייחס את עצמו כל TAG מתאר איזה נתונים הוא מכיל.

כדי לבצע שאילתות על XML, אנחנו יכולים להשתמש ב XPATH או ב- XQuery

XPATH מסתכל על ה- XML כעץ. בשביל זה מגדיר "נתיב" ומחזיר את כל הnodes אשר הנתיב שהם מכיל את ה prefix שביקשנו.

אם נופל 2 מצבים בחלקי הנתיב, אנחנו נקבל תוצאה את כל ה- Nodes שמתאימים לחלק. לאו דווקא מהשורש (כלומר אם הנתיב קיים בתוך אחד מהבנים ולא דווקא בתחילת הנתיב שלו נקבל אותו.

הוספת text() בכל הנתיב מחזיר את הטקסט שבתוך הצמתים (node מחזיר טקסט) או שאם * מחזיר את כל הילדים תחת הנתיב שלפניו. @ מחזיר Attributes וכו'.

אם מגדירים [תנאי] אז ומחזירו לנו כל ה-nodes שהנתיב שלו רק כזה אשר מקיים את התנאי שמופיע בסוגריים.

בנוסף, ניתן להפעיל תנאים על Attributes באמצעות הסוגריים.

- XQuery מגדירה את SQL. היא טובה יותר מ xPath. ייתר תוקף, מאפשר בסיסי שלה חוא מבנה הפרה (Fowler).

בחלק של ה- For ניתן להשתמש ב- 2 משתנים (ובאחד מהם להשתמש בשני). זה יוצר מצב פחות יעיל For בתוך For (שקופית 8). בנוסף, במידה שרשלבה יתכנו שכפולים טוב לנו להשתמש במילת המפתח Distinct.

כמו כן, נוכל להגדיר "פרה" בתוך חלק ה- Result. כלומר הדרך של אלאה בתוך מילה נוספת ויותר לסדר.

Let מאפשר לנו להגדיר משתנה קבוע, למשל, חישוב קבוע של הממוצע ולאחר מכן הצגה של כל הפריטים שמתחת לגבוה מהממוצע.

נשים לב שבעת שימוש ב Distinct המדד אשר הרשימה ומחזיר בו אינו בהכרח לפי המדד הקודם.