דאטה לייק חסכוני עם Apache Iceberg ו-Amazon S3 Tables

AWS Summit Tel Aviv 2026 ,סיכום סשן — TLV-ANT303

תומר איתן, Storage Specialist Solutions Architect, AWS | דן קולודני, Solutions Architect, AWS | ליאור רסיסי, R&D Director, Windward

10 בספטמבר 2026 | משך: כ-39 דקות | הופק מהקלטת הסשן

סיכום מאת קובי אלמוג

על המסמך הזה

המסמך מסכם את הסשן TLV-ANT303 מתוך AWS Summit Tel Aviv 2026, מסלול AWS For Data and Analytics. הוא נבנה מתמלול אוטומטי של ההקלטה המלאה ומהסליידים שחולצו מהווידאו עצמו, כך שאפשר לחזור לתוכן בלי לצפות שוב בארבעים הדקות. חותמות הזמן בגוף המסמך מפנות לנקודה המדויקת בהקלטה.

הסשן בנוי בשלושה קולות: תומר איתן פותח ברקע על S3 ועל S3 Tables, דן קולודני מסביר את המבנה הפנימי של Apache Iceberg ומריץ דמו חי, וליאור רסיסי מ-Windward מספר סיפור לקוח מלא — כולל התקלה שנתקעו בה והמספרים שהתקבלו אחרי התיקון.

איך לקרוא את זה

- **פרק 1 — תקציר מנהלים:** חמש דקות קריאה. מה הוצג, ומה רלוונטי לארגון שמנהל דאטה לייק.
- **פרקים 2–4 — הרקע הטכני:** למה S3 השתנה, איך Iceberg בנוי מבפנים, ומהן שלוש פעולות התחזוקה שאי אפשר לברוח מהן.
- **פרקים 5–7 — S3 Tables:** מה בדיוק מנוהל, Intelligent-Tiering, ורפליקציה קונסיסטנטית.
- **פרק 8 — הדמו:** יצירת טבלה מהקונסול, ההשוואה מול bucket רגיל, ותשאול בשפה חופשית.
- **פרקים 9–10 — Windward:** המסע האמיתי, כולל הבאג שעלה כסף, והמספרים שאחרי.
- **פרק 11 — מה לוקחים מכאן:** מסקנות ונקודות להמשך, ואחריהן מילון מונחים.

> **הערת מקור** המסמך הופק מתמלול אוטומטי של ההקלטה. התמלול מדויק בקטעים המקצועיים, אך שמות ומונחים לועזיים מופיעים בו בשיבוש פונטי (למשל "אייסברג" = Iceberg, "קומפקשן" = compaction) והם נורמלו כאן למונח האנגלי. הציטוטים אומתו מול ההקלטה בחותמת הזמן המצוינת ומובאים כלשונם, כולל שיבושי התמלול. מספרים שמופיעים על הסליידים צוינו ככאלה, ובמקום שבו הסליד והדובר אמרו מספרים שונים — שניהם מובאים.

1. תקציר מנהלים

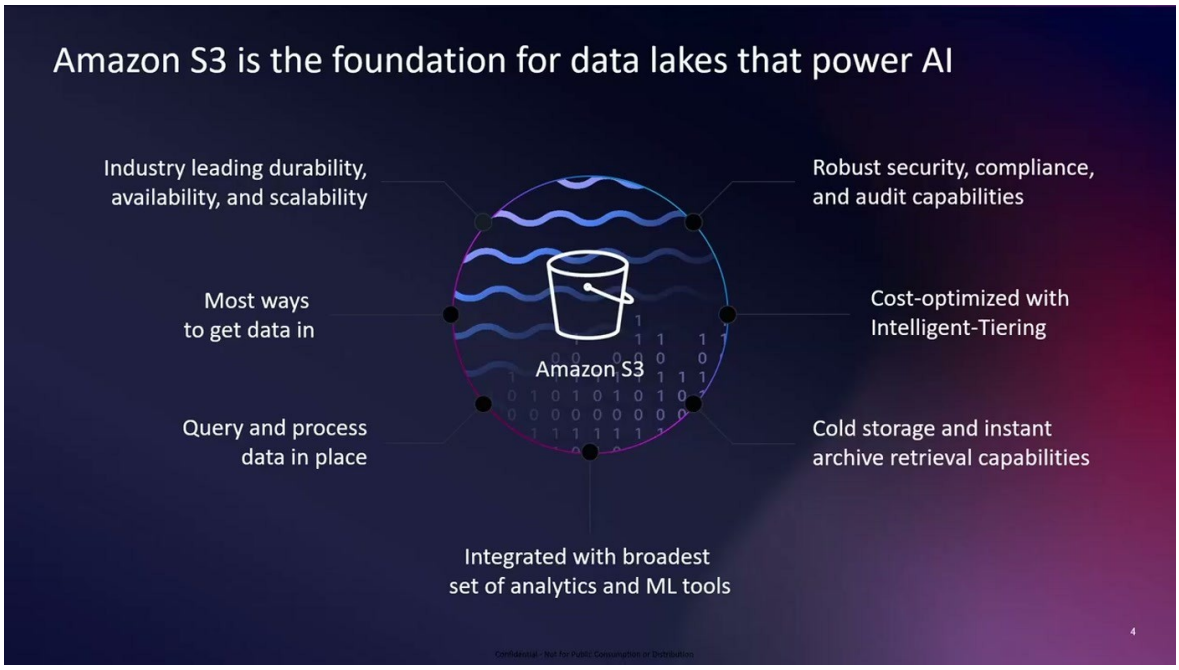
הסשן עונה על שאלה תפעולית אחת: מי מתחזק את טבלאות ה-Iceberg שלכם. Apache Iceberg הפך לסטנדרט דה-פקטו של open table format, אבל הוא מגיע עם חוב תחזוקה קבוע — compaction, מחיקת snapshots ישנים, וניקוי קבצים יתומים. S3 Tables הוא סוג bucket חדש שלוקח את העבודה הזאת מהלקוח. שני שלישים מהסשן הם הסבר על המנגנון, והשליש האחרון הוא לקוח שמראה מה זה עשה לו בחשבון.

- **Iceberg הוא immutable, ולכן הוא צובר קבצים בקצב קבוע.** כל כתיבה יוצרת קבצי metadata וקבצי דאטה חדשים ומזיזה pointer — מה שנותן time travel, אבל "משפיע לי לראה על השילטות ותופס לי גם המון מקום בסטוראג'" [6:09]. זה לא תופעת לוואי, זה התכנון.
- **ההבדל בין self-managed ל-S3 Tables הוא בעלות תפעולית, לא בפורמט.** "בכל מקרה, האייסברג הוא אותו אייסברג ההבדל הוא מי אחראי על הטיפול סביב הטבלה" [9:11]. אותו Iceberg פתוח, אותו REST Catalog, אותם מנועים.
- **המספרים בדמו מראים את הפער בבירור.** אותו דאטה בדיוק נכתב לשני bucketים ללא התערבות: 20 קבצים ב-S3 Tables מול 8,849 ב-S3 רגיל, גודל קובץ ממוצע 7.24MB מול 24.8KB (סליד הדמו, [24:36]).
- **Intelligent-Tiering הוא הזרז לחיסכון, ואין לו מגבלת אובייקטים.** המידע יורד אוטומטית בין שלושה טירים; הסליד מציג 40% ואז 68% הוזלה, והדובר סיכם "עד 80 אחוזים" [16:54]. בניגוד ל-general purpose bucket — אין כאן עמלת ניטור לפי כמות אובייקטים.
- **Windward חתכה יותר מחצי מהעלות בלי לשנות את נפח התנועה.** הסליד שלהם מראה שכמות ה-requests אפילו עלתה (105 מול 100 בבסיס), בעוד compaction ירד ל-30 ו-monitored objects fee ל-4.
- **הרכיב הלא-מנוהל הוא זה שנשבר.** התקלה של Windward — קבצים כפולים ב-manifest, עלויות שהוכפלו, queries עם תשובות שגויות — נעוצה ב-JAR האופן-סורס היחיד שנשאר באחריותם [33:51].

שלושת הדוברים ושיוכם, כפי שהופיעו על סליד הפתיחה

2. הרקע: S3 כיסוד של הדאטה לייק

תומר איתן פתח בהקשר, לא במוצר. הנחת המוצא: כל ארגון עסוק ב-AI, אבל "נקודת המשק שלנו, אם זה האג'נטים או הצ'טבוטים השונים, תמיד תהיה רק קצה הקרחון" [1:31] — ומתחת למכסה המנוע מה שקובע הוא הדאטה. מכאן המשפט שהוא ציטט: "your AI is good as your data" [1:31].



התכונות שמוצגות כבסיס לדאטה לייק: durability, אבטחה, קליטת דאטה, שאילתות in-place, ואחזור מארכיון

S3 יצא ב-2006 — "זו פחות או יותר השנה שבה התחלנו לצרוך את ה-Wi-Fi הביתי" [9:11] — ושימש בעיקר ל-unstructured data: תמונות, מסמכים, סרטונים, לוגים. הסקייל הנוכחי שהוצג: למעלה מ-500 טריליון אובייקטים פרוסים גלובלית, "סדר גודל של 60 אלף אובייקטים לכל בן אדם על פני כדור הארץ", ו-200 מיליון בקשות בשנייה [3:02].

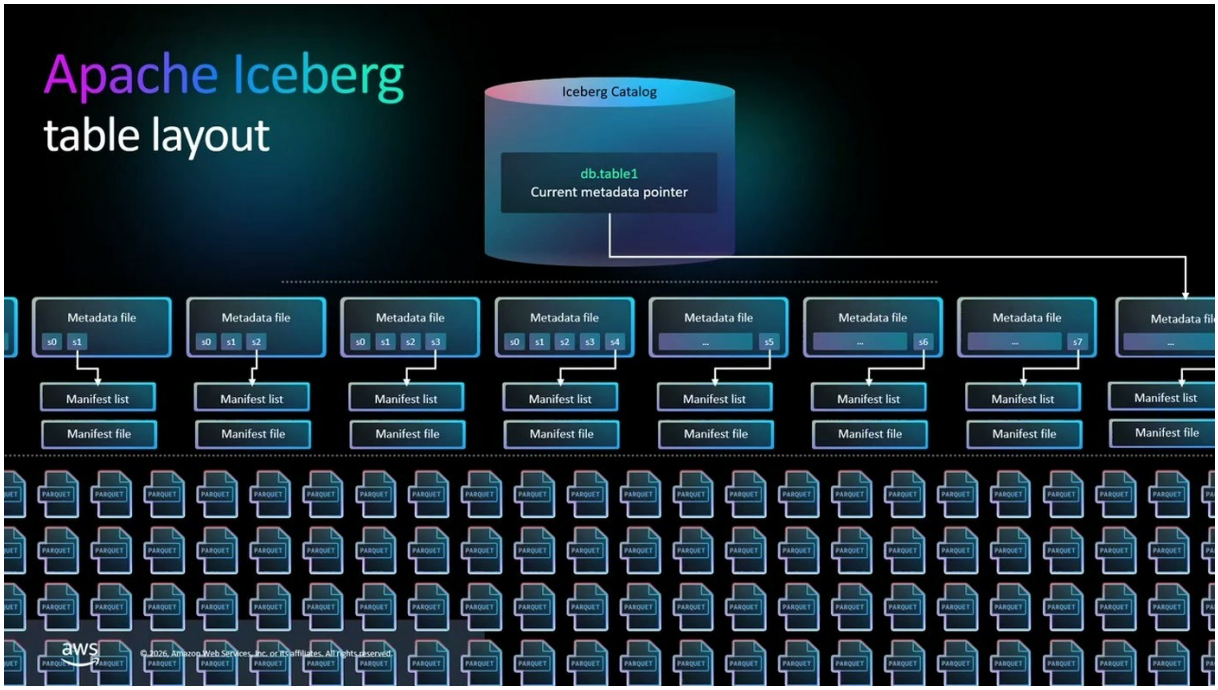
השינוי המהותי הוא בסוג הדאטה. יותר ויותר לקוחות מאחסנים structured data, ופורמט Parquet הוא הפופולרי מביניהם. הסליד הייעודי ממקד את זה: 20 מיליון בקשות בשנייה ומאות פטה-בייט שמשורתים מדי יום — עבור Parquet בלבד.

נתוני התעבורה על קבצי Parquet ב-S3, לצד הלוגואים של Hadoop ו-Spark

וכאן נולד הצורך: "אני רואה ב-day-to-day שלי לקוחות שמחסנים טבלאות של מאות טראבייט, פטאבייט של מידע, וכאשר מדובר על סקייל כל כך משמעותי, נדרשת איזושהי שכבת ניהול מאוד יעילה" [4:38]. השכבה הזאת היא Apache Iceberg.

3. איך Iceberg בנוי — ולמה זה חשוב

דן קולודני הסביר את המבנה בשלוש שכבות. המשפט שמסגר את כל הפרק: "אנחנו לא רוצים להתייחס לדאטה שלנו כמו אוסף של קבצים היינו שמחים להתייחס אליו כמו טבלה ופה בדיוק הפאצ'י אייסברג נכנס לתמונה" [4:38].



שלוש השכבות: קטלוג עם מצביע למצב הנוכחי, קבצי metadata ו-manifest, ומתחתם שדה קבצי Parquet

- **שכבת הדאטה:** קבצי Parquet שבהם שמור המידע עצמו.
- **שכבת המטא-דאטה:** קבצי metadata, manifest list ו-manifest file, שמאפשרים למנוע לדעת בדיוק אילו קבצי Parquet לסרוק.
- **הקטלוג:** מצביע לקובץ ה-metadata שמייצג את הגרסה הנוכחית של הטבלה. "אם יש שאילתה, אז היא עוברת לשכבת המטה דאטה, והיא עוזרת לה לדעת בדיוק איזה קבצי פרקט המנוע צריך לסרוק" [4:38].

העיקרון: אף פעם לא משנים במקום

"כל פעם שאני כותב מידע, ואייסברג הוא לא משנה את המצב הקיים, הוא כותב קבצי מטה דאטה חדשים, קבצי דאטה חדשים, והוא משנה את הפוינטר למצב החדש של הטבלה" [4:38]. זה מה שמייצר את היכולות שמוכרות מעולם הדאטאבייס ומגיעות כאן לדאטה לייק — ACID transactions, schema evolution, ו-time travel — ובאותה נשימה זה גם מקור הבעיה.

"מצד אחד, זה מאוד חזק, כי אני יכול עכשיו לבוא ולתחקר את הטבלה בכל נקודת זמן ולראות איך היא נראתה. מצד שני, זה משפיע לי לראה על השילטות ותופס לי גם המון מקום בסטוראג' ולכן אני חייב לבצע פעולות תחזוקה באייסברג" [6:09].

4. שלוש פעולות התחזוקה שחייבים לעשות

"אז אמרנו שיש פעולות החזוקה. שלוש פעולות עיקריות" [6:09].

- **File compaction —** איחוד קבצים קטנים לקובץ גדול. שתי סיבות: overhead על כל פתיחה וסגירה של קובץ, ומנוע התכנון של השאילתה שלא צריך "להבין על ממש מיליוני קבצים" [6:09].
- **Snapshot expiration —** כל כתיבה יוצרת גרסה חדשה של הטבלה. "אני צריך להחליט כמה ורז'נים אחורה אני רוצה לשמור, ואת השאר אני פשוט רוצה למחוק" [7:40].
- **Orphan file cleanup —** קבצים שנכתבו בתהליך שלא הסתיים כראוי, והקטלוג לא מצביע אליהם. "הם סתם תופסים למקום בסטורג' אני לא יכול לקרוא אותם, אבל אני פשוט צריך למחוק אותם" [7:40].

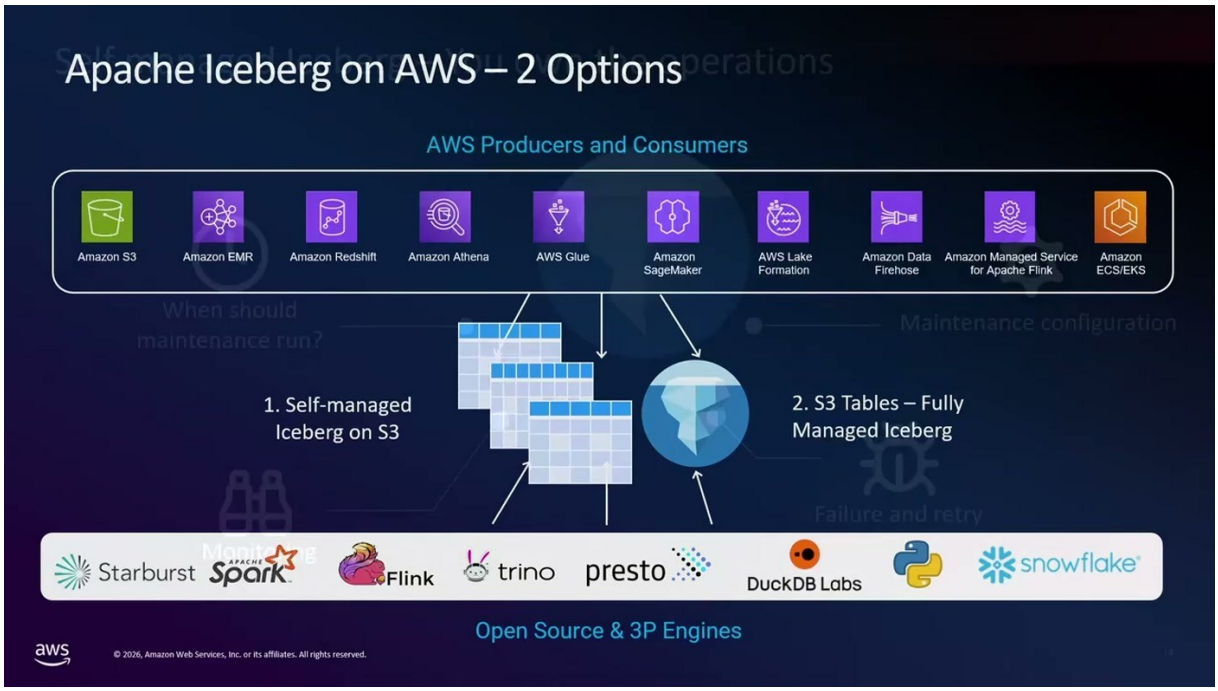
העלות האמיתית היא לא הפעולות עצמן אלא מה שעוטף אותן: scheduling, monitoring, ומנגנון retry כשמשהו נכשל. "ככל שיש לי יותר ויותר תבלאות זה כבר נהיה עומס טיפולי די גדול. תמיד בדמו אם נראה לנו שיש תבלאה אחת או שתיים אז זה מאוד קל אבל בעולם האמיתי יש עשרות או מאות תבלאות" [7:40].

ארבעת מוקדי האחריות שנשארים אצל מי שמנהל Iceberg בעצמו: תזמון, הגדרת תחזוקה, ניטור, וטיפול בכשלים

> **— דן קולודני, [7:40]** ואני רוצה לשאת לכם שאלה האם אני באמת חייב לעשות את הפעולות החזוקה האלה בעצמי והתשובה היא שכנראה שלא

שתי הדרכים

האפשרות הראשונה היא self-managed Iceberg על general purpose bucket: "יש לי הכי הרבה שליטה יש לי הכי הרבה גמישות אבל אני אחראי על כל טיפול הטבלה" [7:40]. השנייה היא S3 Tables, שבה "AWS לוקחת חלק מאוד משמעותי בכל מה שקשור לתחזוקה ולאופטימיזציה של הטבלה" [9:11].



אותה שכבת יצרנים וצרכנים — Athena, EMR, Redshift, Glue, SageMaker, Flink — מעל שתי האפשרויות, ומתחת מנועי קוד פתוח וצד שלישי

5. S3 Tables: סוג bucket חדש

ההיגיון שהוצג הוא היגיון מוכר של AWS: "לקחת איזושהי ארכיטקטורה מורכבת... ולהפוך אותה לשירות מנוהל מקצה לקצה, Fully Managed Service. אז עשינו את זה עם RDS, שירות הדאטאבייס המנוהל, או EKS, שירות הקוברנטיס המנוהל" [9:11]. התוצאה היא S3 Table Bucket — סוג bucket שונה מ-general purpose, ששומר על מאפייני ה-durability וה-scalability של S3 ומוסיף עליהם התאמות ל-Iceberg.

שלוש ההבטחות של השירות — ביצועים וסקייל, בקרות אבטחה פשוטות, ואופטימיזציית עלות אוטומטית — סביב S3 Table Buckets

ביצועים

המגבלה המוכרת ב-bucket חדש רגיל היא GET 5,500 ו-PUT 3,500 לשנייה. "הכפלנו את המספר הזה פי עשרה וכל בקט חדש שיוצרים של S3 טייבלס בעצם מאפשר לנו 55,000 גטים ו-35,000 פוטים מדי וואן" [13:50].

אבטחה

זו הנקודה החזקה יותר, ואף אחד לא נוטה לחשוב עליה מראש. ב-general purpose bucket, "מי שיש לו גישה לאותם bucketים יכול ככה להיכנס מתחת לאייסברג, מתחת לשכבת המטה דייטה ולשנות, לערוך, להזיז, בין אם זה קבצי המטה דייטה, קבצי הפרקט, קבצי המאניפסט והדבר הזה יכול לגרום או לחוסר קונסיסטנטיות או אפילו מצב גרעו יותר של דאטה קוראמשן" [10:42].

ב-S3 Tables אין גישה לאובייקטים מאחורי הקלעים. "כל טבלה היא First Class Citizen של AWS המשמעות היא שיש לה ARN ובאמצעות IAM אנחנו יכולים לתת הרשאות מי יכול ומי לא יכול לשנות טבלאות" [10:42]. מעליה, Lake Formation מאפשר fine-grained access control ברמת טבלה בודדת ואף ברמת עמודה [13:50].

מה "מנוהל" אומר — ומה לא

- **Iceberg נשאר פתוח.** יש תמיכה ב-Iceberg REST Catalog endpoint (IRC): "לא משנה באיזה קליינט שהוא Iceberg Compatible אתם משתמשים, כל עוד הוא תומך ב-API הסטנדרטי של Iceberg" — Flink, Presto, Trino, Spark [12:19].
- **כלי צד שלישי נתמכים.** Snowflake ו-Databricks הוזכרו כמי שנתמכים "כמעט מדי וואן" ויכולים לעבוד ישירות מול S3 Tables [12:19].
- **יש MCP Server ייעודי.** כלי שמאפשר לטבלאות Iceberg להתממשק ל-LLMs ולהיות מתושאלות בשפה טבעית — "אנחנו הופכים את טבלעות האייסברג שלנו לסוג של איזשהו knowledge base" [12:19].

שלוש אסטרטגיות compaction

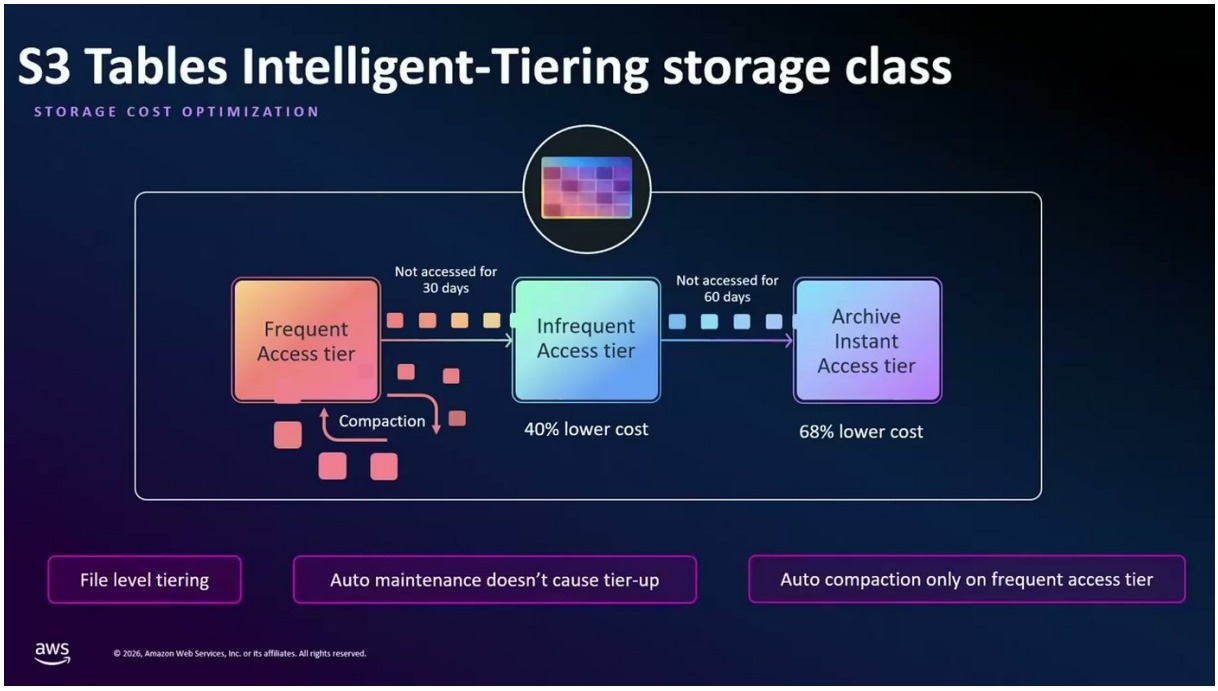
אין יותר צורך להרים compute node ולהריץ Spark job לכל טבלה [13:50]. מהקונסול או מה-API בוחרים אסטרטגיה לפי אופי ה-workload:

אסטרטגיה	מתי לבחור בה	מה היא עושה
Bin-pack	ברירת מחדל, ללא דפוס שאילתות מובהק	מאגד קבצים קטנים לקבצים גדולים יותר
Sort	תשאול יומיומי לפי עמודה ספציפית אחת	מאגד את הדאטה בסדר לוגי, כך שנפתחים פחות קבצים לשאילתה
Z-order	תשאול על פני כמה ממדים במקביל	מסדר את הדאטה לפי מספר עמודות יחד

גם התחזוקה השוטפת עברה לאוטומט: מגדירים minimum snapshots ו-maximum snapshot age, ומחיקת ה-snapshots הישנים והקבצים שאין אליהם הפניה קורית לבד. "אין יותר ניהול סזיפי של סנפשוטים" [15:21]. מנגנון ה-unreferenced file removal מקביל ל-lifecycle של bucket רגיל: unreferenced days שקול ל-deletion marker, ו-non-current days למחיקה לצמיתות [16:54].

6. Intelligent-Tiering: מאיפה מגיע החיסכון

הטענה: Iceberg הוא מקרה קלאסי ל-Intelligent-Tiering, כי דפוס הגישה שלו צפוי. "אנחנו מבצעים אינג'שן של דייטה בצורה מאוד מאוד אינטנסיבית, וככל שהזמן עובר, ככה דפוס הגישה יורד... הסיכוי שאני אשלוף מידע היסטורי, ככל שעובר הזמן הופך להיות פחות ופחות תדיר" [16:54].



המסלול בין שלושת הטירים ושתי סיפי הזמן, ומתחת שלוש ההערות התפעוליות — טיירינג ברמת קובץ, תחזוקה שלא מחממת דאטה, ו-compaction רק בטיר החם

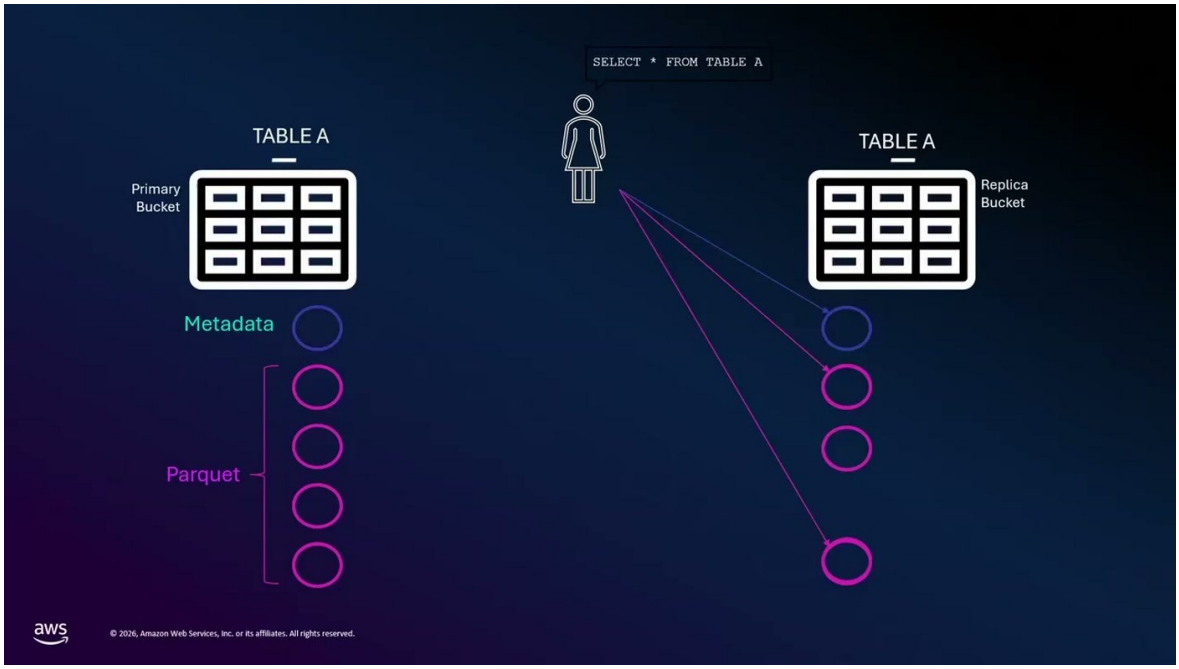
- **המסלול:** כל מידע חדש נכתב ל-Frequent Access tier. ללא גישה במשך 30 יום — מעבר ל-Infrequent Access tier. עוד 60 יום ללא גישה — מעבר ל-[16:54] Archive Instant Access tier.
- **המספרים על הסליד:** 40% הוזלה בטיר השני, 68% בשלישי. הדובר סיכם את זה כ"עד 80 אחוזים" [16:54] — פער שכדאי לקחת בחשבון בכל תחשיב.
- **התחזוקה לא מחממת את הדאטה.** "כל תהליכי הקומפקשן שציינתי רצים לנו אך ורק ב-Frequent Access Tier כדי לא לגרום לחימום של מידע" [18:27] — כלומר דאטה שכבר ירד לטיר קר לא יעלה חזרה בגלל פעולת מערכת.
- **אין עמלת ניטור לפי אובייקטים.** "פה אין שום מגבלה על ניתור לפי כמות האובייקטים אז אין שום סיבה למה לא להגדיר את זה כבר מדי וואן" [18:27] — בשונה מ-Intelligent-Tiering ב-general purpose bucket.
- **אבל — זו החלטה חד-פעמית.** "לא ניתן באמצעות life cycle לנייד את הטבלות מסטנדרט לintelligent tearing זה משהו שנקבע בזמן יצירת הטבלה או בזמן יצירת הבקט" [18:27]. מעבר בדיעבד דורש rebuild של הטבלה.

החלטה שקשה לחזור ממנה הנקודה האחרונה היא ההשלכה התכנונית היחידה שיש לה מחיר ממשי: בחירת storage class בזמן יצירת ה-bucket. Windward נאלצה לבצע rebuild על 14 שנות דאטה בדיוק בגלל זה — ראו פרק 10.

7. רפליקציה שמודעת ל-Iceberg

תומר שאל את הקהל מי הגדיר רפליקציה לטבלאות Iceberg שהוא מנהל בעצמו. "אני רואה מעט ידיים ואני מניח שהסיבה היא מורכבות" [18:27].

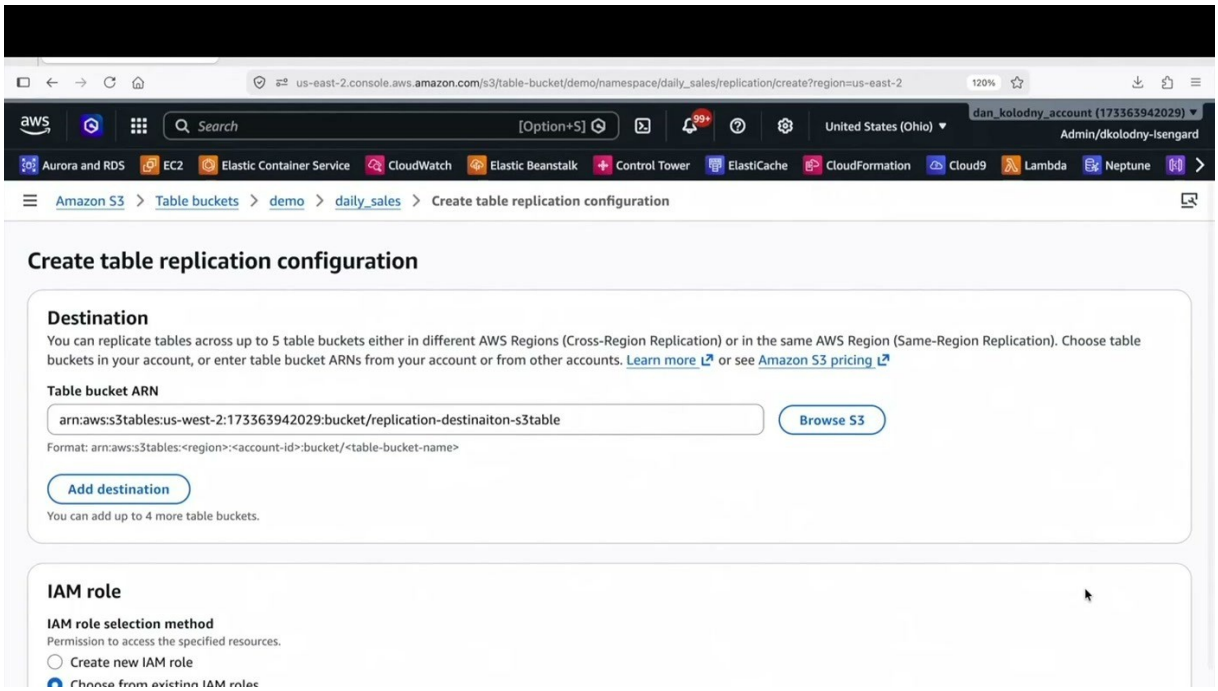
המורכבות ספציפית: ב-metadata של Iceberg יש הפניה קשיחה לקבצי ה-Parquet. "גם אם עכשיו אנחנו נרפלק את האובייקטים באמצעות המנגנון של Astry Replication עדיין תתבצע הפנייה לגרצי הפרקט שנמצאים לי בסורס" [18:27] — כלומר ה-replica מצביע חזרה למקור. כדי שזה יעבוד, למשל לצורכי DR, צריך תהליך metadata rewrite שמפנה מחדש את כל האובייקטים ליעד. בעיה שנייה: "S3 Application עובר בצורה של שגר ושכח. ובאייסברג יש חשיבות מאוד מאוד גדולה לקונסיסטנטיות" [18:27].



הבעיה במצב הנאיבי: שאילתה מול ה-replica ממשיכה להפנות לקבצי Parquet ב-bucket המקורי

הפתרון שהוכרז ב-re:Invent האחרון: "הכרזנו על תמיכה ברפליקשן עבור S3 Tabels רפליקשן שהיא מה שנקרא [19:57] "Iceberg Wear, Iceberg Consistent — רפליקציה מודעת ל-Iceberg, אל bucket יעד שיכול לשבת בחשבון אחר, באזור אחר, או בשניהם. מסך הקונפיגורציה בקונסול מציין שאפשר לרפלק לעד חמישה table buckets.

שלוש ההצדקות שהוצגו: ביצועים — עותק לוקלי קרוב לאנליסטים ולאנשי הדאטה סיינס עם latency נמוך יותר; רגולציה וקומפליינס — עותק נוסף לצורכי עמידה בתקינה; והגנת דאטה — התאוששות מ-drop table בטעות אנוש או מפעולה זדונית שמצפינה או מוחקת קבצים [19:57].



מסך ההגדרה בקונסול: ARN של ה-table bucket היעד באזור אחר, ובחירת ה-IAM role להרשאות הכתיבה

8. הדמו

דן חילק את הדמו לשלושה חלקים: יצירת טבלה מהקונסול, השוואה בין S3 Tables ל-bucket רגיל, ותשאול בשפה חופשית.

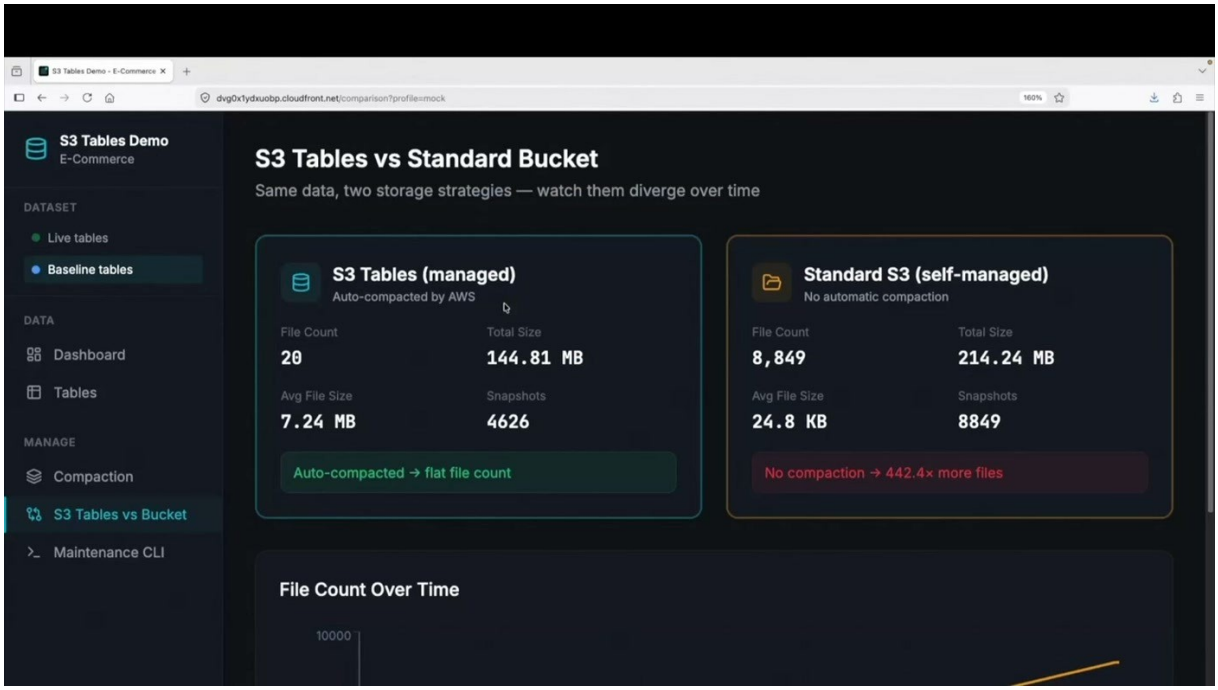
חלק א' — יצירת טבלה

הזרימה שהוצגה: Create table bucket, בחירת storage class — "אני בוחר intelligent tiering, ממליץ תמיד לבחור [21:29] "intelligent tiering — יצירת namespace ("תחשבו על זה עוד קונטיינר לוגי, עוד כמו פולדר שבתוכו אני רושם את הטבלעות" [21:29]), ואז CREATE TABLE מתוך Athena. אחריו INSERT INTO של תשע רשומות מכירה ושאילתת אגרגציה פשוטה.

במסך המיינטננס של הטבלה הוצג ה-target file size — 512MB כברירת מחדל, שדן שינה ל-64MB — ובחירת אסטרטגיית ה-compaction מבין bin-pack, sort ו-Z-order [23:06]. "אז ראיתם כמה קל היה לייצר טבלת אייסברג" [23:06].

חלק ב' — ההשוואה

זה החלק המשכנע. אותו דאטה בדיוק נכתב פעם אחת ל-general purpose bucket ופעם אחת ל-S3 Table, ואז — במילותיו — "ולא נגענו".



לוח ההשוואה בדמו: 20 קבצים מול 8,849, וגודל קובץ ממוצע של 7.24MB מול 24.8KB

> **— דן קולודני, [24:36]** ב-S3 Table יש לנו 20 קבצים, ב-S3 Standard יש לנו כמעט 9,000 קבצים... הקו הכתום זה ה-General Purpose Bucket, זה מראית מספר הקבצים בבקט. אתם רואים שהוא רק עולה, עולה ועולה. והקו הכחול זה ה-13 טייבל. אתם רואים שכל כמה זמן הוא עושה קומפקשן ומספר הקבצים מצטמצם. וזה אני אומר, לא נגענו. הגדרנו את הטייבל וזה עושה את זה לבד.

חלק ג' — תשאול בשפה חופשית

דרך Amazon Quick, עוזר ה-Gen AI של AWS, דן ביקש רשימת S3 Table Buckets ב-us-east-1, ואז תיאור של תוכן ה-bucket של ה-e-commerce — שהחזיר את הטבלאות users, products ו-orders עם העמודות שלהן. אחר כך שאלה עסקית: שלושת המוצרים הנמכרים ביותר.

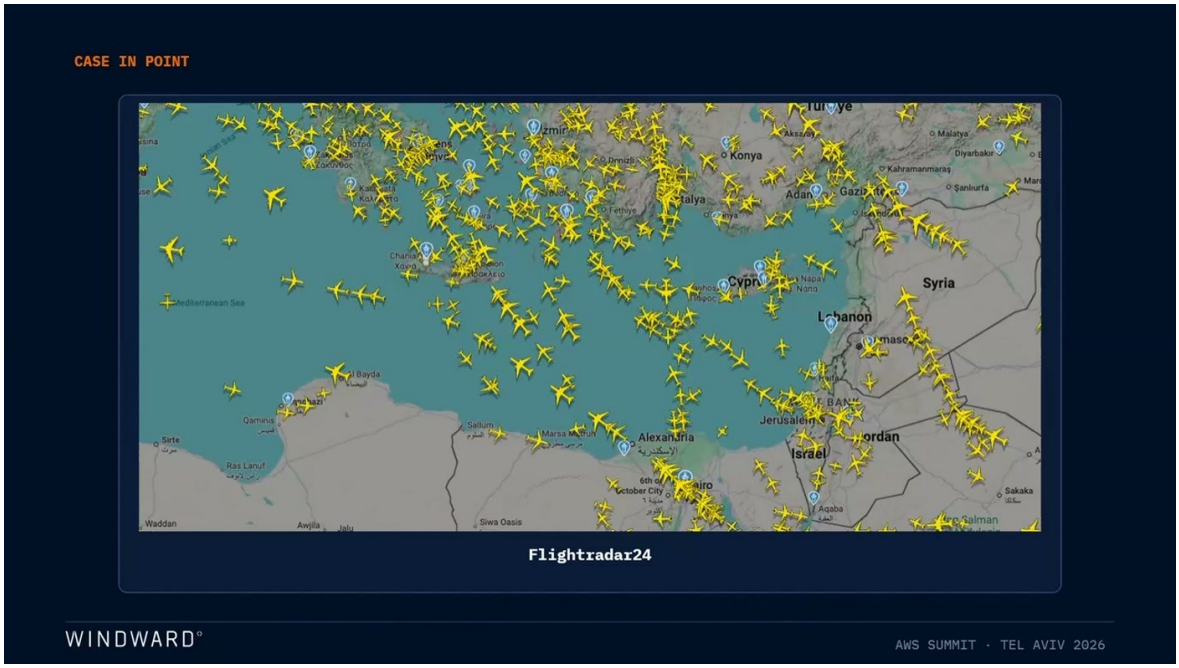
נקודה שדן הדגיש: "הוא מראה לי את השילטה, זה לא בלקבוקס. אני יכול ממש לראות את הסינטקס, אני יכול לראות איך הוא הגיע לתשובה, ויש לי פה ויזביליות" [26:09].

השאילתה האחרונה היא זו שממחישה את הערך התפעולי: הצגת פעולות התחזוקה שבוצעו על כל טבלה ב-bucket — מתי רץ compaction, מתי unreferenced file removal, ומתי snapshot expiration, עם תאריכים מדויקים [27:40]. "אם יש לנו עשרות ומאוד טבלעות, זה מאוד מאוד משמעותי ואת כל זה אנחנו מקבלים "out of the box" [27:40].

9. Windward: מה הבעיה שהם פותרים

ליאור רסיסי, R&D Director ב-Windward: "ווינדוורד היא חברת AI בעולמות הים. התפקיד שלנו הוא לבוא ולעזור ללקוחות שלנו, בין אם אלה ממשלות, צבאות, חברות אנרגיה, בנקים, ולדעת מה היה, מה קורה ומה יקרה בתקווה בעולמות הים, ובעיקר למה" [27:40].

מקור הדאטה המרכזי הוא פרוטוקול AIS — Automatic Identification System — פרוטוקול בטיחות שבו ספינות משדרות את מיקומן. כל שידור כזה נקרא אצלם "בליפ" (blip). המזהה הייחודי של כלי שיט נקרא MMSI, "דומה פחות או יותר ללוחית רישוי של רכב" [29:13] — פרט שיחזור בהמשך כמפתח פרטישן.



האנלוגיה שליאור השתמש בה: מה שאפשר לראות על מפת תנועה חיה — והמקבילה הימית שעליה בנויה Windward

הדוגמה שהוא הביא: 28 בפברואר, אזור המפרץ הפרסי. "אפשר לראות שיש כאן ספינות שמשגרות את עצמנו במהירות מדהימה, ויצרות בדיוק עיגול מושלם, במקרה הזה לחופי קטר. אז לא, זה לא באמת מה שקורה. זה תוצאה של תופעה שקוראים לה GPS ג'מינג" [29:13]. השידורים ניתנים לזיופים ולשיבושים — ולכן הערך של Windward הוא בשכבת העיבוד, לא באיסוף.

הפייפליין המקורי

התהליך התחיל מקבצי CSV ב-S3, ועליהם סדרת אפליקציות Spark שבהן ה-output של אחת הוא ה-input של הבאה. בסוף התהליך מתבצע downsampling: "אנחנו שומרים רק את המידע שהוא מאוד מעניין, ואותו מחצינים לכלל החברה. כל מה שעד לשם הוא מידע פנימי, איזה שהם תוצרי ביניים, שהם אפילו סייס וויז, לא בצורה של פרקט" [30:44].

סדרי הגודל: "אנחנו יכולים לראות בהבדל בכמולות, שהרדנו מ-500 מיליון בליפים ביום, בסדר גודל של 20 מיליון חדשים ביום. מאוד משמעותי כשאנחנו עובדים עם 14 שנים של דאטה ואף יותר" [30:44].

10. Windward: המסע ל-S3 Tables

שלב 1 — self-managed Iceberg

הטריגר היה בקשה מצוות המחקר שרצה גישה לתוצרי הביניים הפנימיים. "נענו קצת בחוסר נוחות בכיסא, כי בכל זאת זה CSV שלא מאופתמים לשום קווירי שהוא" [30:44]. הפתרון הראשון היה Iceberg self-managed מעל — S3 S3 Tables עדיין לא היה קיים — עם אפליקציית Spark יומית.

"בשלב הזה זה הייתה טבלה אחת, שהיא התתקנה אחת ביום. הכל טוב ויפה. כתבנו תהליכים שעשו את הקומפקשן, שמחקו סנפשוטים ובעיקר שמחקו אורפן פיילס. מי מכם שעובד עם ספרק יודע שיש הרבה מאוד קבצים כאלה" [32:17].

שלב 2 — סטרימינג ו-S3 Tables

"אבל ברגע שזה טוב רוצים יותר" [32:17]. במקביל הארכיטקטורה עברה מ-batch לסטרימינג: אפליקציות ה-Spark הפכו למיקרו-שירותים על EKS, וקבצי ה-CSV הוחלפו בטופיקים ב-MSK. כשהגיעה הבקשה לדאטה הנוסף, השאלה הייתה איפה לשמור אותו — "ובדיוק בתזמון מושלם, ב-Amazon Reinvent 24, יצאו עם [32:17] "S3Tables.

החיבור בין MSK ל-S3 Tables נעשה דרך MSK Connect. וכאן ליאור עשה את ההבחנה שהיא לב הסיפור: "אנחנו נראה שיש לנו מיקרוסרויסים מנוהלים, יש לנו קאפקא מנוהל, יש לנו S3 טבל מנוהל, יש לנו MSK Connect מנוהל, והדבר היחיד שלא מנוהל זה הג'ר שהוא בנדל לתוך אותו MSK Connect... ופה אנחנו נותנו את ה-open source ג'ר של אייסברג" [33:51].

שלב 3 — התקלה

התלונות הגיעו בהדרגה: "אנחנו ראינו שהעלות עלתה בצורה הרבה יותר דרמטית ממה שרצינו... די מהר העלות שלשה את עצמה". במקביל — "תלנועות הקריות על דאטה חסר במערכת, או שאותן פואריז החזירו תשובות לא נכונות", ו-snapshots שנערמו ולא נמחקו [33:51].

האבחון, שבוצע יחד עם ה-Specialists של AWS, הצביע בדיוק על הרכיב היחיד שלא היה מנוהל.

> **— ליאור רסיסי, [33:51]** מצאנו שהבעיה הייתה ברכיב היחיד, שהוא לא manage, ב-JAR של ה-Iceberg שהיה לנו. באמת, בג פתוח שראינו בגיטאב, שמדבר על בעיה של קבצים כפולים בתוך המניפסט. וכמו שדן הסביר, ברגע שה-manifest is corrupted, כל ה-queres בעצם הם manipredictable.

שלב 4 — האופטימיזציות

- **צמצום הקונקטורים ב-MSK Connect** והורדת דחיפות הכתיבה, כך שהקבצים שנוצרים גדולים יותר [33:51].
- **ביטול הניהול הכפול של הטבלאות.** הדאטה נוהל בשתי צורות — פרטישן לפי תאריך ופרטישן לפי MMSI. "אם נשתמש בזיסורטינג, אנחנו נוכל, בקומפקשן כזה, אנחנו נוכל לחסוך את הניהול הכפול של הטבלאות האלה" [35:24].
- **מעבר ל-Intelligent-Tiering.** שינוי שדרש ממילא rebuild של הטבלה — "על כל 14 שנים של הדאטה". ההכרזה על Intelligent-Tiering ב-re:Invent 2025 סיפקה את הסיבה לעשות את זה פעם אחת ולפתור הכול יחד [35:24].

התוצאות

- **פלואו ששלף דאטה של שנים: מארבע שעות ריצה ל-15 דקות.** "אנחנו ירדנו בעצם ממשהו כמו 4 שעות ריצה ל-15 דקות" [36:57].
- **כמות הדאטה הנסרקת: מ-30GB ל-800MB, בזכות Z-order.** "שזה אומר שגם העלות העתינה הייתה הרבה יותר נמוכה, וגם שהקוראים זה יהיו הרבה יותר מהירים" [36:57].
- **התחזוקה יצאה מהאחריות שלהם.** "כשזו עוד היה בטבלה אחת, זו עוד היה נחמד, יכולנו לעשות את זה בעצמנו, אבל ברגע שעברנו לטבלעות הרבות שאנחנו צריכים לנהל, כל זה שזה לא מנוהל על ידינו, זה דבר שהוא מאוד מאוד חישוב" [36:57].

פילוח החיסכון לפי קטגוריה, כשכל עמודה מנורמלת לעלות שלה לפני השינוי (100%)

הסליד הוא הראיה החזקה ביותר בסשן כולו, כי הוא מפריד בין "ירדה התנועה" ל"ירדה העלות". עמודת ה-requests עלתה ל-105 — כלומר נפח העבודה לא קטן — בעוד storage ירד ל-74, compaction ל-30, ועמלת ה-monitored objects ל-4. ליאור: "אנחנו יכולים לראות בעמודה הימנית את כמות הריקווסטים שלא ירדה, גם אפילו גדלה קצת, כלומר כמות הדאטה לא השתנתה. מה שכן, העלות שלנו על כל אחד מהפילרים... ירדה בצורה סופר דרמטית" [36:57]. כותרת המשנה על הסליד מסכמת: Same traffic, less than half the cost.

11. מה לוקחים מכאן

- **השאלה הראשונה בכל פרויקט Iceberg היא מי מתחזק.** לא הפורמט, לא המנוע — התחזוקה. אם התשובה היא "אנחנו", צריך להעמיד תשתית של scheduling, monitoring ו-retry לכל טבלה, ולסבול אותה בכל טבלה נוספת.
- **מספר הקבצים הוא מדד הבריאות הכי זול למדידה.** ההשוואה בדמו — עקומה שרק עולה מול עקומה משוננת שחוזרת למטה — היא בדיוק מה שאפשר לנטר גם ב-Iceberg self-managed היום, לפני שמחליטים כלום.
- **בחירת storage class היא החלטה בזמן יצירה.** אי אפשר להעביר טבלה ל-Intelligent-Tiering ב-lifecycle. ההמלצה שנשמעה מהבמה הייתה לבחור Intelligent-Tiering מהיום הראשון, במיוחד כשאין עמלת ניטור לפי אובייקטים ב-S3 Tables.
- **רפליקציה של Iceberg היא לא רפליקציה של אובייקטים.** עותק של הקבצים בלי metadata rewrite לא נותן DR אמיתי. מי ששוקל תוכנית המשכיות על טבלאות Iceberg צריך לוודא שהפתרון מודע ל-Iceberg.
- **"מנוהל" נקבע לפי הרכיב החלש בשרשרת.** הלקח המרכזי של Windward: ארכיטקטורה שכולה שירותים מנוהלים נשברה בגלל JAR אחד. שווה למפות היכן נשארו רכיבים באחריות עצמית, גם כשהם קטנים.
- **צריך להבין את Iceberg לעומק גם בשירות מנוהל.** הקפיצה הגדולה של Windward בביצועים הגיעה מבחירת Z-order נכונה לפי דפוס התשאול — החלטה שאף שירות לא יקבל במקומכם.

> **— ליאור רסיסי, [38:28]** דבר האחרון הוא שאופן סורס הוא טוב עד שהוא משתבש, ואז אין איזה שהם אבב אבימא שיכולים לעזור. טוב שיש מנאג' סרוויסז של אמזון, וטוב שיש את הספשייליסט שעזרו לנו לפתרות כל הבעיות.

מילון מונחים

מונח	מה זה	איפה הופיע
Apache Iceberg	Open table format שמוסיף שכבת metadata מעל קבצי Parquet ומאפשר ACID, schema evolution ו-time travel	פרק 3
S3 Tables Bucket	סוג bucket ייעודי ל-Iceberg, שבו AWS מבצעת את התחזוקה ואין גישה ישירה לאובייקטים	פרק 5
Compaction	איחוד קבצים קטנים לגדולים, בשלוש אסטרטגיות: bin-pack, sort, Z-order	פרקים 4, 5
Snapshot expiration	מחיקת גרסאות ישנות של הטבלה לפי מדיניות של מספר snapshots וגיל מקסימלי	פרקים 4, 5
Orphan / unreferenced files	קבצים שנכתבו אך הקטלוג לא מצביע אליהם — תופסים מקום ואינם קריאים	פרק 4
IRC — Iceberg REST Catalog	ה-API הסטנדרטי שדרכו כל קליינט תואם Iceberg עובד מול הטבלאות	פרק 5
Intelligent-Tiering	מעבר אוטומטי של קבצים בין Frequent, Infrequent ו-Archive Instant Access לפי דפוס גישה	פרק 6
Metadata rewrite	הפניה מחדש של המטא-דאטה לקבצי היעד, תנאי לרפליקציה שמישה של Iceberg	פרק 7
AIS / blip / MMSI	פרוטוקול שידור מיקום של ספינות, שידור בודד, ומזהה ייחודי של כלי שיט	פרק 9
MSK Connect	שירות הקונקטורים המנוהל של Kafka ב-AWS, ששימש את Windward לכתיבה מטופיקים ל-S3 Tables	פרק 10

הסשן ננעל בהזמנה לשאלות ובקוד QR למשוב. שלושת הדוברים נשארו באולם לשאלות אישיות.