אוטומציה — Agentic Lakehouse של הנדסת דאטה על AWS

tlv-ant306 — סיכום סשן, AWS Summit Tel Aviv 2026

Barak Edward, Solutions Architects, AWS | Ruli Weisbach, EVP R&D-ו Mor Aish, ו-Alicia Plotnikov
AIDevEx Team Lead, AppsFlyer

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

מסלול: AWS For Data and Analytics | רמה: 300 — Advanced

על המסמך הזה

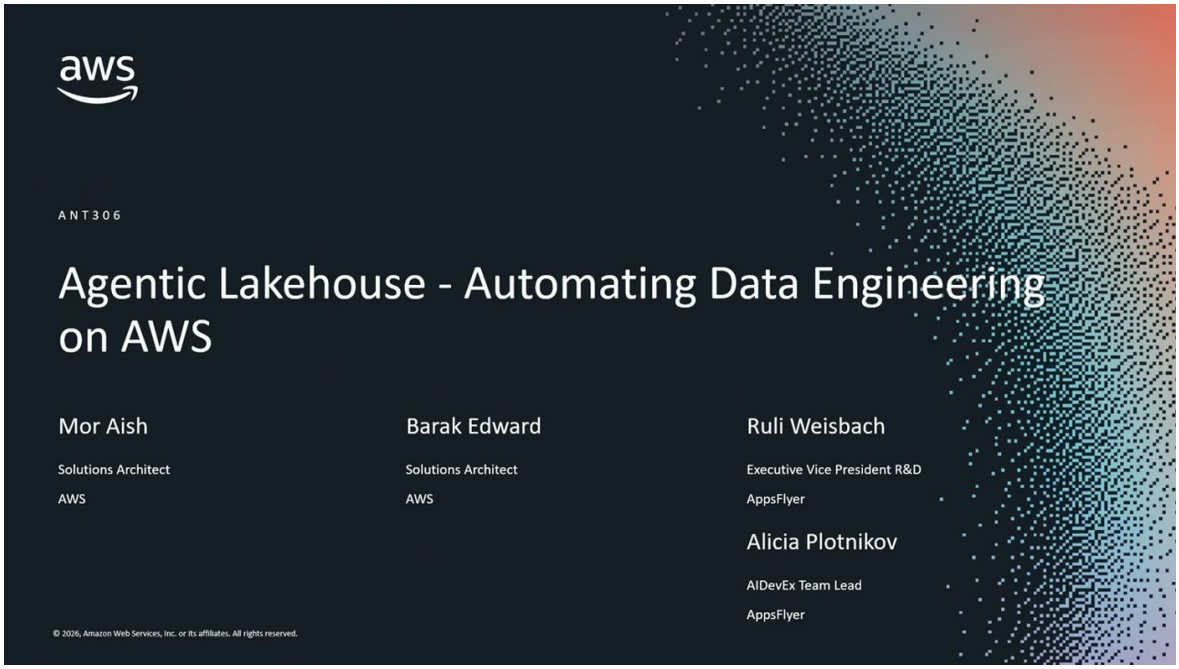
המסמך מסכם את הסשן ANT306 מתוך AWS Summit Tel Aviv 2026, בהתבסס על ההקלטה המלאה שפורסמה באתר הסאמיט. הוא נבנה מתמלול אוטומטי של ההקלטה ומהסליידים שחולצו מהווידאו עצמו, כך שאפשר לחזור לתוכן בלי לצפות שוב בכל הסשן. חותמות הזמן בגוף המסמך מפנות לנקודה המדויקת בהקלטה.

הסשן בנוי משלושה חלקים ברורים: רקע מושגי (Mor), דמו מוקלט של תהליך עבודה אג'נטי מלא (Barak), ומקרה בוחן מהשטח של AppsFlyer (Ruli ו-Alicia). כל חלק עומד בפני עצמו, ואפשר לקרוא אותם בנפרד.

איך לקרוא את זה

- **פרק 1 — תקציר מנהלים:** חמש דקות קריאה. מה הוצג ומה שווה לקחת מזה הלאה.
- **פרקים 2–3 — הרקע:** למה היום-יום של מהנדס דאטה נשחק, מהו agentic loop, ולמה MCP הוא הפתרון לבעיית האינטגרציות.
- **פרקים 4–6 — הדמו:** חברת DataFlow הפיקטיבית, הבעיה בדשבורד, ושבעת הצעדים שהסוכן ביצע ב-Kiro מול Redshift, Athena, Glue ו-S3 Tables.
- **פרק 7 — מהתגובה לפרודקשן:** סוכן אירועים על AgentCore שמגיב לתקלות בעצמו, ושלושת כללי האצבע לפרודקשן.
- **פרקים 8–9 — AppsFlyer:** הסקייל, ה-Logical Data Layer, ולמה סוכן מטא-דאטה אחד פתר בעיה שארגון הדאטה כולו לא הצליח לפתור.
- **פרק 10 — מה לוקחים מכאן:** מסקנות, נקודות להמשך ומונחון.

הערת מקור המסמך הופק מתמלול אוטומטי של ההקלטה. התמלול מדויק בקטעים העבריים, אך מונחים לועזיים מופיעים בו בשיבוש פונטי (למשל "אג'נט קור" = AgentCore, "אייסברג" = Iceberg, "עתינה" = Athena) ותוקנו כאן מול הסליידים. הציטוטים אומתו מול ההקלטה בחותמת הזמן המצוינת ומובאים כלשונם, כולל שיבושי התמלול. מספרים שמופיעים על הסליידים צוינו ככאלה.



שלושה דוברים משני ארגונים — שני Solutions Architects מ-AWS ושני אנשי AppsFlyer

1. תקציר מנהלים

הסשן לא ניסה לשכנע שסוכני AI עובדים. הוא ניסה להראות שהם כבר מספיק פשוטים כדי להכניס אותם למחזור העבודה היומי של צוות דאטה — ושמי שלא עושה את זה משלם על כך בשעות אדם. שני החלקים המעשיים, הדמו של AWS ומקרה הבוחן של AppsFlyer, מגיעים לאותה מסקנה משני כיוונים שונים לגמרי.

- **הבעיה שהוצגה היא עסקית, לא טכנית.** צוות של שישה מהנדסי דאטה שאחראי על לפחות 40 pipelines מבלה את השעות 9:00–12:00 בכיבוי שריפות. Mor ציטטה מהנדסים שראיינה: בניית pipeline חדש לוקחת "בין שבועיים לשלושה", והתובנות מגיעות "חמישה עד שבעה ימים מאחור" [1:30].
- **MCP הוא מה שהופך את זה לישים.** במקום N×M אינטגרציות בין סוכנים לכלים, כל סוכן וכל שירות מתחברים ל-MCP בלבד: "הגענו לאן פלוס הם ירדנו בסדר גודל ומורכבות לינארית" [6:03]. בפועל הדמו כולו רץ על שלושה MCP servers בלבד.
- **הדמו הוא תהליך שלם, לא שאילתה בודדת.** שבעה צעדים — קריאת סכמה ב-Redshift, קריאת טבלת Iceberg דרך Athena, כתיבה והרצה של שני Glue jobs, יצירת materialized view, וחיבור מחדש של ה-dataset ב-QuickSight — כולם בשפה חופשית מול Kiro, עם אישור אנושי בין צעד לצעד.
- **אותו סט כלים משרת גם תגובה אוטומטית לתקלות.** ארכיטקטורת האירועים שהוצגה (EventBridge → AgentCore → Lambda) מחוברת "בדיוק לאותם שלושת MCP סרברס" שהסוכן ב-Kiro משתמש בהם [23:01], ומפרסמת אבחון והצעת תיקון לערוץ Slack.
- **AppsFlyer הראו שהצוואר האמיתי הוא מטא-דאטה, לא דאטה.** אחרי שאיחדו את הדאטה תחת Logical Data Layer, הסקייל החזיר את הכאוס: "פתאום הסדר שעשינו נהיה לאי סדר" [32:12]. הפתרון היה סוכן מטא-דאטה אחד — "one metadata agent to rule them all" [36:49].
- **המספר היחיד שנמדד בפועל הוא של AppsFlyer.** לפי הסלייד: פי 2 משתמשים שבועיים פעילים בשלושה שבועות, ו-80% אימוץ. Alicia הוסיפה שבערוץ ה-MCP ראו "אפילו, כפול 3" [36:49].
- **המסר החוזר: הסוכן לא מחליף את המהנדס.** Barak חזר על מה ש-Mor אמרה בפתיחה: "Agentic AI לא מחליף את מהנדס הדאטה. הוא משחרר אותו להתמקד במה שהארגון הגדיר לו כחשוב" [24:32].

2. הבעיה: חמישה פילרים של אובדן פרודוקטיביות

המסגרת של הסשן: lakehouse שסוכנים מקיפים אותו ופועלים עליו

Mor פתחה בסצנה מוכרת — מגיעים בבוקר עם תוכנית ליום, ואז מגיע המייל: "פייפליין קרס, נתונים לא יצטקנו כמו שצריך, הדוח של ההנהלה, רק כולם בפניקה" [0:00]. משם היא עברה לנתונים.

מחקר IDC מנובמבר 2025, כפי שהוצג, מצא שארבע מתוך שש יכולות קבלת החלטות מרכזיות בארגונים משתפרות משמעותית עם AI agents: כ-70% שיפור בארגון ובאנליזה של דאטה, ובין 60% ל-70% שיפור בניטור, בלמידה, בסימולציות ובתרחישי what-if [0:00, 1:30].

> **Mor Aish [1:30] —** ואלו לא בעיות טכניות, אלו בעיות עסקיות שמתרגימות בסופו של דבר להפסדים כספיים אמיתיים.

חמשת הפילרים

- **Context switching:** מתחילים משימה, נכנסת הודעה ב-Slack על תקלה, ואחרי הטיפול "והשכחתם איפה הייתם והתחלתם את הכל מההתחלה" [1:30].
- **Data quality issues:** נתונים חסרים ב-data lake, ערכים שגויים, טיפוסים לא מתאימים [3:00].
- **Error handling ו-debugging:** משהו נשבר, וצריך לעבור בין מערכות ולוגים כדי למצוא למה [3:00].
- **ניהול משאבים:** ה-pipeline רץ לאט מדי, ואיך עושים אופטימיזציה [3:00].
- **חיפוש אחר תיעוד:** האם מישהו כבר עשה את זה, ואיפה ה-best practices [3:00].

ההשוואה שהיא הציגה הייתה דו-טורית. בצד המסורתי: להקים S3 buckets ידנית, לכתוב Glue job מאפס, לעשות deploy, לתזמן, להריץ, לתקן שגיאות, להקים Crawler ולקנפג אותו. בצד האג'נטי: לבקש בשפה טבעית "מצא איפה נמצאים נתוני לקוחות", "צור Glue job לעיבוד הנתונים האלו", "הרץ את ה-job, נטר ותקן שגיאות", "צור Crawler שיקטלג את הפלט" [3:00]. השאלה שהיא הפנתה לקהל הייתה קצרה: "יש פה מישהו שמעדיף את צד שמאל" [3:00].

> **המסר המרכזי** "האג'נט לא בא להחליף את המאנדס להפך הוא בא לשחרר אותו לעבודה אסטרטגית" [3:00]. המשפט הזה חזר שלוש פעמים בסשן — פעם אחת בכל חלק.

3. איך זה עובד: agentic loop ו-MCP

הלולאה

Mor פירקה את ה-agentic loop לשלבים: הסוכן מקבל קלט מהמשתמש, עובר למצב reasoning מול מודל שפה כדי להבין מה נדרש, מחליט באילו כלים להשתמש, מריץ אותם, מקבל תוצאות — "ואז, שימו לב, זה החלק החשוב הוא ממשיך לחשוב עם המידע החדש" [4:31]. הלולאה חוזרת עד שהתשובה מוכנה.

> **Mor Aish [4:31] —** זו לא פעולה חד פעמי, זו חשיבה מורכבת ורב שלבית.

ארבעת הרכיבים שבתוך הסוכן, כפי שהוצגו: זיכרון (קצר טווח להקשר הסשן, ארוך טווח לאירועים היסטוריים ולעובדות), כלים לחיבור לשירותים, מוח מבוסס LLM שמתכנן את הצעדים, ופרסונה — תפקיד, מטרה והוראות. "וזה מה שהופך אותו לא רק לפלי אוטומציה פשוט אלא למשהו שבאמת מבין את ההקשר ולומד מן העבר" [4:31].

בעיית N×M

כאן הגיע החלק המשכנע ביותר ברקע. סוכן צריך להתחבר ל-data warehouse, לאינטרנט, ל-data lake — ולכל אחד API, סטנדרטים וחוקים משלו. עם N סוכנים ו-M כלים נדרשות N×M אינטגרציות: "היא לא סקיילבילית, והיא בעיקר בעיקר מתכון לקיאוס" [6:03].

הפתרון שהוצג הוא Model Context Protocol — MCP. כל סוכן מתחבר ל-MCP, כל מערכת מתחברת ל-MCP, והמספר יורד ל-N+M.

> **Mor Aish [7:35] —** אפשר לחשוב על זה כפורט USB-C לישומי אג'נטיק AI. והחלק הכי טוב זה קוד פתוח.

בצד הלקוח יושבת אפליקציה אג'נטית — SageMaker Unified Studio, QuickSight, Kiro, Claude, Cursor — ובצד השרת יושבים MCP servers מעל שירותי האנליטיקס: Redshift, Athena, Glue, EMR. הארכיטקטורה תומכת בחיבורים מקומיים ומרוחקים. Mor הדגימה ארבע בקשות בשפה חופשית: "list down the Redshift clusters I have", "analyze my EMR environment for recent failures", "create a glue caller for my S3 bucket that runs weekly", ו-"analyze the sales data and show me the top 10 products" [7:35]. היא סיכמה: "It's that simple, והוסיפה שיש תמיכת MCP לכל שירותי האנליטיקס — מסטרימינג ועד חיפוש וממשל דאטה" [9:09].

4. התרחיש: DataFlow Solutions

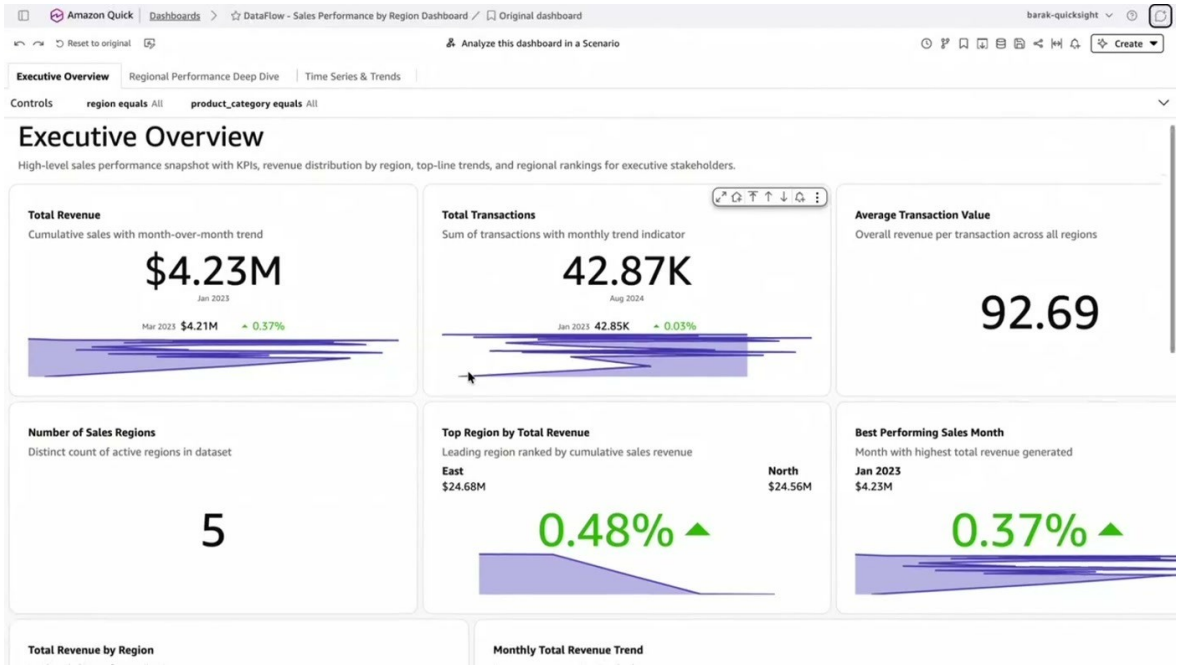
Barak פתח בסקר ידיים בקהל — מי מרגיש שהבוקר שלו עמוס בכיבוי שריפות, ומי חושב שאפשר לעשות שימוש טוב יותר בזמנו [9:09]. משם הוא עבר לחברה פיקטיבית, DataFlow Solutions, שנבנתה כולה לצורך ההדגמה.

הפרופיל: צוות דאטה בן שישה אנשים בהובלת סרה, אחראי על לפחות 40 data pipelines, עובד מול S3, Redshift, Glue ו-Athena, ומשתמש ב-QuickSight ל-GenBI. היום-יום שלהם — 9:00 עד 12:00 עבודה ריאקטיבית על מה שהצטבר לפני שעות העבודה, הפסקה, ואז תחזוקה של ה-pipelines הקיימים.

> **Barak Edward [10:40] —** זאת אומרת שחצי מהיום שלהם בכלל לא מוקדש לעבוד או תחזוקה וכמעט כלום ממנו לא משאיר זמן ליצירה של דברים חדשים. זה די קשוח, לא?

שני ה-pipelines

- **Pipeline ראשון — מ-raw ל-BI:** מיליארד רשומות ב-Parquet מגיעות ל-S3, נטענות ישירות לטבלה sales_transactions ב-Redshift, ומשם ל-dataset ב-QuickSight להצגה בדשבורד ולשאלות בצ'אט, ב-Direct Query mode [12:13].
- **Pipeline שני — מסלול ניהולי:** טבלאות Iceberg שמנוהלות ב-Glue Data Catalog וגם כ-S3 Tables, ניתנות ל-query דרך Athena, ומקוטלגות ומנוהלות ב-SageMaker Unified Studio [12:13].

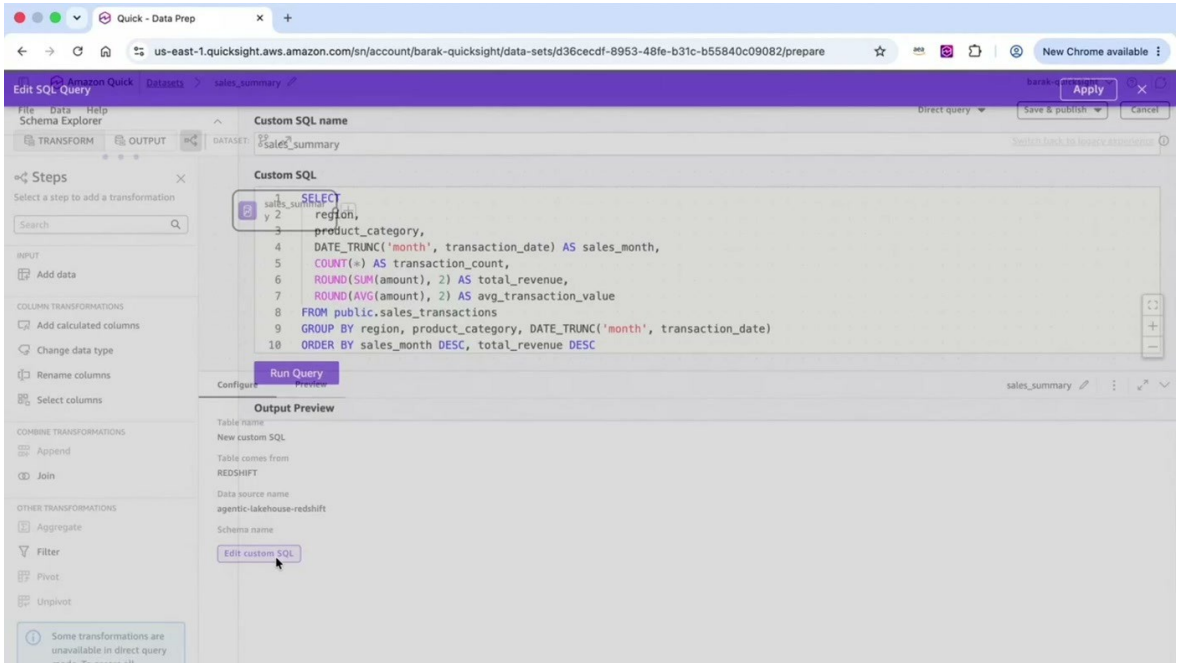


הדשבורד של DataFlow ב-QuickSight — סיכום מנהלים מעל מיליארד רשומות בטבלה ב-Redshift

הבעיה שהתגלתה

Barak שאל את הצ'אט שאלה שהתשובה עליה לא מופיעה בדשבורד — מהי הטרנזקציה הממוצעת לפי region ו-product category — ולא קיבל תשובה מספקת. הוא הדגיש שזו לא מגבלה של היכולות הסמנטיות ב-QuickSight: "בהכרח בגלל שבדייטסט המקור של הדשבורד הזה צורת השליפה היא לא אידיאלית לצורך הנוכחי שאנחנו מנסים לענות עליו כרגע" [13:46].

צלילה פנימה חשפה את הסיבה: ה-dataset פונה לטבלה ב-Redshift דרך Custom SQL ב-Direct Query, ושלוש עמודות נוצרות בזמן שליפה. מעל מיליארד רשומות, זה מקור לבעיות.



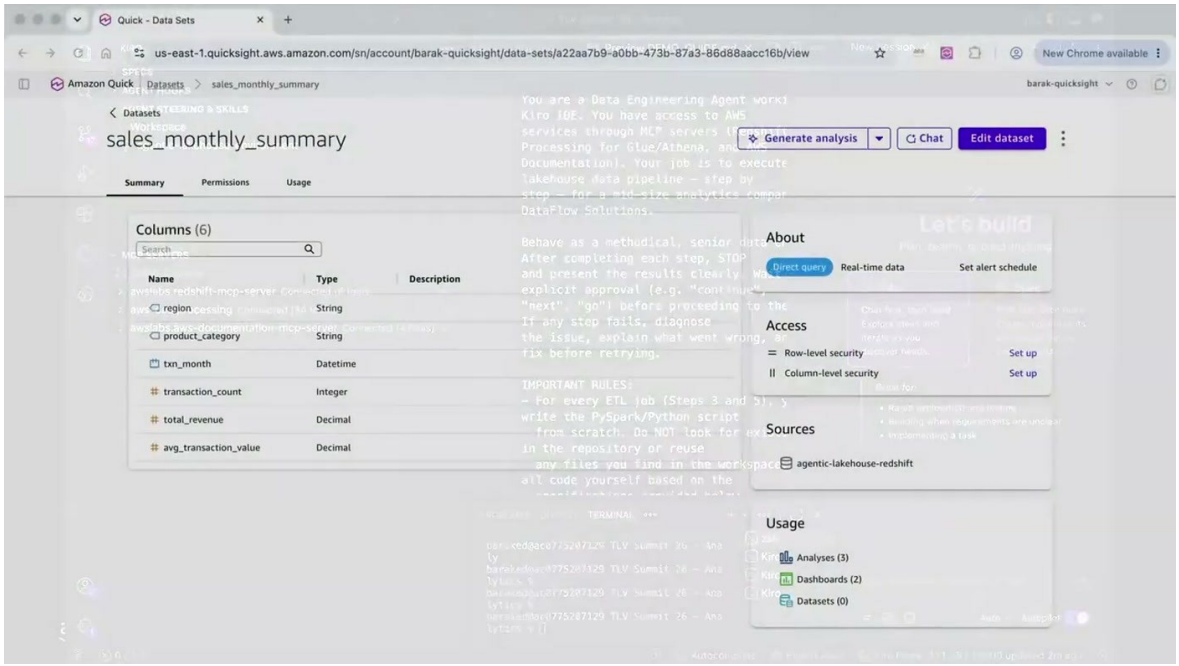
ה-Custom SQL שמאחורי ה-dataset: GROUP BY על public.sales_transactions עם transaction_count, total_revenue ו-avg_transaction_value מחושבים בזמן ריצה

ההחלטה של סרה: להפסיק לשלוף ישירות מטבלה בת מיליארד רשומות, ליצור במקומה materialized view ב-Redshift, ולהפנות אליה את ה-dataset החדש [15:19]. השאלה שהסשן כולו נבנה סביבה היא איך היא ביצעה את זה.

5. הסביבה האג'נטית: Kiro ושלושה MCP servers

הפלטפורמה שנבחרה היא Kiro, ה-IDE של AWS, שלפי Barak הוכרז GA בנובמבר האחרון ב-re:Invent [15:19]. סרה הגדירה בו שלושה MCP servers, ורק שלושה:

MCP Server	מה הוא עושה בדמו
Redshift	פעולות קריאה, מפשוטות ועד מורכבות, מול ה-Redshift cluster
Data Processing	קריאת raw data ו-Parquet, כלומר Iceberg, באמצעות Athena; וגם יצירה והרצה של Glue jobs
AWS Documentation	לוודא שהסוכן עובד לפי הצורה המתועדת והמומלצת



המעבר בין ה-dataset ל-Kiro: משמאל שש העמודות של sales_monthly_summary, ומעליהן ה-prompt וה-MCP servers המחוברים (redshift-mcp-server, data-processing, aws-documentation-mcp-server)

ה-prompt שהוגדר לסוכן לא היה שורה אחת. הוא מתאר את כל הפעולות בפירוט, מגדיר לסוכן פרסונה של Data Engineer בצוות של סרה, וקובע כלל עבודה מפורש: "יעצור אחרי כל פעולה, וייתן לה ה-human אישור להמשיך" [16:52]. על הסלייד נראה גם כלל נוסף — שעבור כל ETL job הסוכן יכתוב את סקריפט ה-PySpark מאפס ולא יחפש קבצים קיימים ב-workspace.

שימו לב שלושת ה-MCP servers האלה הם אותם שלושה שהסוכן האוטומטי בפרק 7 מתחבר אליהם. זו הנקודה הארכיטקטונית של הסשן: אותה שכבת כלים משרתת גם עבודה אינטראקטיבית וגם תגובה אוטומטית לאירועים.

6. הדמו: שבעה צעדים בשפה חופשית

התהליך שסרה הגדירה מראש, כפי שהוצג בדיאגרמה ואז בוצע בפועל [15:19, 16:52]:

#	הצעד	השירות	משך
1	קריאת הטבלאות הרלוונטיות ב-Redshift והבנת הסכמה	Redshift MCP	שניות
2	קריאת ה-source data — טבלת Iceberg — דרך Athena	Data Processing MCP	שניות
3	Glue job ראשון: בדיקות data quality, אגרגציה לטבלה מרכזית, כתיבה ל-S3 Tables	Glue	119 שניות
4	צפייה במקטע נתונים מהטבלה שנוצרה ב-S3 Tables ואישור	Data Processing MCP	עשרות שניות
5	Glue job שני: העתקת הנתונים מ-S3 Tables ל-materialized view ב-Redshift	Glue + Redshift	כ-40 שניות
6	צפייה בנתונים ובסוג הטבלה ב-Redshift לאימות	Redshift MCP	עשרות שניות
7	הסבר איך לטעון את הנתונים ל-dataset חדש או לתקן קיים	QuickSight	—

בכל צעד הסוכן עצר, הציג את התוצאה, וסרה אישרה. בצעד השלישי הוא לא רק כתב את הקוד אלא גם ניהל את ההרצה: "אז הסוכן כותב את הקוד, מעלה את הג'וב ובצורה איתרטיבית דואג שהוא ירוץ... והוא מבדק כמובן שהוא לא נכשל ואם הוא נכשל הוא יוצר קוד חדש" [18:23].

> **Barak Edward [18:23] —** אחרי 119 שניות, ליתר דיוק, הג'וב מסתיים בהצלחה.

התוצאה

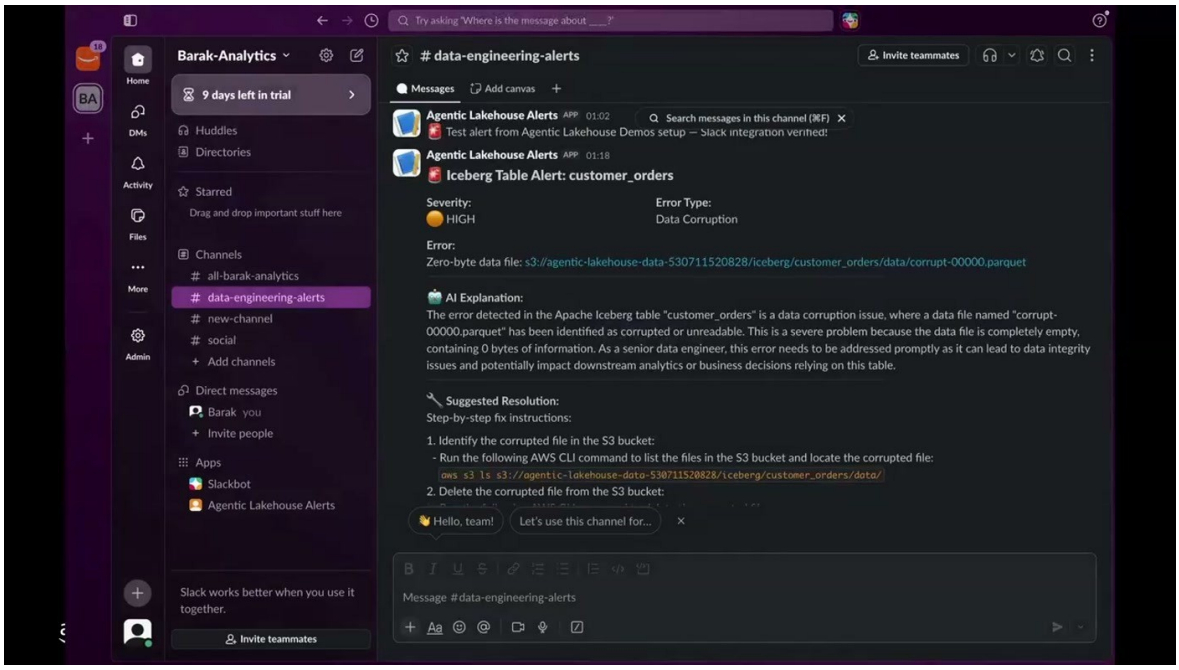
ה-dataset החדש מפנה לטבלה החדשה ב-Redshift, ושלוש העמודות מופיעות בו מראש — "כאשר אנחנו בדייר‍ק קווירי ולא בקסטום סיקוול" [19:54]. חזרה לדשבורד, אותה שאלה שקודם לא קיבלה מענה מחזירה עכשיו תשובה טקסטואלית תוך עשרות שניות, ובבקשה נוספת גם ויזואליזציה.

> **Barak Edward [21:26] —** אז בעצם סרה בעזרת שינוי של הגישה לדאטה, והדאטה עצמו, הכל באמצעות סוכנים, בבלי שינוי של השכפה הסמנטית בקויק, קיבלה תשובה לשאלה שמקודם לא הייתה לה תשובה.

7. מ-Kiro לפרודקשן: סוכן שמגיב לבד

החלק השני של הדמו ענה על השאלה המתבקשת: מה קורה כשאף אחד לא יושב מול המקלדת. הרקע — שישה מהנדסים מול 40 pipelines.

> **Barak Edward [21:26]** — שעה עכשיו שלוש בבוקר. סכמה של טבלת אייסברג נשפרה, והפעם הראשונה שאולי מישהו ישים לב לזה תהיה בשמונה בבוקר. יצליחו לתקן את זה אולי עד שתים עשרה? לא חבל על כל הזמן הזה?



ערוץ ה-Slack data-engineering-alerts: התראה על טבלת Iceberg, סיווג חומרה וסוג שגיאה, הסבר מה קרה, והוראות תיקון צעד-אחר-צעד עם פקודות ה-CLI

ההתראה בערוץ אינה רק "משהו נשבר". היא מזהה את הקובץ הפגום ב-S3, מסבירה שמדובר ב-data corruption בטבלת Iceberg בשם customer_orders, ומפרטת את שלבי התיקון. Barak הוסיף שמכאן הדרך קצרה: אפשר להעתיק את ההודעה המלאה כ-prompt ל-Kiro ולבקש תיקון [21:26].

הארכיטקטורה

"אתם בטח שואלים את עצמכם כמה פשוט או לא פשוט לממש את מה שהראיתי עכשיו, אז כזה פשוט" [23:01]. השרשרת שהוצגה: שגיאה בשירות כמו Glue או S3 → אירוע שנתפס ב-Lambda → EventBridge → טעינת הסוכן ב-AgentCore, עם הפניה למטריקות וללוגים ב-CloudWatch לקבלת התמונה המלאה. הסוכן עצמו מונע ב-LLM חזק — Sonnet או Opus — ומחובר בדיוק לאותם שלושה MCP servers מהדמו הקודם. הפלט הולך לתיעוד ב-DynamoDB ולהודעה בערוץ ה-Slack [23:01].

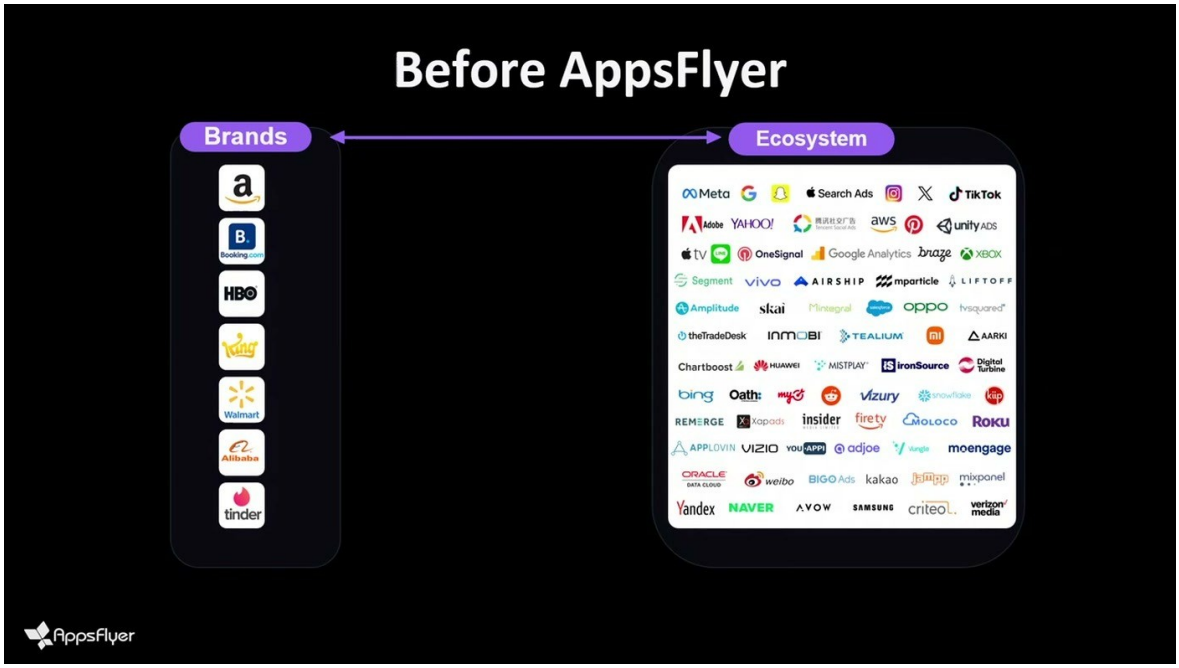
שלושה כללים לפרודקשן

- **Least privilege access:** "אין שום סיבה שהסוכן שלכם, שמדמד Data Engineer, יקבל יותר הרשאות מה Data Engineer, נכון? אם כבר, פחות" [24:32].
- **מודל מתאים ל-use case:** לא לשרוף כסף ומשאבים על מודלים יקרים במקומות שלא צריך אותם [24:32].
- **לא כל תהליך צריך LLM:** "לפעמים גם הגישה הטובה והישנה של אוטומציה תיתן את האפקט הטוב יותר והרחב יותר" [24:32]. הדוגמה שניתנה — בדיקות data quality בסיסיות כמו מייל או תעודת זהות, שבהן LLM רק יוריד את הדיוק [26:02].

> **Barak Edward [24:32]** — Agentic AI לא מחליף את מהנדס הדאטה. הוא משחרר אותו להתמקד במה שהארגון הגדיר לו כחשוב.

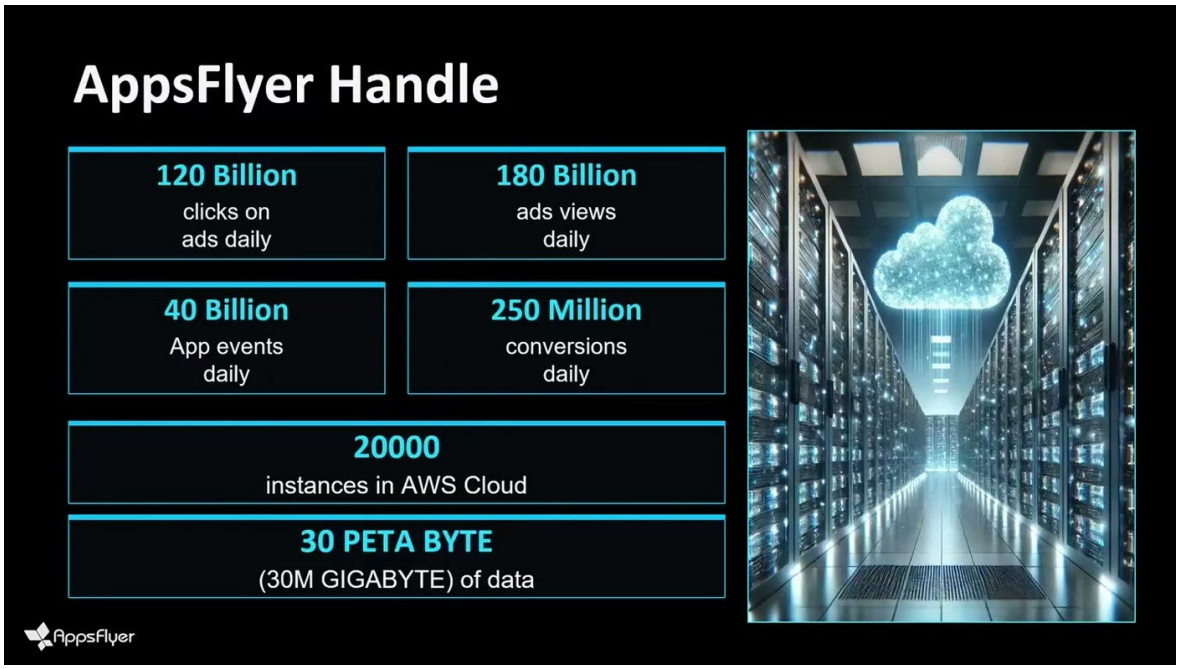
8. AppsFlyer: הסקייל והבעיה

Ruli Weisbach, EVP R&D ב-AppsFlyer, הסביר תחילה מה החברה עושה: מדידת ביצועי שיווק עבור מותגים עם אפליקציות מובייל. הפרסמים — TikTok, Snapchat, YouTube, Google, Facebook — שולחים אירוע בכל חשיפה או קליק; ה-SDK של AppsFlyer, שמותקן כמעט בכל אפליקציה, שולח אירוע על כל התקנה ופעולה; והפלטפורמה מבצעת את ההתאמה בין השניים [27:34].



התמונה שלפני AppsFlyer: מותגים מצד אחד, אקוסיסטם פרסום מפוצל מהצד השני, ובלי דרך לקשר ביניהם

Ruli Weisbach [29:06] — אנחנו באפסלו מתמודדים עם משהו סדר גודל של בין 300 ל-400 ביליון איבנטים ביום שזורמים לתוך הפלטפורמה שלנו.



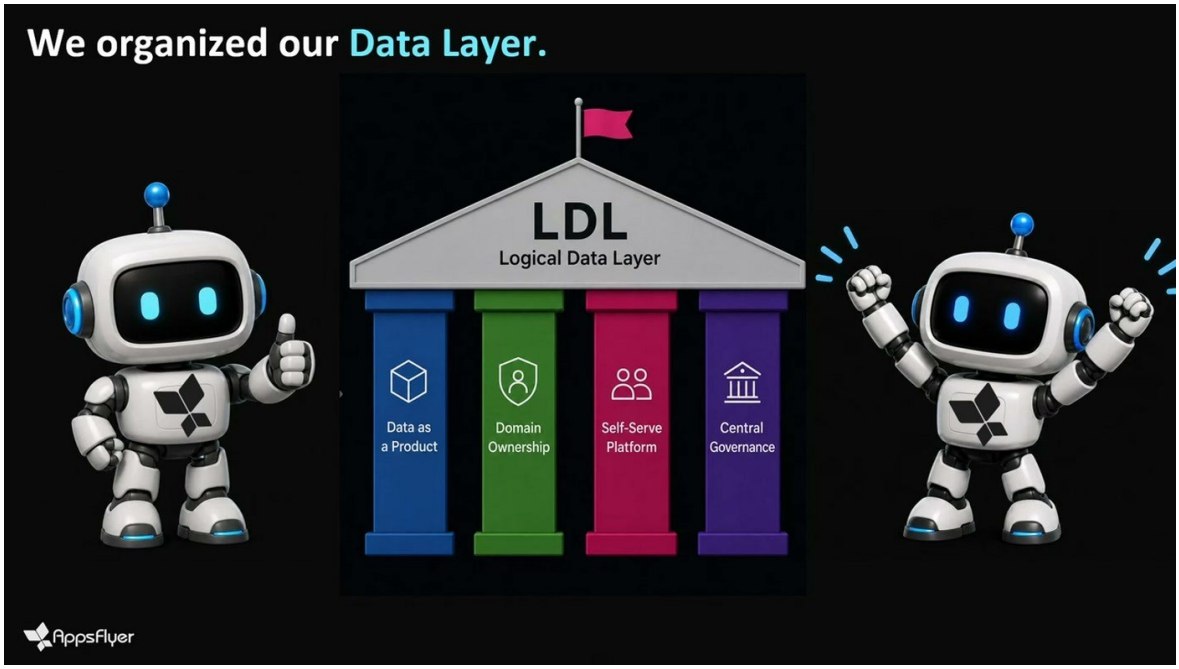
המספרים כפי שהוצגו על הסלייד: 120 מיליארד קליקים ו-180 מיליארד חשיפות ביום, 40 מיליארד אירועי אפליקציה, 250 מיליון המרות, 20,000 instances ו-30 פטה-בייט

הארכיטקטורה שהוצגה: האירועים נכנסים ל-AWS באירלנד דרך AWS WAF, זורמים לשכבות Kafka, ומשם ל-processing על Aerospike, DynamoDB ו-EMR לצורך ה-matching. התוצאות נכתבות ל-[29:06] S3. הלוגיקה, לדבריו, אינה מורכבת — הוולידום והסקייל הם המורכבות, בתוספת הדרישה לחישוב ב-near real time.

מ-S3 ל-Logical Data Layer

לפני שנתיים-שלוש הבינו ב-AppsFlyer שכדי לתת ללקוחות יכולות agent ו-MCP, לא מספיק שהדאטה יהיה ב-S3.

> **Ruli Weisbach [30:39] —** כי אין מה לעשות כדי שאייג'נטים וכדי ש-LLM אם יבינו את הדאטה שלנו צריך להיות מורגן בצורה הרבה יותר טובה.

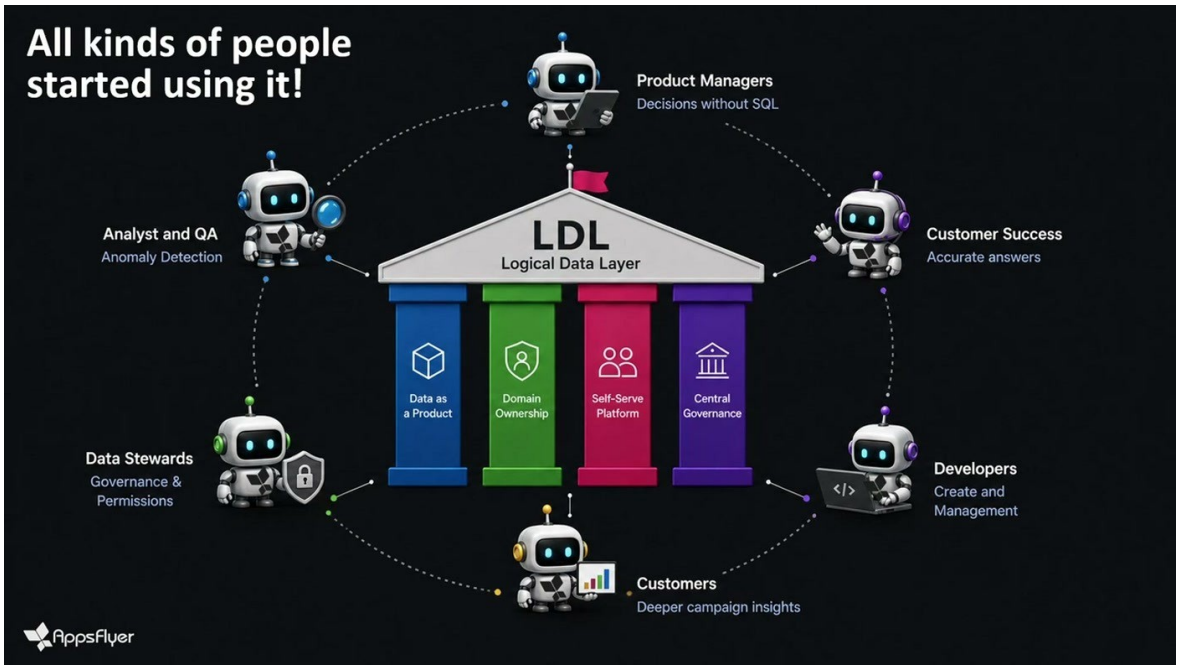


ארבעת העמודים של ה-LDL: Data as a Product, Domain Ownership, Self-Serve Platform ו-Central Governance

Alicia Plotnikov, שמובילה את צוות AIDevEx, פירטה את ארבעת העקרונות: centralized — כל הדאטה במערכת אחת; organized — חינוך ה-data owners לתעד, לפרט ולתייג לפני הכניסה למערכת; self-serve — כל צוות כותב וקורא בלי לחכות בתור; ו-[32:12] AI ready.

אבל הצלחת הפרויקט היא שיצרה את הבעיה הבאה. Ruli תיאר שתי מכשלות: ה-diversity של הפרטנרים, שכל אחד שולח דאטה בפורמט ובמבנה אחר — "יש לנו איבנטים שלפעמים בסכמה יש להם מאות קולומים כדי לתאר את כל האפשרויות" [30:39] — והסקייל של הצרכנים.

> **Ruli Weisbach [30:39] —** סקל גדול, דיברסיטי גדול, מייצר קומפלקסיטי מאוד מאוד גדול, והגענו למצב שגם הלקוחות, גם אנשי הפרודרק, גם האנג'יניר התחילו ללכת לאיבוד בדבר החדש הזה שיצרנו שנקרא logical data layer.



מי התחיל לצרוך את ה-LDL: מפתחים, אנליסטים ו-data stewards, customer success, QA, PMs — וגם סוכנים לקוחות

"כל אחד בנה שכבה חכמה משל עצמו שתעזור לו לקרוא את המידע הזה בצורה קלה ומהירה. פתאום הסדר שעשינו נהיה לאי סדר" [32:12]. ארבע התוצאות שהיא מנתה: כל אחד אסף context משלו ובנה source of truth פרטי; תקשורת בין סוכנים נעשתה מסובכת כי לא כולם דיברו באותה שפה; שינויים בפרודקשן האטו, כי כל שינוי חייב עדכון ואישור של כל הסוכנים שתלויים בו; ו-data validation הפך מסורבל, כי לא היה ברור מול איזו אמת לאמת [33:43].

9. סוכן המטא-דאטה של AppsFlyer

המסקנה הייתה שצריך source of truth אחד. התובנה שהובילה לפתרון: כבר מתחילת פרויקט ה-LDL חינכו את ה-data owners לפרט את המידע — table description, column description, KPI, lineage, תיעוד. כלומר, המטא-דאטה כבר היה שם.

> **Alicia Plotnikov [33:43] —** ולמה לא לעשות את זה עוד יותר מדליק ועוד יותר מסובך? בואו נזרוק to odd agent the mix, single metadata agent, שהם יכולים לדבר איתו ולהשיג את המטה-Data הזה.

מה נבנה בפועל

- **איסוף לצומת אחד:** כל מקורות המטא-דאטה — repos, טבלאות, תיעוד, lineage — נאספו ל-DataHub. Alicia הדגישה שזו לא דרישה: "אנחנו השתמשנו בדאטה אב, אבל אתם יכולים להשתמש בכל סלושן אפשרי, אפילו טבלה באיזשהו דאטה בייס" [35:16].
- **כלי ראשון — API: API** על גבי המטא-דאטה, לשאלות ישירות: מה זו הטבלה הזאת, מה זו העמודה הזאת, מהו ה-lineage [35:16].
- **כלי שני — RAG:** המטא-דאטה נדחף דרך Lambda ל-S3, עליו הופעלה chunking strategy ונבנה RAG באמצעות AWS Knowledge Base. הכלי הזה נועד לשאלות הוליסטיות: "האם יש קולומים עם IPv6 בהם? או האם יש קולומים עם משמעות הבאה?" [35:16].
- **הסוכן:** "אג'ן שרץ ב-AWS בדרוק Agent Core Runtime ומשתמש במודל של קלוד" [35:16] — לשאלות ישירות פונה ל-API, לשאלות הוליסטיות ל-RAG, ואם חסר לו context הוא קופץ ביניהם.
- **שתי כניסות:** MCP ו-A2A (Agent to Agent), "כך שגם בני אדם וגם אג'נטים יוכלו להשיג את המתדדתה שלהם בקלות" [35:16]. בנוסף ה-API וה-RAG נותרו פתוחים ישירות למי שרוצה לבצע את הלוגיקה בצד שלו.

ארבע הבעיות מפרק 8 נסגרו אחת-אחת: source of truth אחד במקום אוסף פרטי; תקשורת agent-to-agent פשוטה יותר כי כולם מדברים באותו vocabulary; שינויים בפרודקשן קלים יותר כי יש קומפוננטה אחת לעדכן — "הוא יודע מה קורה בגרסה הישנה ובגרסה החדשה, וכל מה שהאג'נטים והישויות צריכים לעשות, זה להחליט מתי לעבור" [36:49]; ו-data validation פשוט יותר כי כולם נשענים על אותו מטא-דאטה.

משתמשים שבועיים פעילים ב-Slack bot של ה-LDL, עם סימון נקודת השחרור של סוכן המטא-דאטה

הגרף מראה weekly active users ב-Slack bot שנבנה ל-onboarding של פרויקט ה-LDL, עם החץ שמסמן מתי נכנס סוכן המטא-דאטה בערוץ ה-A2A. "כפול 2 active weekly users בשלוש שבועות הראשונים" [36:49] — ולפי הסלייד, 80% user adoption. Alicia הוסיפה שזה עוד לפני ערוץ ה-MCP, שבו "הראיינו אפילו, כפול 3".

Alicia Plotnikov [36:49] — מדיבליפרים, אנליסטים, PMים, אין צורך יותר לחפש בדוקומטציה, למצוא אנשי מבטח ולשאול אותם שאלות, לגשש בחושך או לנחש. כל התשובות במטה דאטה אג'נט אחד.

המסקנה של AppsFlyer "יש חשיבות מאוד גדולה בלסדר את הדאטה שלכם בשביל להביא לביצויים אמיתיים, אבל לא פחות, יותר חשוב זה לקחת את הצ'אנס הזה לסדר את המטה דאטה לר שלכם" [36:49]. זו הנקודה שמבדילה את מקרה הבוחן הזה מהדמו: ב-AppsFlyer ההשקעה הגדולה לא הייתה בסוכן, אלא בשכבת המטא-דאטה שמתחתיו.

10. מה לוקחים מכאן

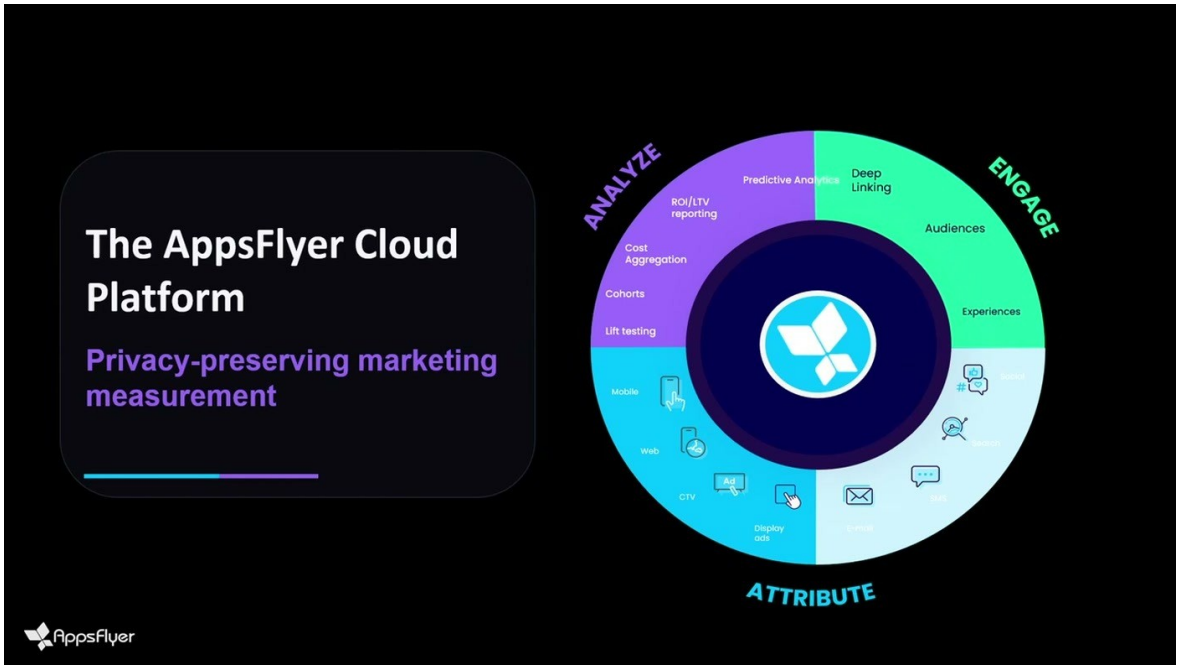
- **MCP הוא נקודת הכניסה הזולה.** הדמו כולו — קריאה מ-Redshift ומ-Athena, יצירה והרצה של Glue jobs, כתיבה ל-S3 Tables — רץ על שלושה MCP servers מוכנים, בלי קוד אינטגרציה. זו נקודת התחלה שאפשר לנסות בשבוע.
- **Human in the loop הוא לא פשרה, הוא התכנון.** ה-prompt של סרה הורה לסוכן לעצור אחרי כל צעד ולחכות לאישור. הדמו רץ שבעה צעדים בדיוק כך, וזה מה שאיפשר לראות טעות לפני שהיא מגיעה לטבלה.
- **אותה שכבת כלים משרתת אינטראקטיבי ואוטומטי.** המעבר מ-Kiro ל-← Lambda → EventBridge AgentCore לא דרש כלים חדשים, רק trigger ו-runtime אחרים.
- **איכות הפלט נגזרת מאיכות המטא-דאטה.** AppsFlyer הגיעו למסקנה הזאת אחרי שבנו LDL מלא וגילו שהוא לא מספיק. סוכן מעל דאטה לא מתועד יחזיר ניחושים.
- **לא כל בעיה צריכה LLM.** הכלל הזה נאמר בסשן שכולו על סוכנים, ודווקא משם הוא נשמע אמין. בדיקות פורמט בסיסיות עדיפות באוטומציה קלאסית.
- **המדידה היחידה שהוצגה היא מדידת אימוץ.** פי 2 משתמשים ב-3 שבועות ו-80% אימוץ — לא חיסכון בשעות ולא שיפור ב-latency. מי שמתכנן pilot כדאי שיגדיר מראש איזו מטריקה הוא מודד.

נקודות להמשך

נושא	למה זה חשוב	חותמת זמן
הרשאות הסוכן בפועל	least privilege הוזכר ככלל, אבל לא הודגם איך מממשים אותו מול MCP servers	24:32
עלות ההרצה	Sonnet או Opus לכל אירוע — לא הוצגה הערכת עלות	23:01
אמינות של קוד שנכתב בזמן ריצה	הסוכן כותב PySpark מאפס בכל הרצה; לא נדונו גרסאות, בדיקות או review	16:52
מה קורה כשהסוכן טועה	הודגמה רק לולאת תיקון של job שנכשל, לא של job שהצליח והחזיר נתון שגוי	18:23
A2A	הוצג כערוץ שני של סוכן המטא-דאטה, בלי פירוט טכני	35:16

11. מונחון

מונח	משמעות
MCP	Model Context Protocol — פרוטוקול פתוח לחיבור סוכנים לכלים ולשירותים. הוצג כ"פורט USB-C" של עולם ה-agents
Agentic loop	מחזור חוזר של reasoning, בחירת כלי, הרצה ועיבוד התוצאה, עד לתשובה סופית
Kiro	ה-IDE של AWS לפיתוח מבוסס סוכנים; לפי הדובר הוכרז GA בנובמבר ב-re:Invent
AgentCore Runtime	סביבת הרצה מנוהלת לסוכנים ב-Amazon Bedrock; בשן שימשה גם את הסוכן האוטומטי של AWS וגם את סוכן המטא-דאטה של AppsFlyer
Apache Iceberg	פורמט טבלאות פתוח ל-data lake, עם ניהול סכמה וגרסאות
S3 Tables	טבלאות Iceberg מנוהלות על S3
Materialized View	תוצאת שאילתה שנשמרת כטבלה, במקום חישוב מחדש בכל שליפה — הפתרון המרכזי בדמו
Direct Query	מצב ב-QuickSight שבו השאילתה רצה מול המקור בזמן אמת, בלי טעינה ל-SPICE
LDL	Logical Data Layer — שכבת הדאטה המאוחדת של AppsFlyer, על ארבעה עמודים
A2A	Agent to Agent — ערוץ תקשורת בין סוכנים, במקביל ל-MCP
RAG	שליפה סמנטית ממאגר מסמכים כהקשר למודל; ב-AppsFlyer נבנה מעל AWS Knowledge Base



הפלטפורמה של AppsFlyer כפי שהוצגה בסיום החלק שלהם — מדידה שיווקית משמרת פרטיות