DATA+AI SUMMIT 2022

Migrating Complex SAS Processes to Databricks

- Case Study

Jesse Beaumont
Chief Technology Officer
TENSILE AI

Uday Kumar
Chief Digital Officer
akira

ORGANIZED BY databricks

Agenda

- Introductions
- Featured Case Study
- Challenges and Opportunities
- Existing Solution
- Modernizing with Databricks
- Benefits
- Conclusion

Introductions

Jesse Beaumont

TENSILE AI

Data & Analytics Engineering

Healthcare ML/AI Solutions

SAS Migration

Uday Kumar

akira

Digital Transformation

Cloud Modernization

Big Data & Analytics

Featured Case Study

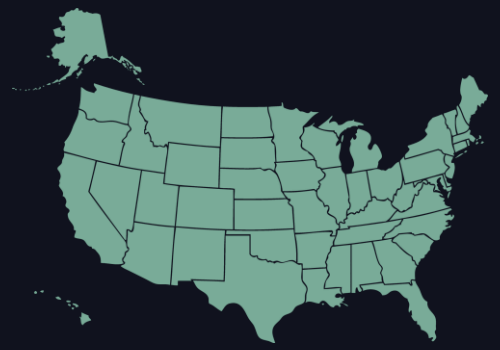
Center for Medicaid and CHIP Services

CHIP = Children's Health Insurance Program

CMCS is a division within CMS, that serves as the focal point for all the national program policies and operations for Medicaid, CHIP, Basic Health Program, comprised of **74 M** beneficiaries across 54 states.

CMCS Data Quality Measures

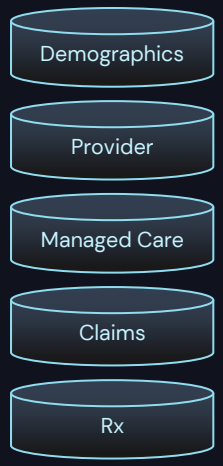
Process to assess the integrity of State Submitted healthcare data used to improve quality of care, assess beneficiary access to care, improve program integrity, and provide support to states and territories.



54 States / Territories

Every Month

Healthcare Data



3,000 + Distinct Measures

- Process of Data Quality (DQ) checks
- Rules for checking data integrity
- States wait weeks to view status report
- Corrections require full re-submission
- ~ 1500 of 3000+ *active* measures

Featured Case Study

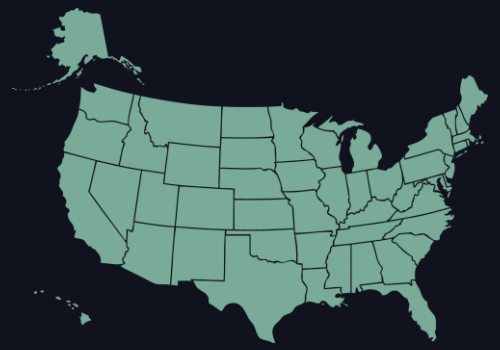
CMCS Data Quality Measures

Code is comprised of

- SAS Macro / SAS DATA step / SAS Procedures
- Generations of Code History
- Multiple Contributors

Data includes

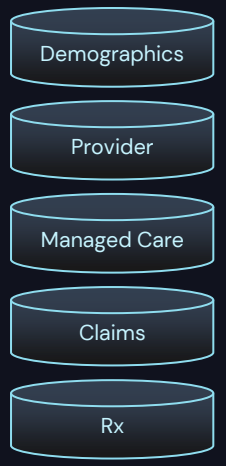
- Beneficiary Eligibility
- Provider Details
- Service Utilization
- Claims
- Third Party Liabilities
- Expenditures



54 States / Territories

Every Month

Healthcare Data



3,000 + Distinct Measures

Featured Case Study

Problems

Technical

- Difficult to analyze runtime performance metrics
- Artificial restart points
- No graceful exception handling
- No visibility in progress or data readiness
- Shared computing resources amongst batch sessions
- Incapability for parallel processing

Business Operations

- Multiple Contributors
- Long batch periods to process each state data submission
- Difficult to add or change measures
- Not benefitting from available Cloud Architecture
- No longer able to efficiently meet demands

Cultural

- Data assets are produced by siloed business units
- Misconceptions of available capabilities
- Resistance to Change

Featured Case Study

CMCS Initiatives

Business Operations

Leverage capabilities inherent in Databricks into enable multiple data analytics teams to evaluate a wide range of Medicaid business questions

Technical

Migrate current SAS processes to Databricks and optimize the new processes to run in a cloud environment and open-source/open-standard technologies

Cultural

Provide a primer for optimizing legacy solutions to Databricks using modern architecture patterns

Databricks was an available technology option

...yet not proven or widely adopted

Challenges

SAS Macro-Driven Batch Scripts

Eligibility	Expenditures	Providers	Third Party Liability
Utilization	Managed Care	Fee for Service	Missing-ness

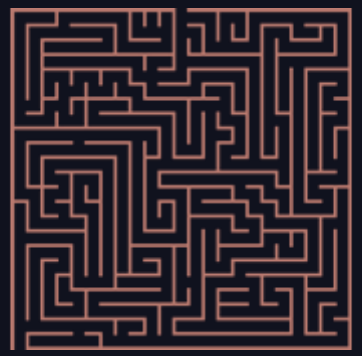
- Unclear Data/Code Dependencies
- Over-expanded code for Managing Resources
- Metadata Blended in with Processing Code
- Data Cleansing Laced Throughout
- Heavy Reliance on Passthrough

Challenges

Code Conversion Automation and Obstacles

The Automation Option

- Not starting from scratch
- Base Elements
  - SQL
  - File I/O
  - procedural code
  - . . .
- Time
- Reduced Human Error



Inefficiencies resident in original code
. . . will be repeated
. . . in a newer technology

Output Delivery

- Web Report Studio
- Information Delivery Portal
- BI Dashboard
- ODS
- IntrNet (*_webout*)
- proc sgplot, gchart
- . . .

Nuances

- Formats vs ANSI Types (*best.*)
- Clever coding methods
  - by grouping with blinded sorting
  - heavy macro recursion
  - advanced hash object functionality
- Allowable proc SQL syntax
- Tagsets / Templates
- Autocall Library
- Stored Secured Macros

Propagate Design Flaws / Defects

- Restart Points
- Linear Execution

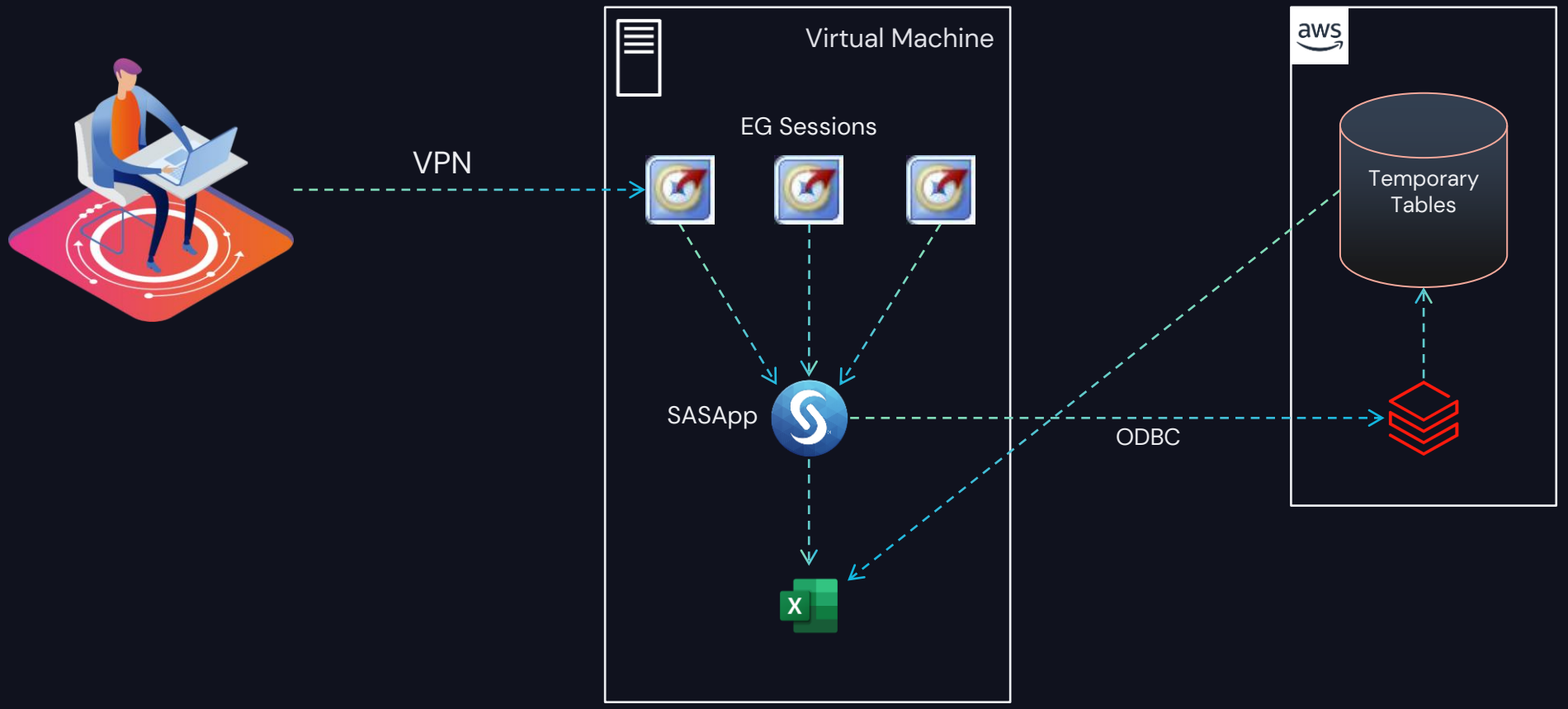
Performance and Validation

- Code is Not Optimized for the Platform
- Difficult to respond to system validation issues
- Code Quality – Who's is it to own?

Existing Solution

Existing Solution

Cumbersome End-User Experience



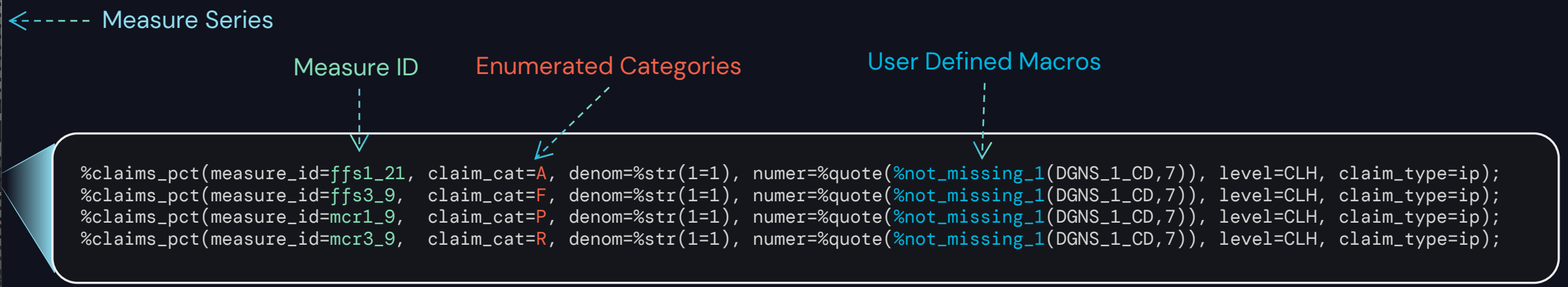
Existing Solution

Over-expanded code for Managing Resources

Aggregating Results

Existing Solution

Metadata Blended with Processing Code




```
%claims_pct(measure_id=ffs1_21, claim_cat=A, denom=%str(1=1), numer=%quote(%not_missing_1(DGNS_1_CD,7)), level=CLH, claim_type=ip);
%claims_pct(measure_id=ffs3_9,  claim_cat=F, denom=%str(1=1), numer=%quote(%not_missing_1(DGNS_1_CD,7)), level=CLH, claim_type=ip);
%claims_pct(measure_id=mcr1_9,  claim_cat=P, denom=%str(1=1), numer=%quote(%not_missing_1(DGNS_1_CD,7)), level=CLH, claim_type=ip);
%claims_pct(measure_id=mcr3_9,  claim_cat=R, denom=%str(1=1), numer=%quote(%not_missing_1(DGNS_1_CD,7)), level=CLH, claim_type=ip);
```

```
%macro claim_cat_sql();
   . . .
   ,case when (clm_type_cd in ('1','3', 'A','C') and adjstmt_ind in ('0','4') ) then 1 else 0 end as claim_cat_AA
   ,case when (clm_type_cd in ('1','3') and adjstmt_ind in ('0') ) then 1 else 0 end as claim_cat_ AB
   ,case when (clm_type_cd in ('A','C') and adjstmt_ind in ('0') ) then 1 else 0 end as claim_cat_ AC
   ,case when (clm_type_cd in ('1') and xovr_ind = '1' ) then 1 else 0 end as claim_cat_ AD
   ,case when (clm_type_cd in ('1','A') and adjstmt_ind in ('0') ) then 1 else 0 end as claim_cat_ AE
   ,case when (clm_type_cd in ('3','C') and adjstmt_ind in ('0') ) then 1 else 0 end as claim_cat_ AF
   ,case when (clm_type_cd in ('2','B') and adjstmt_ind in ('0','4') ) then 1 else 0 end as claim_cat_ AG
   ,case when (clm_type_cd in ('1','A') and adjstmt_ind in ('0','4') ) then 1 else 0 end as claim_cat_ AH
   . . .
%mend;
```

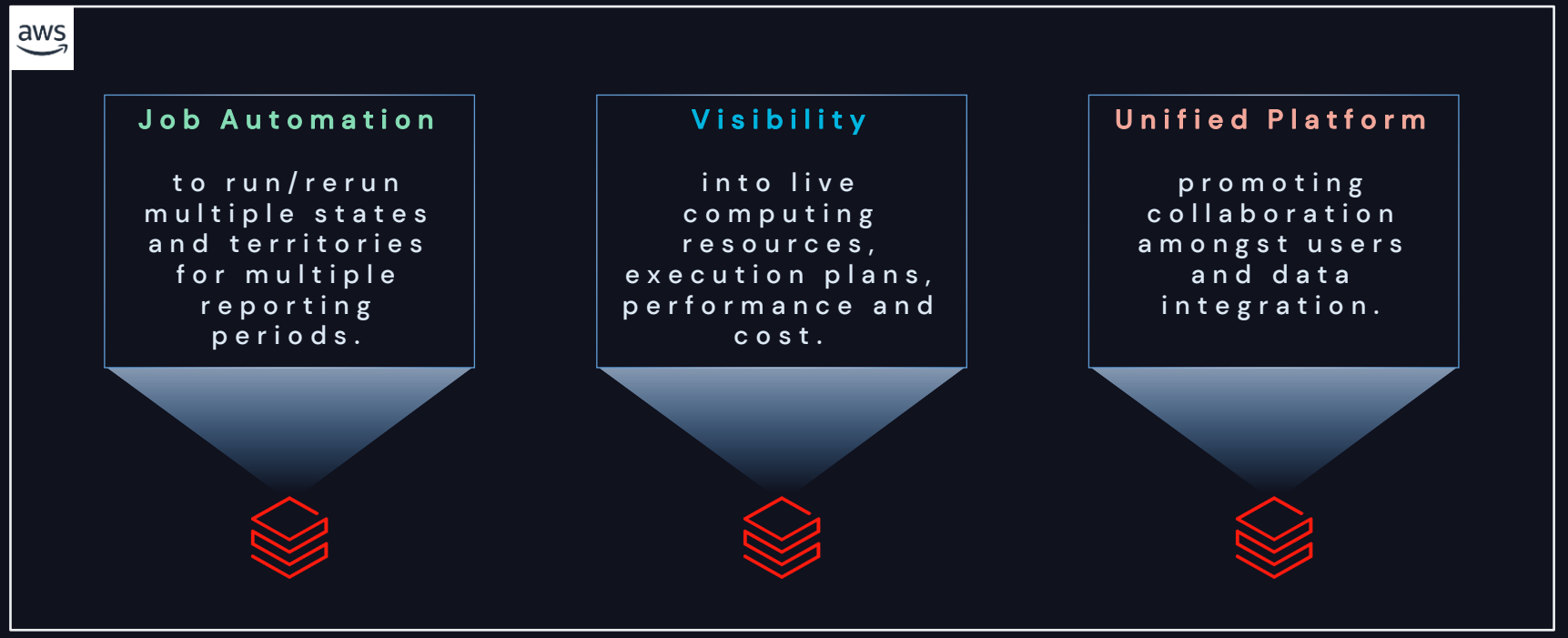
51 Sparsely Populated Columns !

Do not propagate problems into the new system.

Modernizing with Databricks

Modernizing with Databricks

Expanded Capabilities



Git Integration

Reduced Runtimes

Open Source Languages

Notifications

Redesigned Solution

Dislodge the Measures from the Processing

```
%claims_pct(measure_id=ffs1_21, claim_cat=A, denom=%str(1=1), numer=%quote(%not_missing_1(DGNS_1_CD,7)), level=CLH, claim_type=ip);
%claims_pct(measure_id=ffs3_9,  claim_cat=F, denom=%str(1=1), numer=%quote(%not_missing_1(DGNS_1_CD,7)), level=CLH, claim_type=ip);
%claims_pct(measure_id=mcr1_9,  claim_cat=P, denom=%str(1=1), numer=%quote(%not_missing_1(DGNS_1_CD,7)), level=CLH, claim_type=ip);
%claims_pct(measure_id=mcr3_9,  claim_cat=R, denom=%str(1=1), numer=%quote(%not_missing_1(DGNS_1_CD,7)), level=CLH, claim_type=ip);
```

Familiar Namespace to Users

```
run_901 = [
    .
    .
    ['901', 'claims_pct', 'ffs1_21', 'A', '1=1', "%not_missing_1(DGNS_1_CD, 7)", 'CLH', 'ip'],
    ['901', 'claims_pct', 'ffs3_9',  'F', '1=1', "%not_missing_1(DGNS_1_CD, 7)", 'CLH', 'ip'],
    ['901', 'claims_pct', 'mcr1_9',  'P', '1=1', "%not_missing_1(DGNS_1_CD, 7)", 'CLH', 'ip'],
    ['901', 'claims_pct', 'mcr3_9',  'R', '1=1', "%not_missing_1(DGNS_1_CD, 7)", 'CLH', 'ip'],
    .
    .
]
```

Modernizing with Databricks

Deploy an API Library to Databricks

Accept *macro-like* syntax

```
.
.
select
   sum(case when ({DQClosure.parse(x['numer'])}) then 1 else 0 end) as numer
   . . .
   from
       {DQMeasures.getBaseTable(dqm, x['level'], x['claim_type'])}
   where
      ({DQM_Metadata.claim_cat[x['claim_cat']]}) and {DQClosure.parse(x['denom'])})
.
.
```

Derive specific category at runtime.

Modernizing with Databricks

Deployed API Library to Databricks

Cmd 1

```
from dqm import DQMeasures
dqm = DQMeasures('202202', 'CA')
```

Deployed Library

Cmd 2

```
dqm.init()
```

```
2022-04-20 19:03:30,958 - dqm_log - INFO - Creating Base Views...
2022-04-20 19:03:30,958 - dqm_log - INFO - Creating Base Eligibility Views...
2022-04-20 19:04:02,833 - dqm_log - INFO -           base_elig_view
2022-04-20 19:04:03,300 - dqm_log - INFO -           base_dtrmnt_view
2022-04-20 19:04:03,300 - dqm_log - INFO - Creating Base Claim Line Views...
2022-04-20 19:04:03,757 - dqm_log - INFO -           prep_cll_ip_view
2022-04-20 19:04:04,187 - dqm_log - INFO -           prep_cll_lt_view
2022-04-20 19:04:04,670 - dqm_log - INFO -           prep_cll_ot_view
2022-04-20 19:04:05,535 - dqm_log - INFO -           prep_cll_rx_view
2022-04-20 19:04:05,535 - dqm_log - INFO - Creating Base Claim Header Views...
2022-04-20 19:04:05,980 - dqm_log - INFO -           prep_clh_ip_view
2022-04-20 19:04:06,439 - dqm_log - INFO -           prep_clh_lt_view
2022-04-20 19:04:06,965 - dqm_log - INFO -           prep_clh_ot_view
2022-04-20 19:04:07,559 - dqm_log - INFO -           prep_clh_rx_view
2022-04-20 19:04:07,559 - dqm_log - INFO - Creating Base Eligibility Info Views...
```

Views

Modernizing with Databricks

From the measure, not the process...

Cmd 1

```
dqm.run(spark, ['ffs1_21', 'ffs3_9', 'mcr1_9', 'mcr3_9'])
```

Measure Ids

Cmd 2

```
dqm.run(spark, dqm.where(measure_cat='FFS', claim_file='IP'))
```

Category / Service

Cmd 3

```
dqm.run(spark, dqm.where(series='901'))
```

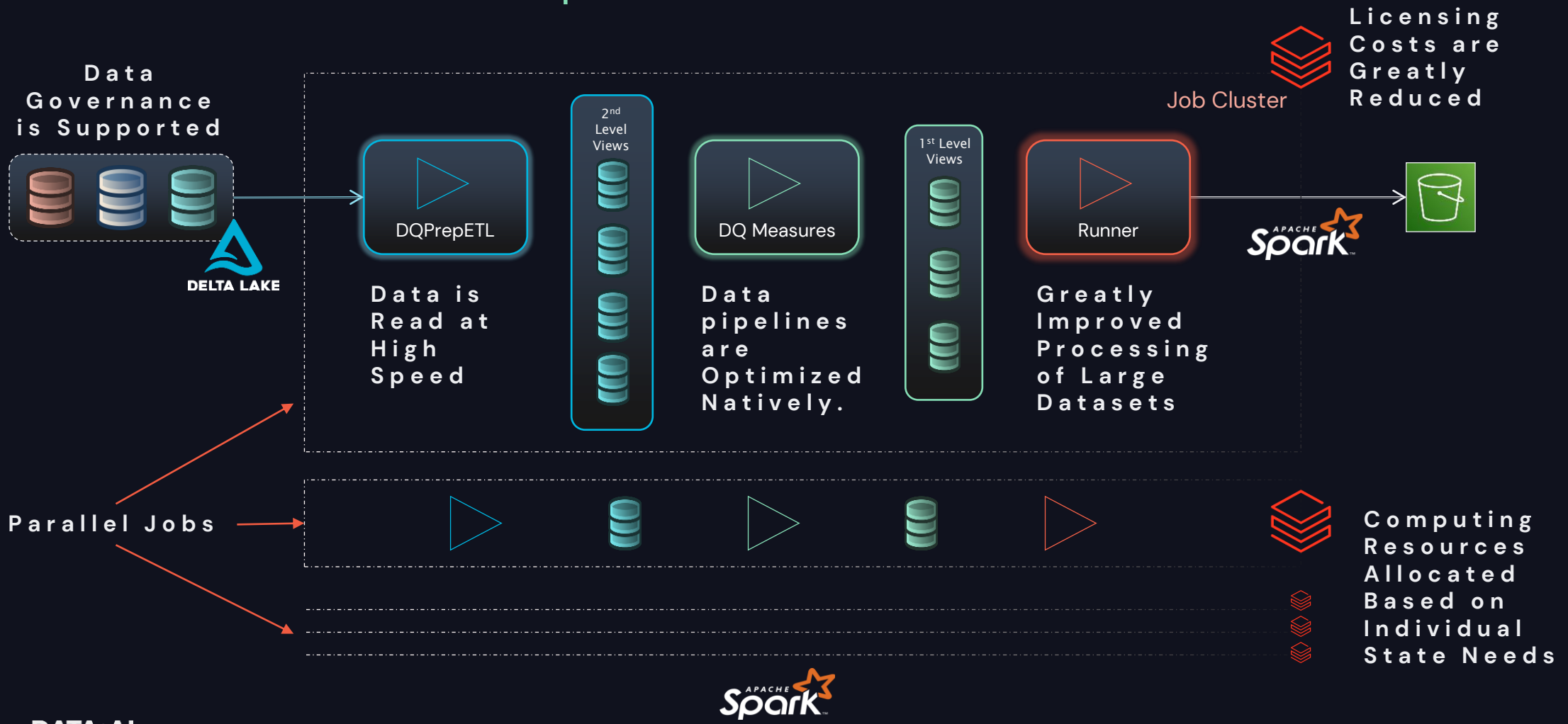
Series Module

```
2022-04-20 19:05:25,764 - dqm_log - INFO - Running 583 measure(s).
2022-04-20 19:05:42,941 - dqm_log - INFO - (1/583) 0% Measure: MCR13_17 (Series: 901)
2022-04-20 19:06:33,756 - dqm_log - INFO - (2/583) 0% Measure: MCR10_1  (Series: 901)
2022-04-20 19:06:58,163 - dqm_log - INFO - (3/583) 1% Measure: FFS26_5  (Series: 901)
2022-04-20 19:07:24,765 - dqm_log - INFO - (4/583) 1% Measure: MCR32_14 (Series: 901)
2022-04-20 19:07:32,068 - dqm_log - INFO - (5/583) 1% Measure: FFS9_100 (Series: 901)
2022-04-20 19:07:55,522 - dqm_log - INFO - (6/583) 1% Measure: MCR32_17 (Series: 901)
2022-04-20 19:08:01,660 - dqm_log - INFO - (7/583) 1% Measure: FFS5_5   (Series: 901)
2022-04-20 19:08:27,423 - dqm_log - INFO - (8/583) 1% Measure: MCR30_1  (Series: 901)
2022-04-20 19:09:35,859 - dqm_log - INFO - (9/583) 2% Measure: MCR17_4  (Series: 901)
```

Benefits

Benefits

Streamlined Data Pipeline



Benefits

Recorded Performance Gains

California

- 12M Medicaid/CHIP Enrollees
- 323M obs (8 files) *as of* May 2022
- Restate 3 months of data submissions
- Original End-to-End Processing Time

  ~ 7 hours
- Databricks

  ~ 67 mins

  (average runtime per module in parallel)

80%

Run-time Reduction

Multiple States may Run in Parallel

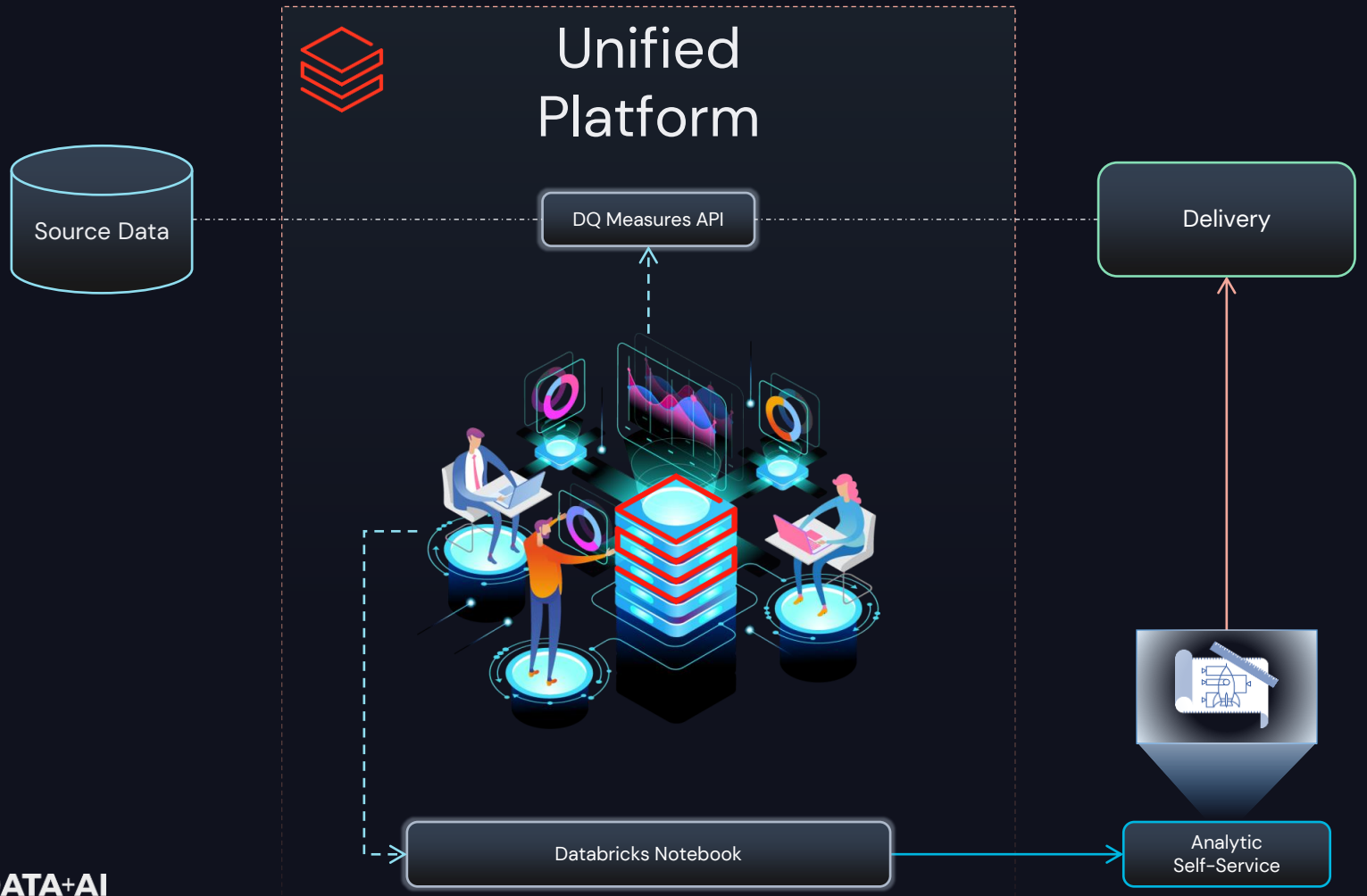
States
Measures
Reporting Periods

Monitored
Rebalanced
Calibrated

Conclusion

Client Success

Modernized on Databricks



Business Operations

- Development of New Measures by the data analytics teams
- Expedited Delivery to States
- Adaptive Data Governance

Technical

- Integrated Data Assets
- Reduced Redundancy
- Performance Visibility

Cultural

- Collaborative Development
- Community Support
- User Acceptance & Engagement

DATA+AI SUMMIT 2022

Thank you

Uday Kumar

Chief Digital Officer

akira

Jesse Beaumont

Chief Technology Officer

TENSILE AI