
AWS
re:Invent

GAM301

How Zynga modernized mobile analytics with Amazon Redshift RA3

Bharath Anandaram
Architect, Zynga

Gena Gizzi
Solutions Architect, AWS

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.



game tech

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS analytics ecosystem

Data, visualization, engagement, & machine learning


Data


Dashboards


Digital user engagement


Predictive analytics

Analytics


Data Warehousing


Big data Processing


Serverless Data processing


Interactive Query


Operational Analytics


Real time Analytics

Data lake infrastructure & management


Infrastructure


Security & management


Data Catalog & ETL

Data movement

Migration & streaming services


zynga
Connect the world through games

Agenda

Importance of analytics at Zynga

Motivation to migrate

Migration options—why Amazon Redshift?

Database scorecard

Migration process

Learnings



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Importance of analytics at Zynga

AWS re:Invent

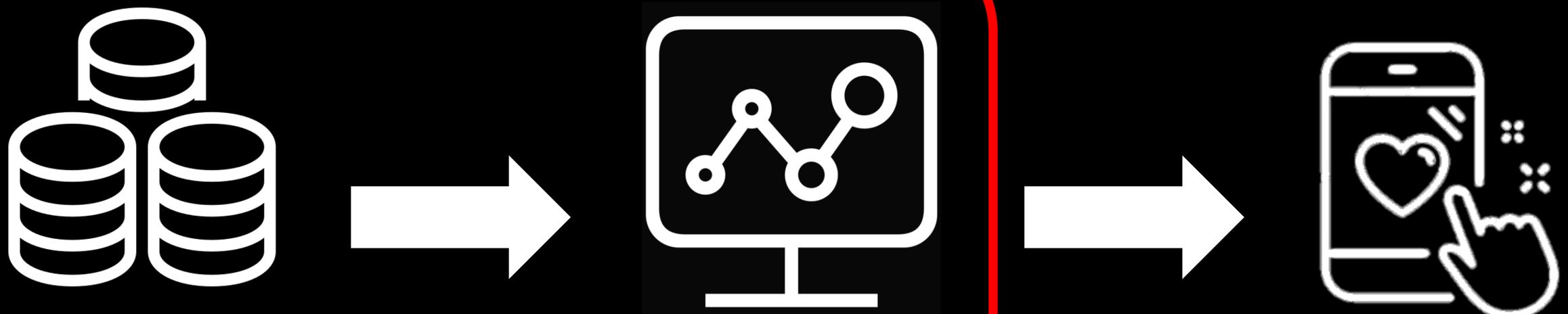
© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Data Analytics in Gaming



1. Collect
2. Analyze
3. Optimize

Data Analytics in Gaming



1. Collect
2. Analyze
3. Optimize

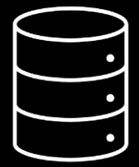
Data Infrastructure

Analytics at Zynga



Tens of Millions

Mobile MAU



A few TB

Uncompressed data per day



Tens of Billions of records

ingested per day



Millions of queries

per day

Motivation to migrate

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

A day in the life…

6:00 AM PT	Our central ETL has just completed
9:00 AM PT	Analysts and data scientists have just started their work
10:00 AM PT	Product managers are not able to query the latest data
10:30 AM PT	The data engineering team is handling resource issues

Current legacy warehouse



High maintenance overhead

45d/year on ops work

Maintain dual cluster



Inability to scale (storage and compute)

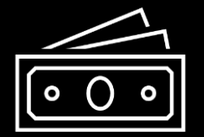
Store 8 weeks of data

Resource issues



Inelastic

Compute requiring over-provisioning



Expensive

Very low ROI

Why Amazon Redshift?



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Evaluation criteria for Zynga

Criteria ↓	Market solutions →
Maintenance overhead	
Scalability	
Cost	
Storage	
Performance	
Migration cost	
Compatibility with current tech	
Future product offerings	



Evaluation criteria for Zynga

Criteria ↓	Market solutions → Legacy
Maintenance overhead	D
Scalability	D
Cost	B
Storage	D
Performance	C
Migration cost	N/A
Compatibility with current tech	B
Future product offerings	D

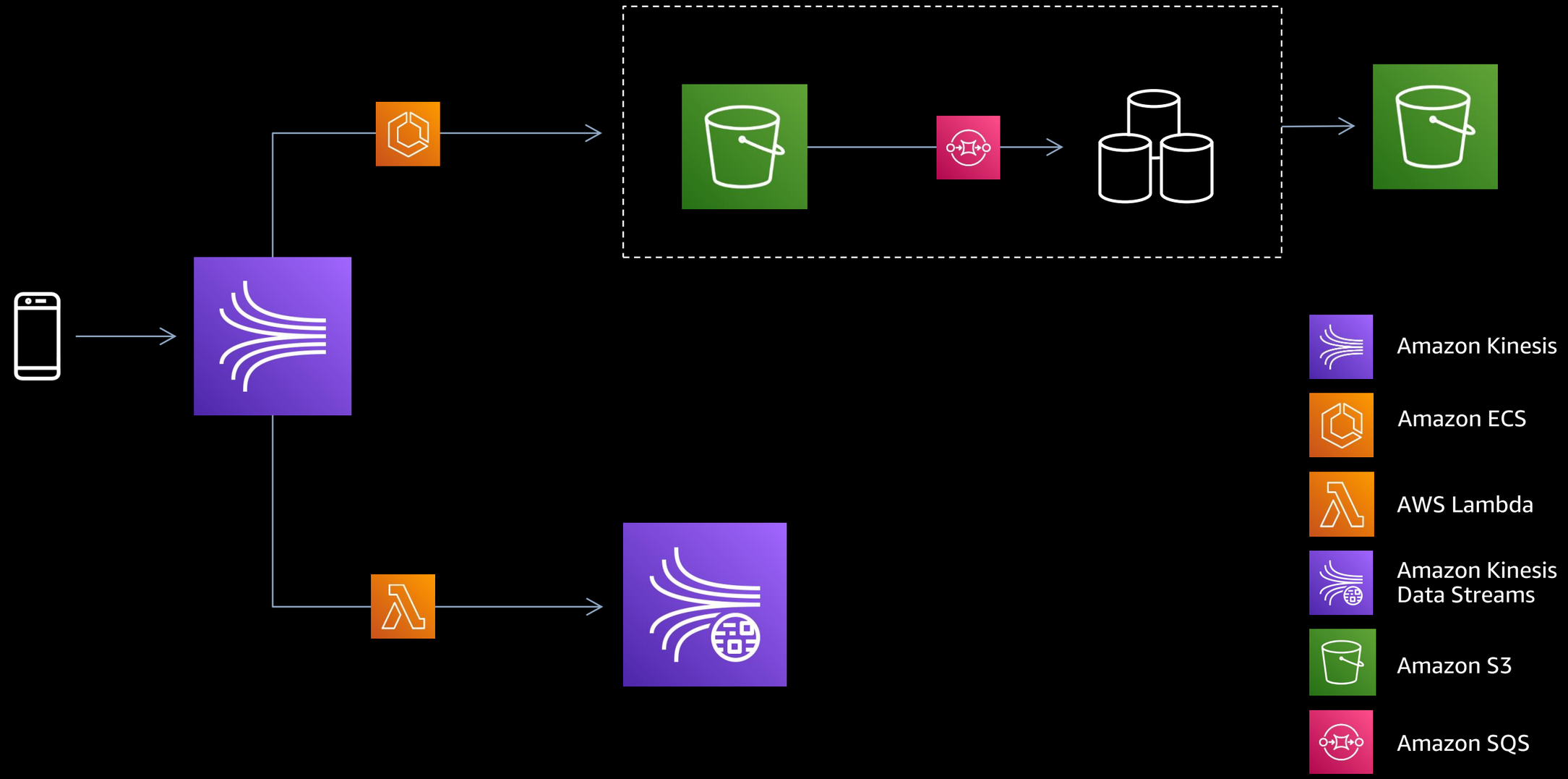
Evaluation criteria for Zynga

Criteria ↓ Market solutions →	Legacy	Solution B	
Maintenance overhead	D	A+	B+
Scalability	D	A+	A-
Cost	B	D	A
Storage	D	A-	A+
Performance	C	B	A+
Migration cost	N/A	D	B-
Compatibility with current tech	B	D	A
Future product offerings	D	B	A

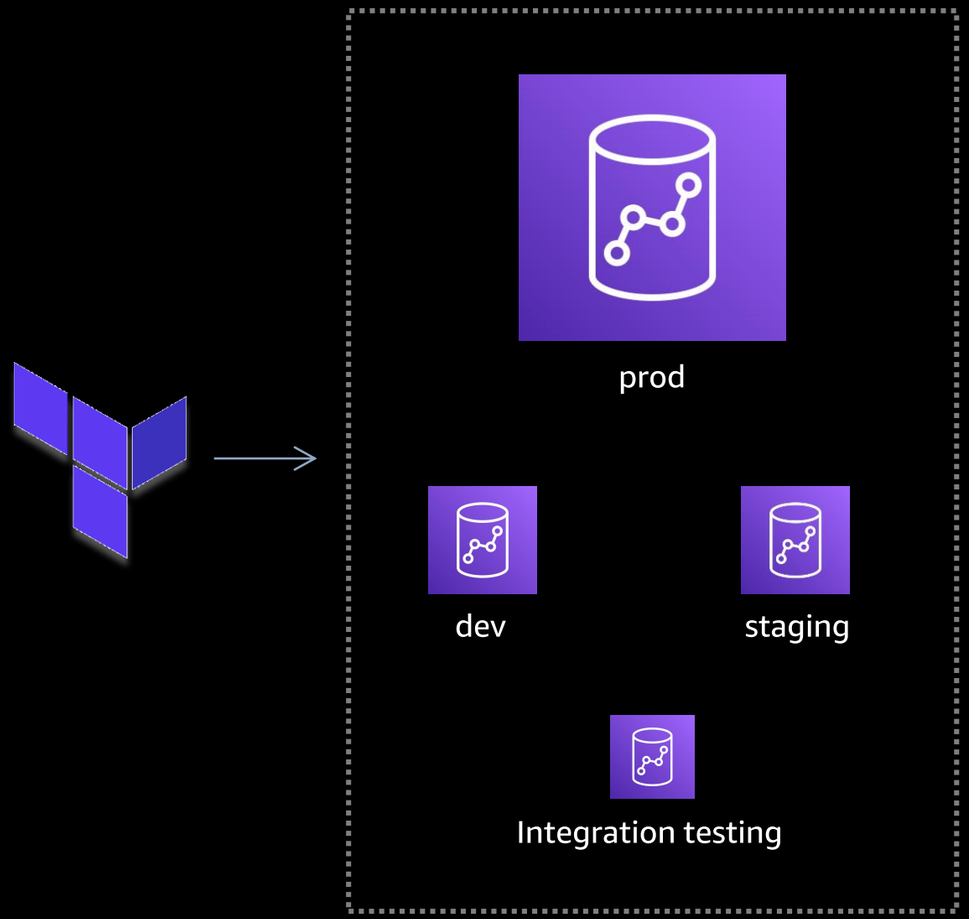
Migration

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

High level architecture



Phase 1: Provisioning

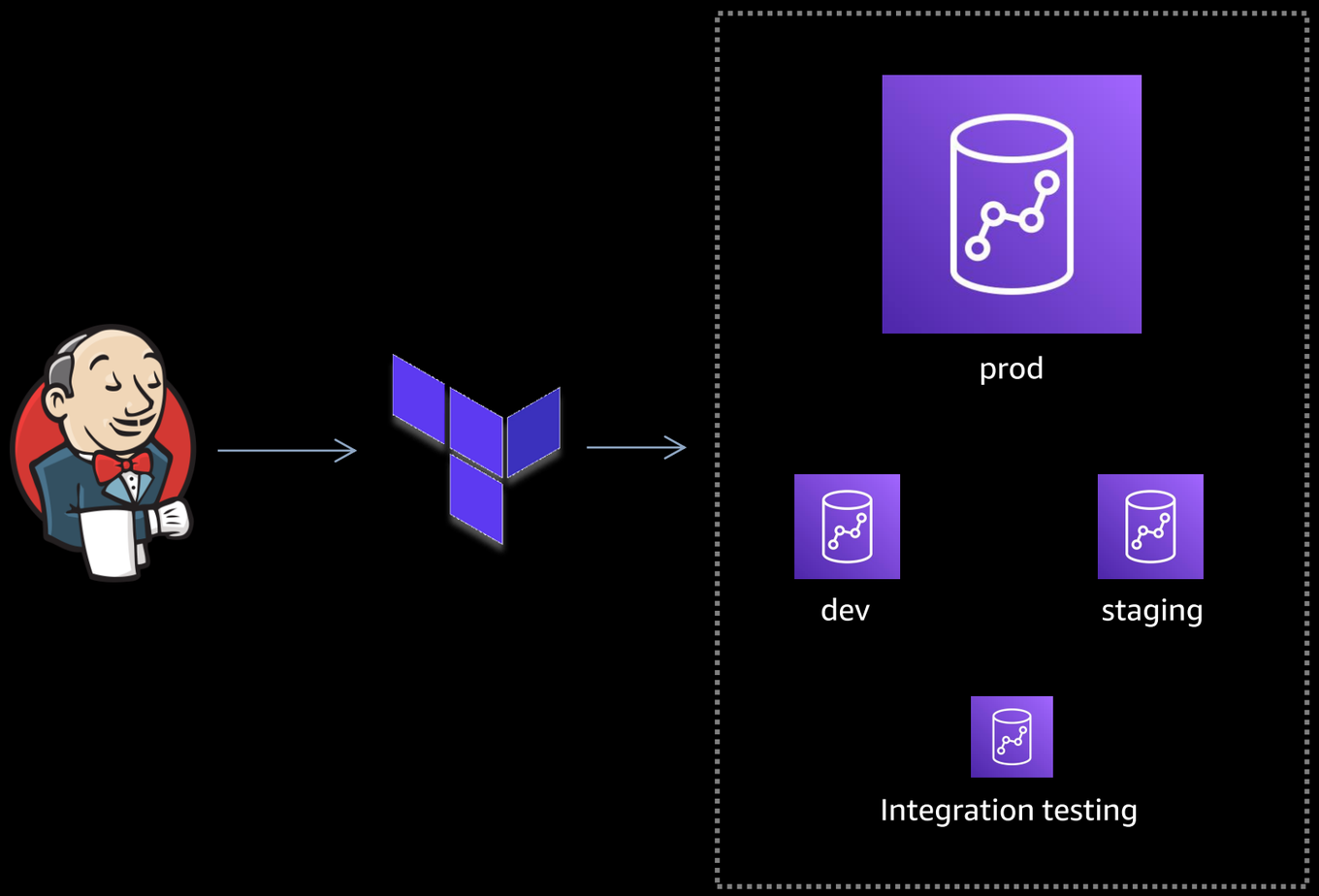


Infrastructure as Code

TF—Amazon Redshift cluster resource

Spinning up multiple resources

Phase 1: Provisioning



Infrastructure as Code

TF—Amazon Redshift cluster resource

Spinning up multiple resources

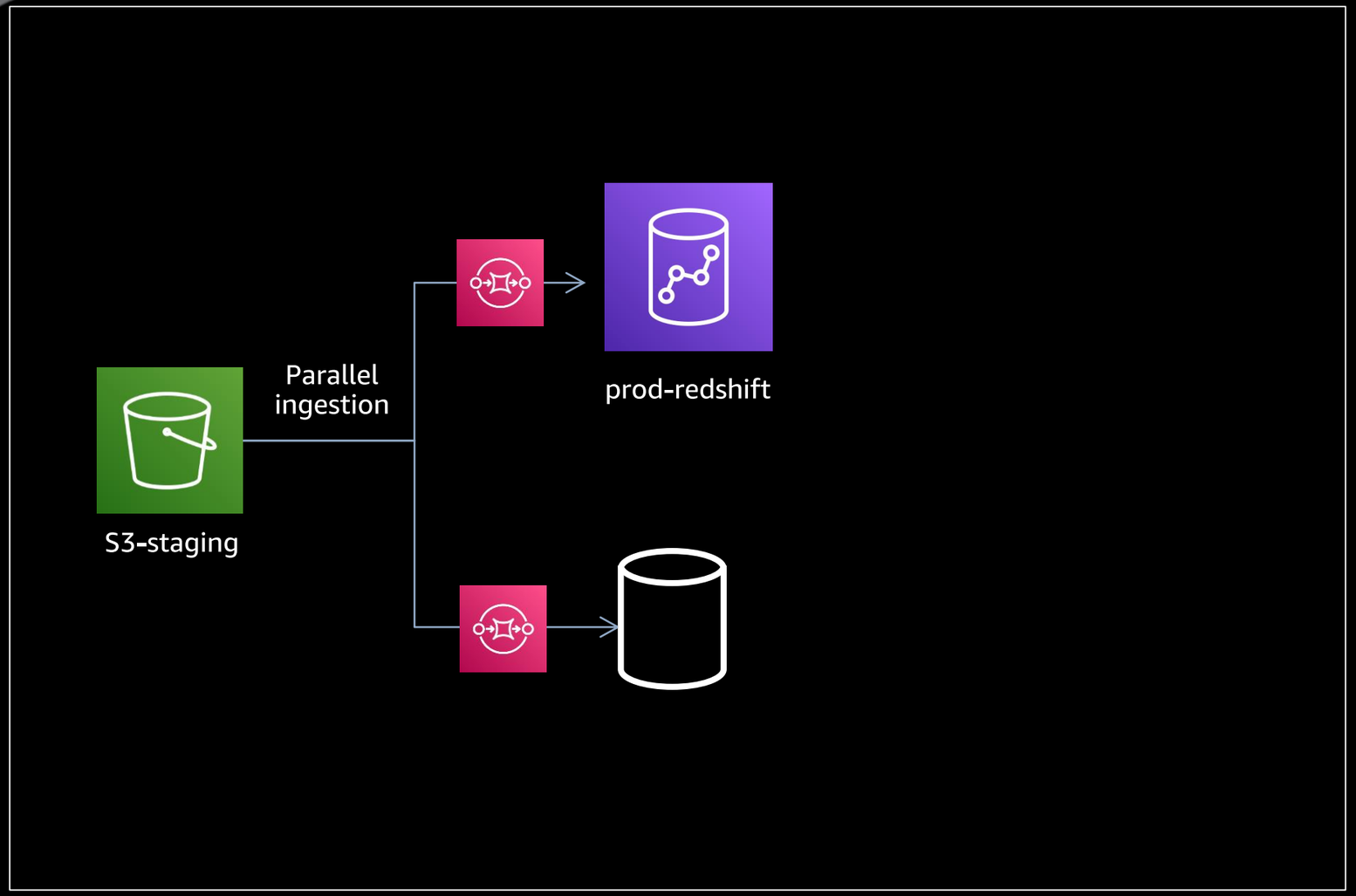
Jenkins to trigger TF apply

Maintains state history

Config:

N-RA3.16XL, Auto WLM, CS-2

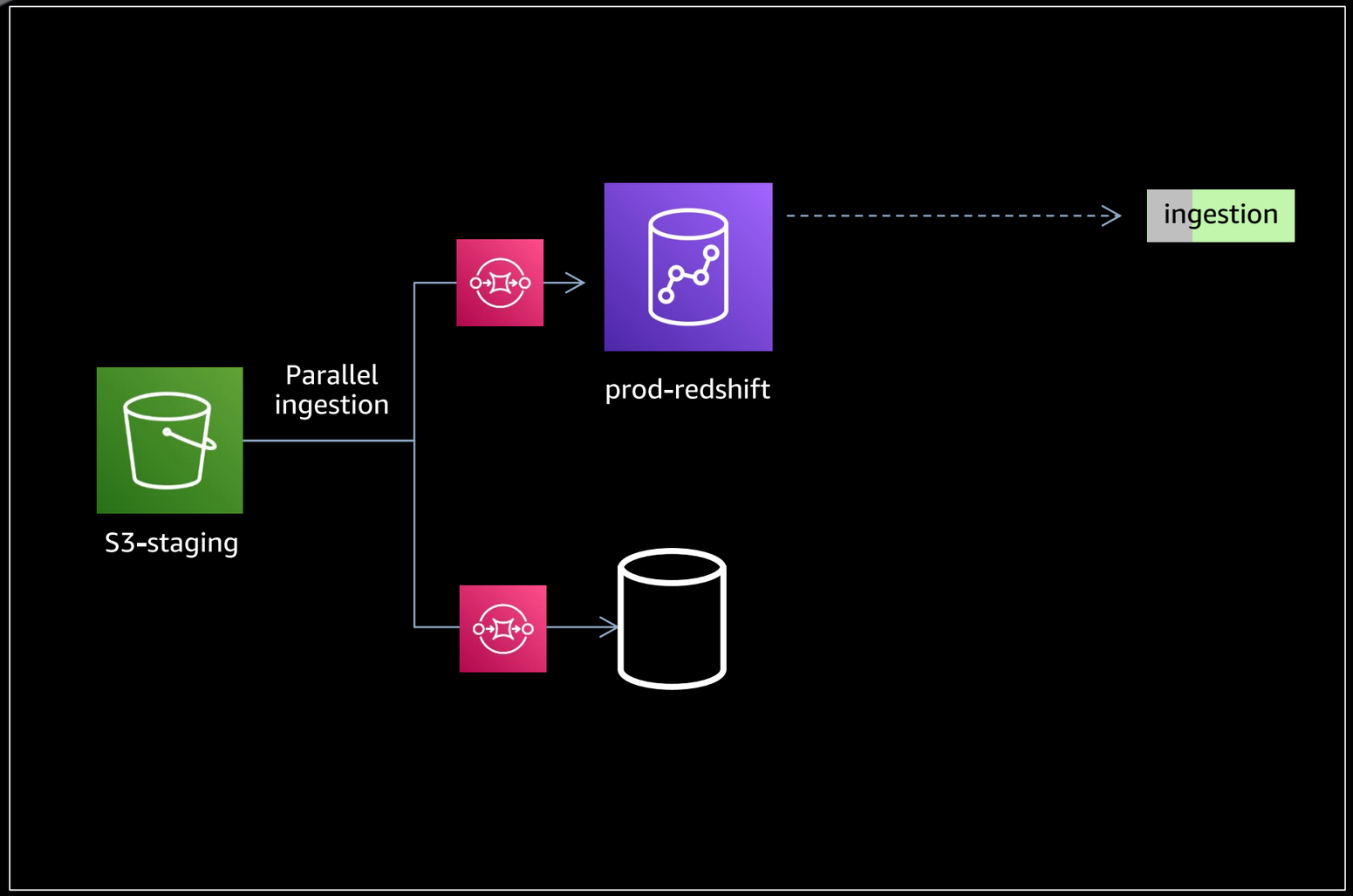
Phase 2: Data migration



Setup new consumers

Queryable data exposed via view

Phase 2: Data migration



Setup new consumers

Validate data

Evaluate performance

Key Takeaways:

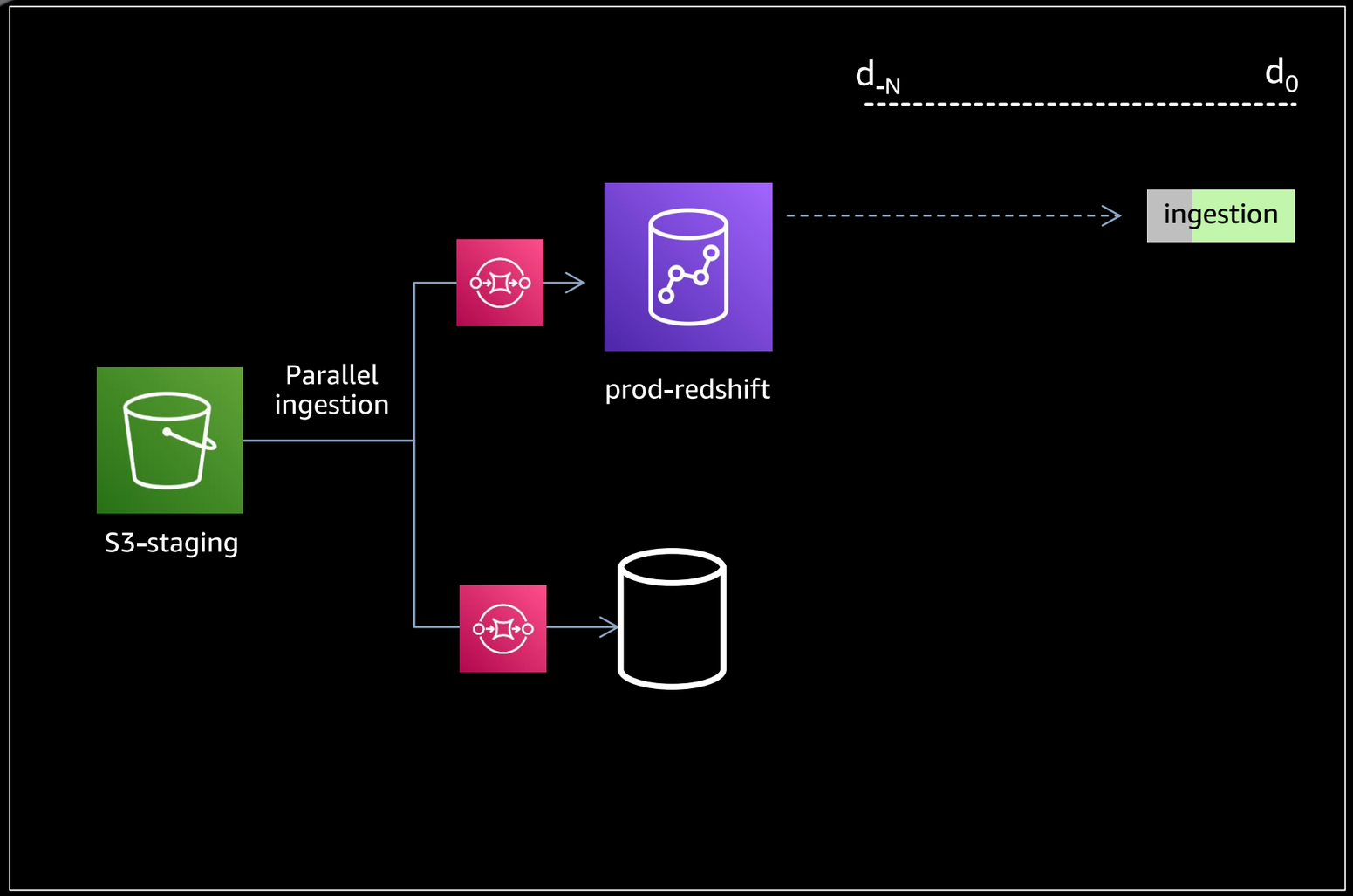
Compress your data files

Use a manifest file

Iterate on table encodings

Copy performance: 6x faster

Phase 2: Data migration

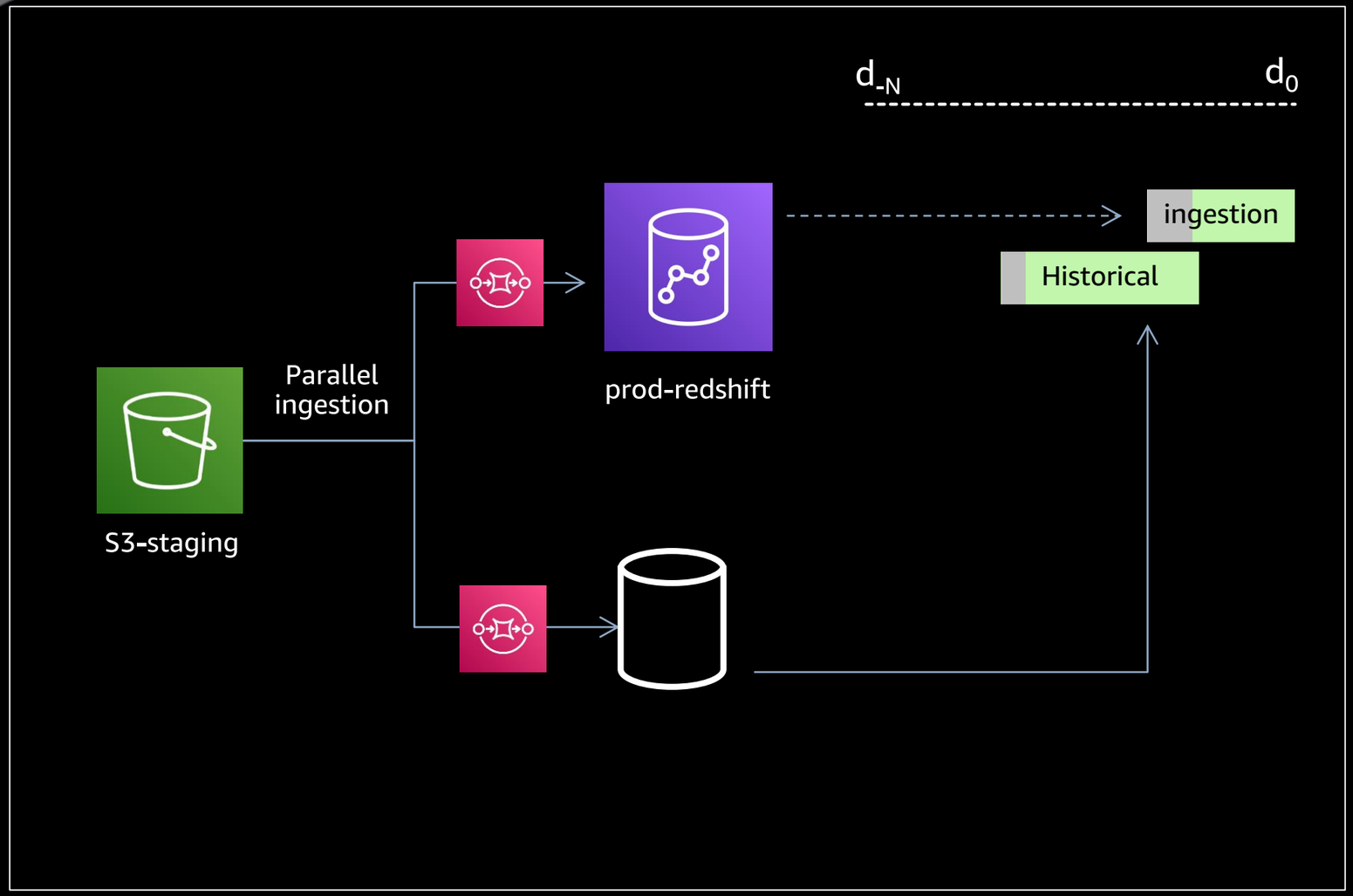


Setup new consumers

Validate data

Setup views

Phase 2: Data migration



Setup new consumers

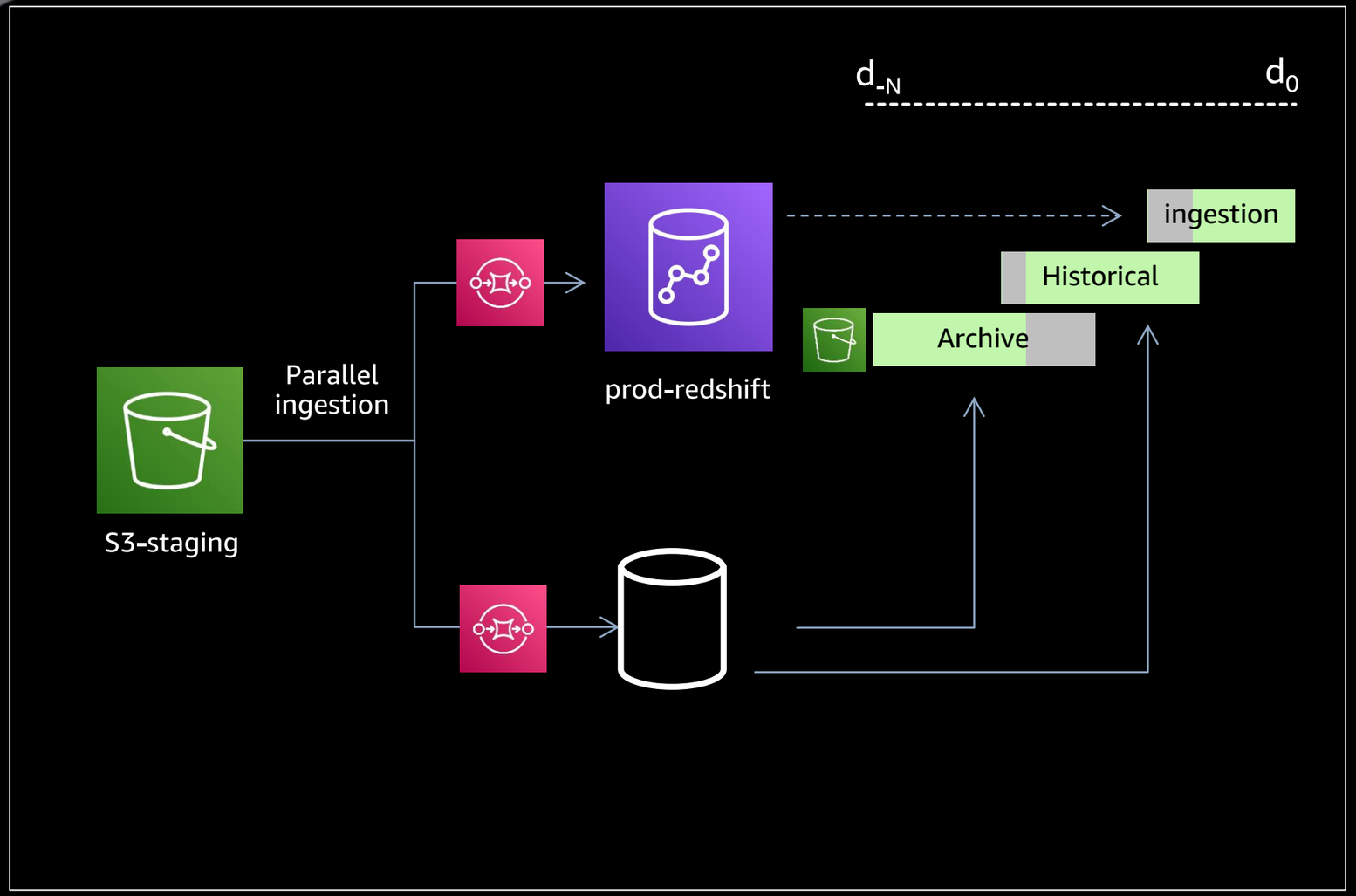
Validate data

Setup views

Migrate historical data

Benefit of migrating and ingesting data in sortkey order and in parallel

Phase 2: Data migration



Setup new consumers

Validate data

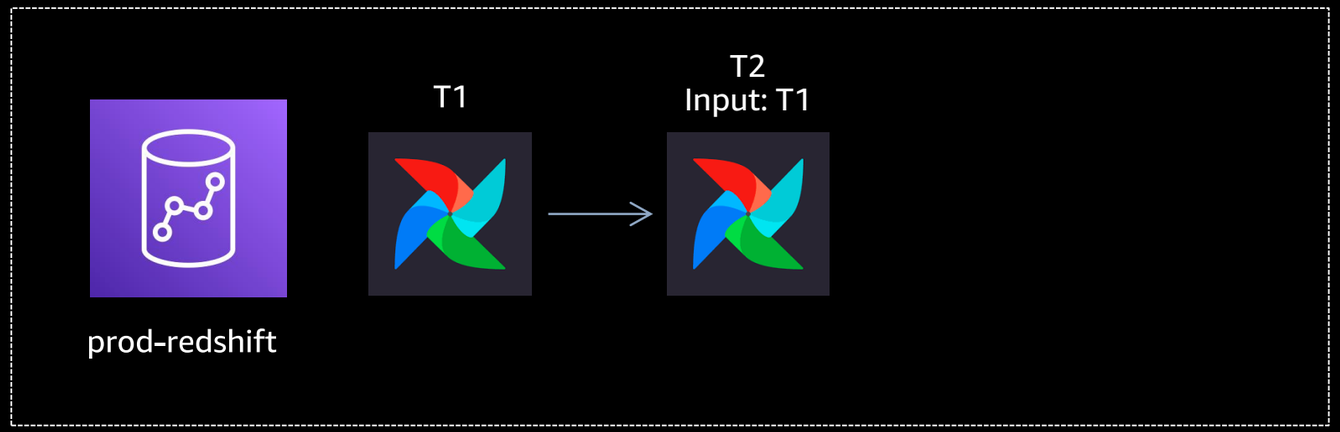
Setup views

Migrate historical data

Potential for archiving

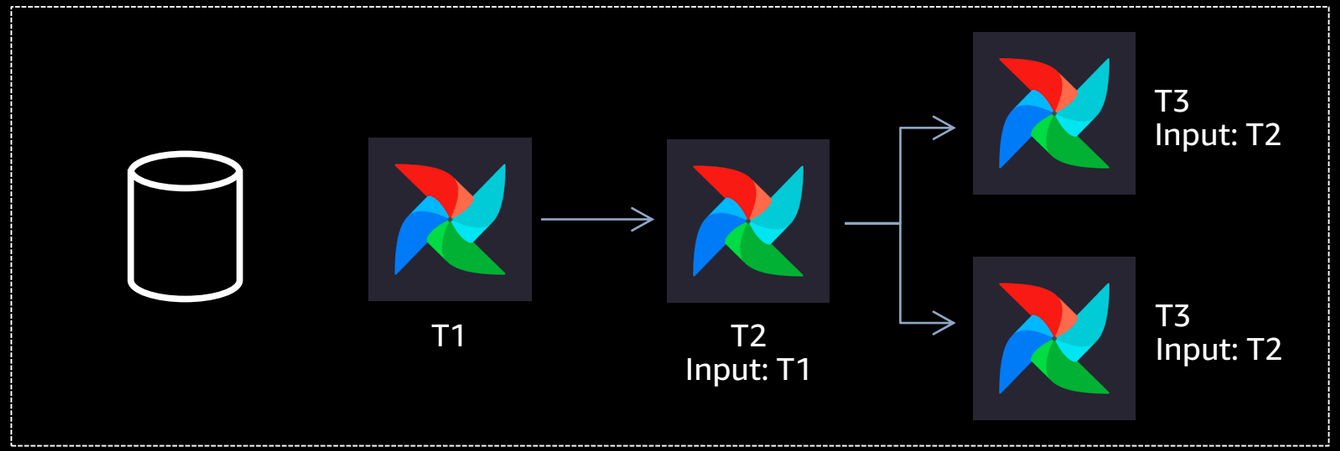
Spectrum Queries: 10x faster

Phase 3: ETL migration



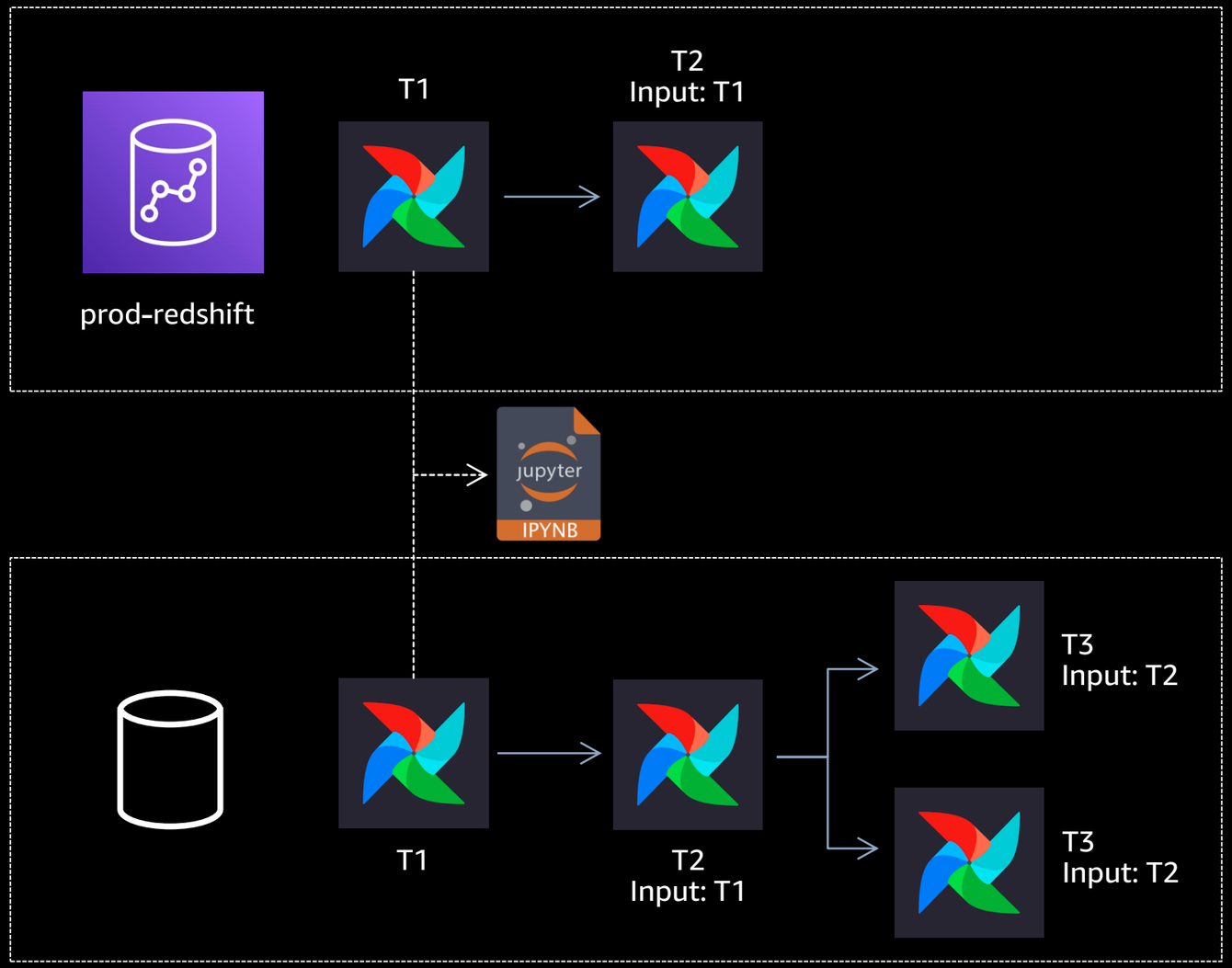
Build components for Amazon Redshift

Start with source of lineage



Airflow

Phase 3: ETL migration

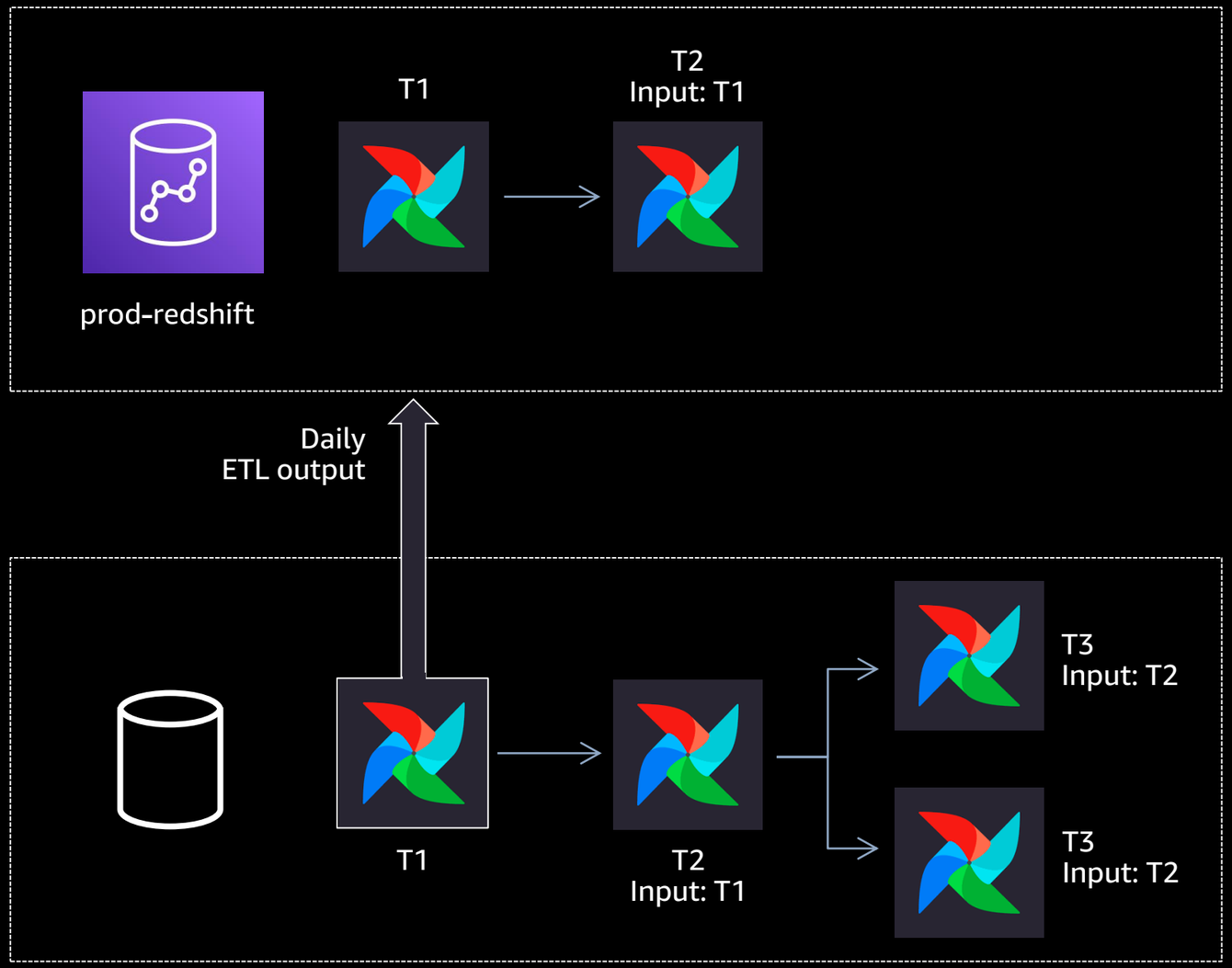


Build components for Amazon Redshift

Start with source of lineage

Validate data

Phase 3: ETL migration



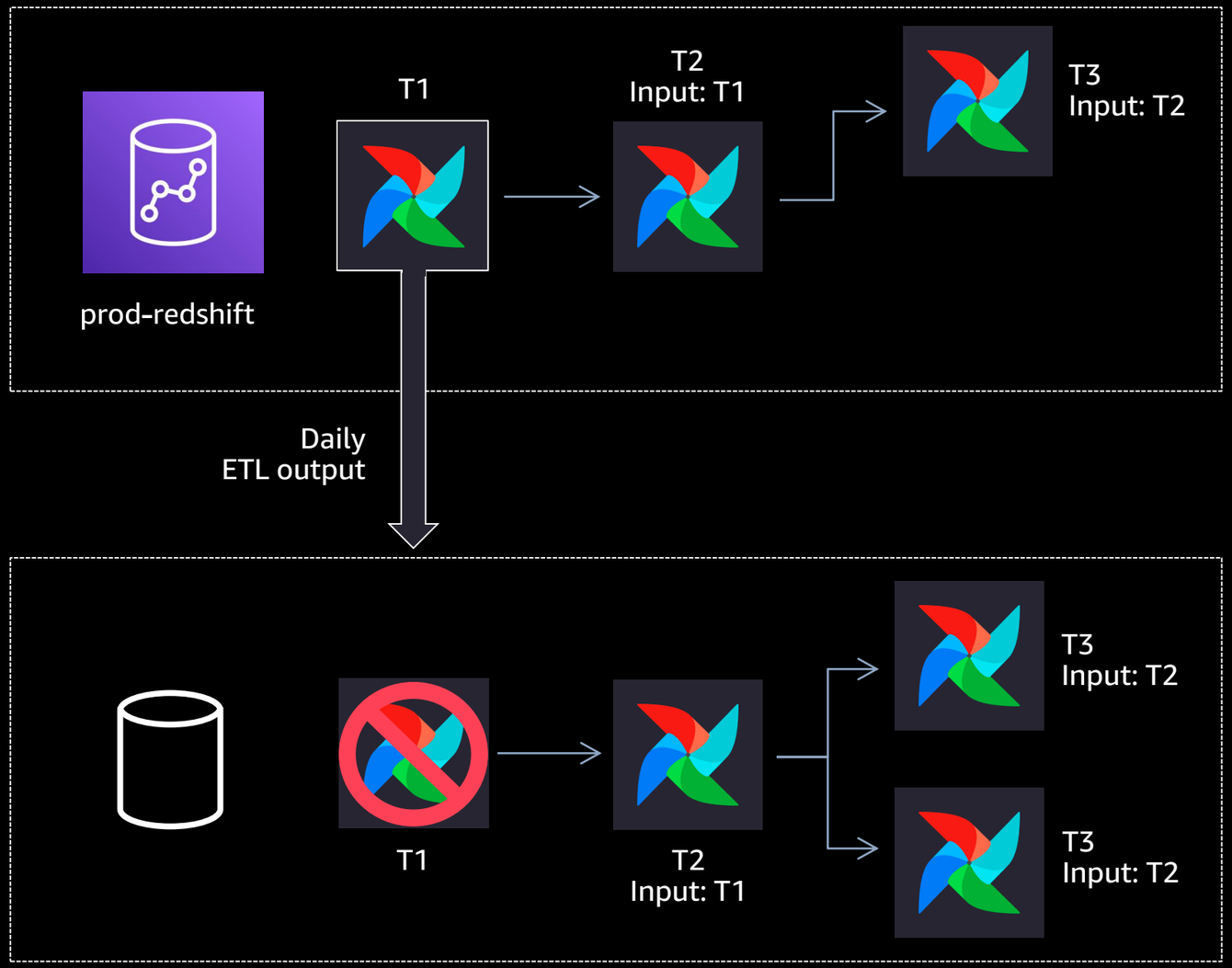
Build components for Amazon Redshift

Start with source of lineage

Validate data

Move historical outputs

Phase 3: ETL migration



Build components for Amazon Redshift

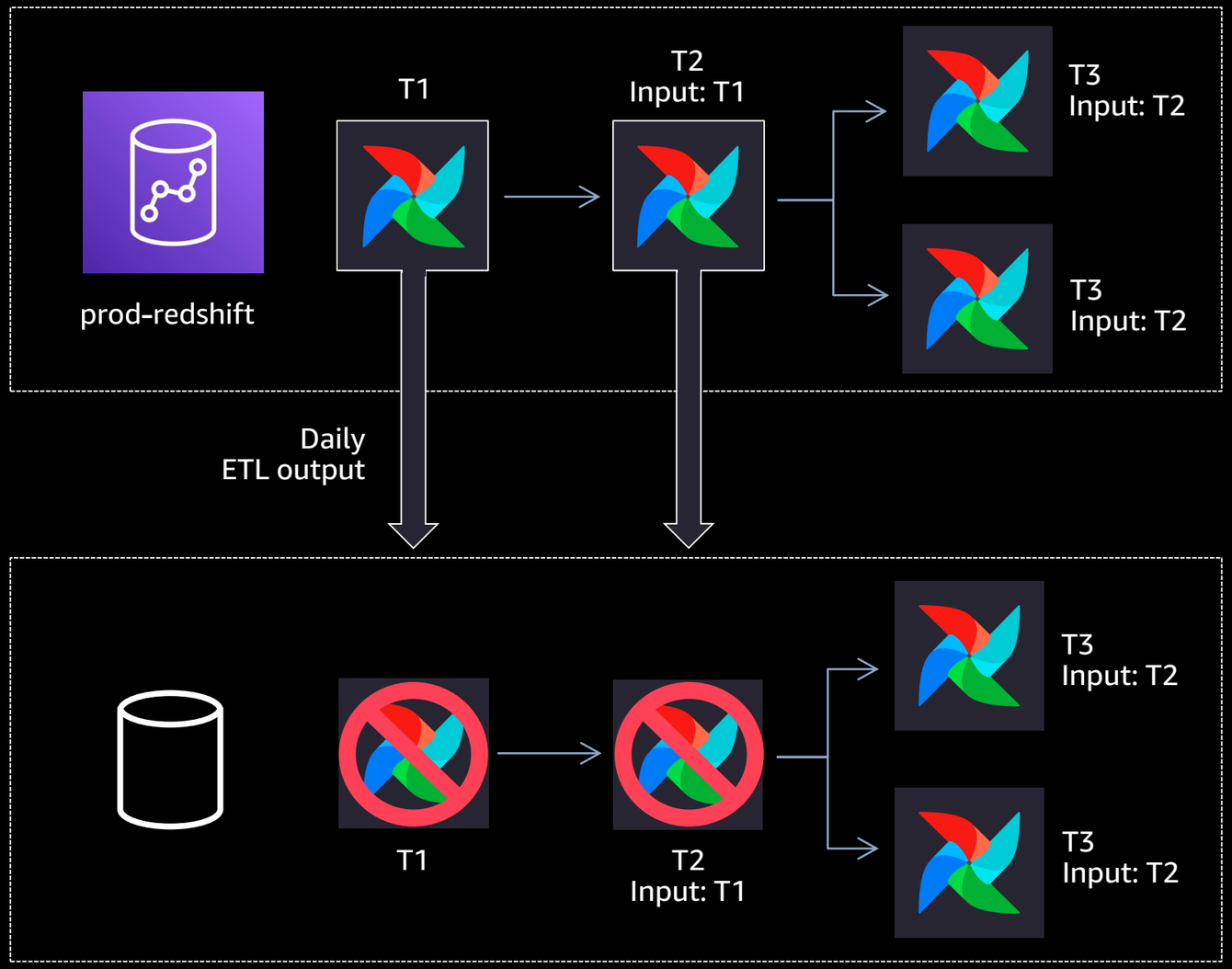
Start with source of lineage

Validate data

Move historical outputs

Move daily outputs

Phase 3: ETL migration



Build components for Amazon Redshift

Start with source of lineage

Validate data

Move historical outputs

Move daily outputs

Repeat

ETL performance: 2x faster



Phase 3: ETL migration

- Key takeaways
  - Perform DDL’s or multiple steps in transaction
  - Analyze and Vacuum as necessary
  - Use temporary tables to pre-sort and distribute your data for joins in ETL
  - Use unload to make most use of concurrency scaling

Phase 4: Consumer migration











prod-redshift

Other clients







- Understand query usage
- Start with known consumers
- Verify SSO and connections
- Create WLM groups
- Hunt down other clients

Phase 4: Consumer migration











prod-redshift

Other clients



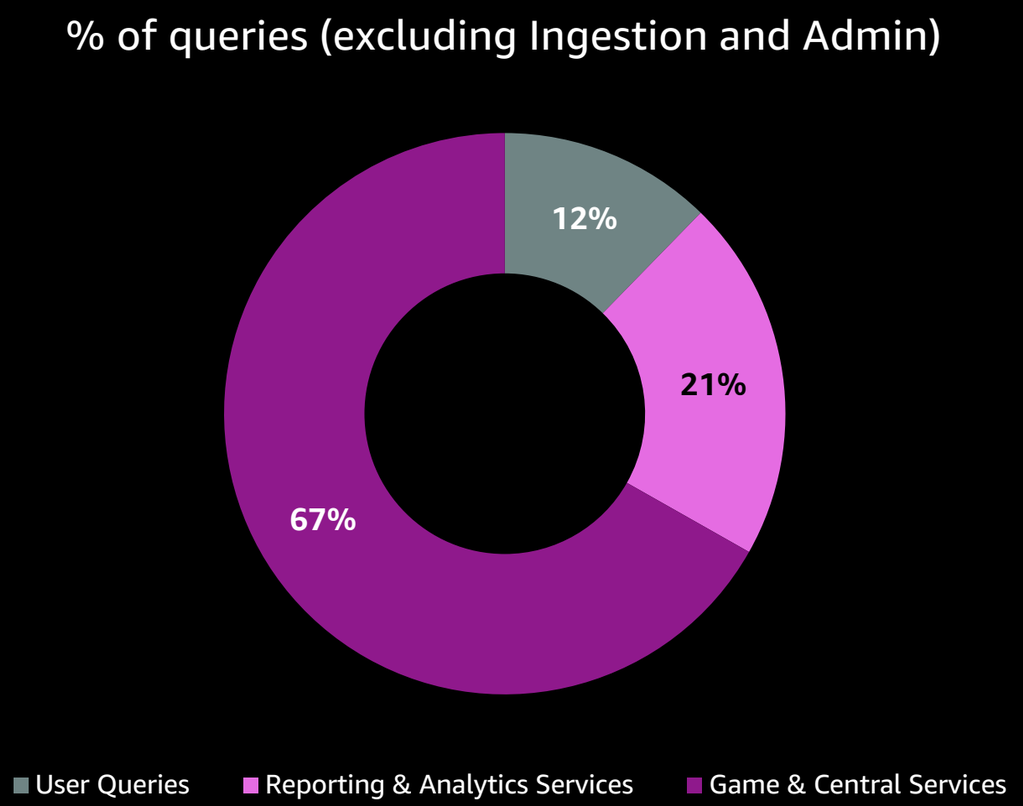




- Analysts
- Marketing
- Finance
- Data Scientists
- PM’s
- Game Dev’s
- Customer Support
- Monitoring

Phase 4: Consumer migration

Host	Language	Library/Driver
Ec2	Bash	Psql
	Go	sql-go adapter
	Java	Jdbc
	Node.js	db-npm
	Php	Pdo, pg_connect
	Python	Databricks, python, zeppelin/jdbc
	Ruby	PG gem
EKS	Go	sql-go adapter
	Java	Jdbc
	Php	PDO, pg_connect
EMR	Java	Spark/jdbc
Lambda	Python	python client
Windows	Java	Pentaho/jdbc



Phase 4: Consumer migration

Amazon Redshift Data API

DATA API—Improves Redshift developer experience and simplifies data access from different languages

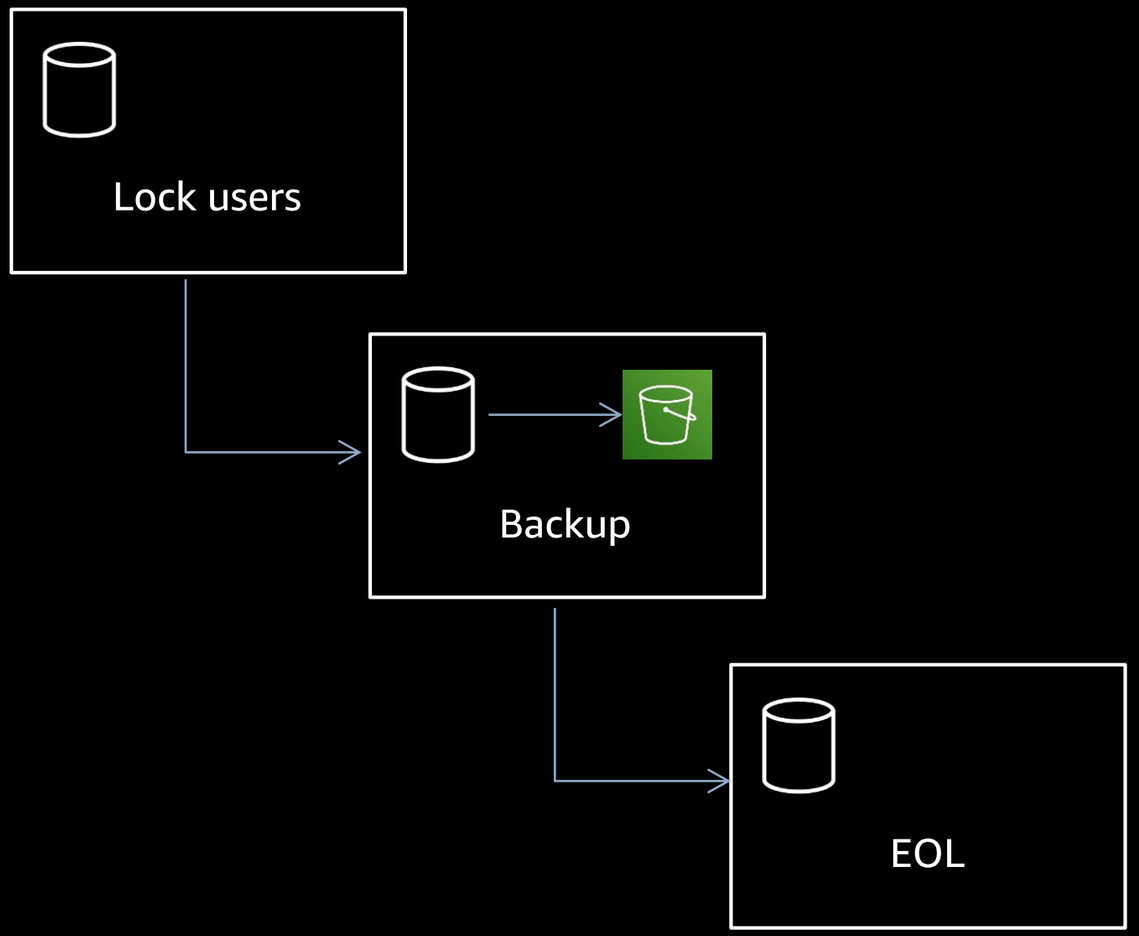
Fast and reliable performance—All Redshift work

Feature rich—Async, Caching, etc.

Consistent—One SDK

Secure—IAM Auth

Phase 5: Lockout and EOL



Over communicate timelines

Stop upstream services

Lock user accounts

Monitor for any activity

Take backup

Shut down system

Wins post migration

Consistent ingestion lag - We’re able to reduce the ingestion SLA

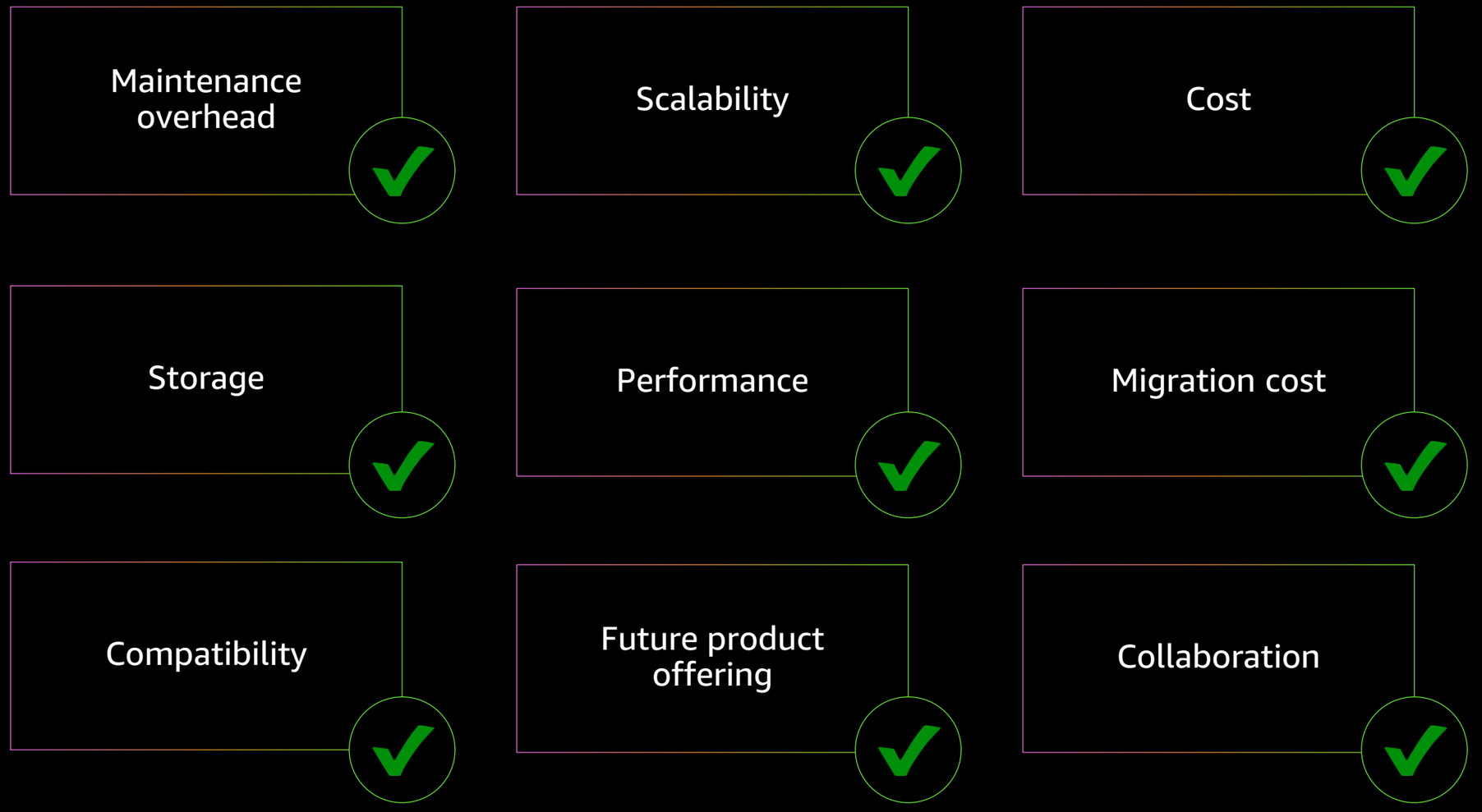
Improved copy performance - Increased throughput measured in mbps

Improved ETL performance

Scaling as needed

Long trend analysis

Revisit: Evaluation criteria



What’s next?

Spectrum, Athena, Glue unlocks building blocks for Data Lake

Current Ingestion opens more opportunity for near real time analysis

Amazon Redshift Federated query

Thank you!

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Please complete the session survey

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.