

Informe Técnico

- Proyecto Final - - Internship Teracloud -

Autor: *Ezequiel Coggiola*

Equipo: *Interns Data*

Fecha de entrega: *03/12/2025*

<i>Introducción</i>	2
<i>Objetivo del proyecto</i>	2
<i>Preguntas de negocio</i>	3
<i>Estructura</i>	4
<i>Modelo de Datos (Processed y Curated)</i>	5
<i>Recursos y configuraciones</i>	7
<i>Utilización</i>	16
<i>Estimación de costos y optimización</i>	18
<i>Análisis</i>	20
<i>Desafíos en el desarrollo</i>	22

Introducción

Este documento presenta el desarrollo completo de un pipeline ETL end-to-end implementado con servicios de AWS como parte del proyecto final del Internship Program de Teracloud. El objetivo del trabajo es diseñar, construir y documentar una arquitectura moderna de datos basada en un Data Lake que permita a un negocio e-commerce ficticio, TechStore, transformar sus datos brutos en información analítica lista para responder preguntas clave del negocio.

El proyecto abarca la construcción de la infraestructura, la creación del pipeline ETL, la organización de los datos en diferentes capas (raw, processed y curated), el análisis mediante consultas en Athena y la visualización mediante Amazon QuickSight. A lo largo del informe se detallan las decisiones técnicas tomadas, los desafíos surgidos durante el desarrollo y las optimizaciones de costo implementadas.

Objetivo del proyecto

TechStore es un e-commerce ficticio que desea migrar sus datos históricos a la nube y establecer un proceso automático y escalable para analizar tendencias comerciales, métricas operativas y comportamiento de clientes.

El proyecto tiene como objetivos principales:

- Construir un Data Lake en AWS usando S3 como almacenamiento centralizado.
- Establecer un pipeline ETL automático para procesar datos históricos y futuros.
- Definir tablas analíticas en la capa curated para responder preguntas de negocio.
- Permitir que un cliente no técnico pueda generar dashboards sin interactuar con la consola de AWS.

El dataset utilizado proviene de Kaggle y fue adaptado para simular una base transaccional completa del e-commerce. Está compuesto por siete tablas principales que cubren distintas etapas del proceso comercial, desde navegación hasta compra y post-venta. En total, el conjunto supera el millón de filas y aproximadamente 250 MB de datos en formato CSV.

Las tablas incluidas son:

- **orders**: 33.580 filas, 10 columnas. Contiene información de cada orden, incluyendo método de pago, descuentos, país, dispositivo y total de la compra.
- **order_items**: 59.163 filas, 5 columnas. Registra los productos incluidos en cada orden, sus precios unitarios, cantidades y montos finales.
- **products**: 1.197 filas, 6 columnas. Catálogo completo de productos, con información de precios, costos y márgenes.
- **reviews**: 10.780 filas, 6 columnas. Reseñas de productos asociadas a órdenes reales.
- **sessions**: 120.000 filas, 6 columnas. Sesiones de navegación con datos de dispositivo, fuente de tráfico y país.
- **customers**: 20.000 filas, 7 columnas. Información demográfica y de registro de clientes.
- **events**: 760.958 filas, 10 columnas. Eventos de interacción dentro de sesiones (clics, vistas, agregar al carrito, pagos, etc.).

Esta diversidad de tablas permite construir un pipeline robusto que cubre el funnel completo del negocio: adquisición, navegación, conversión y comportamiento post-compra.

Preguntas de negocio

Las preguntas de negocio que guían el diseño de las tablas curated y los dashboards se pueden resumir en tres preguntas simples que permiten entender rápidamente cómo está funcionando el e-commerce:

1) *¿Cómo está funcionando el negocio este mes?*

Buscamos ver ingresos, número de clientes, ticket promedio y variación respecto al mes anterior.

2) *¿En qué parte del funnel se pierden más usuarios?*

Analizamos las etapas clave del recorrido del usuario: page views → add to cart → checkout → purchase.

3) *¿Cómo se comportan los usuarios según país, canal y horarios?*

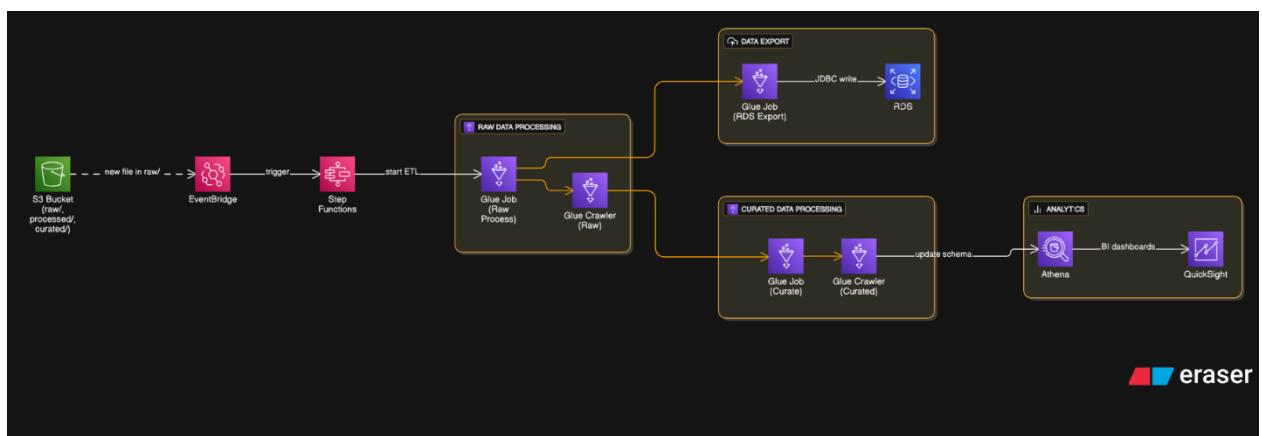
Permite entender qué mercados generan más ingresos, qué canales funcionan mejor y en qué momentos del día hay mayor actividad.

Estas tres preguntas son la base del análisis comercial y orientan directamente cómo se estructuraron las tablas curated y los dashboards finales.

Estructura

El sistema está basado en una arquitectura de Data Lake con procesamiento serverless mediante AWS Glue. A continuación se presenta una descripción clara y simplificada del recorrido que hacen los datos a lo largo del pipeline, sin entrar todavía en detalles técnicos de cada recurso.

Diagrama ETL propuesto



- 1. Carga de datos en S3 (raw/)** – A través de un script en Python.

Primero se cargan los datos históricos del negocio en la carpeta raw/ del bucket S3. Luego, cada mes, el cliente incorpora nuevos archivos mediante un script que valida el formato y los sube automáticamente, asegurando que todo el proceso comience siempre desde la misma capa de ingestión.

- 2. Job Glue 1:** Transformación de datos raw → processed.

Esta etapa toma los archivos crudos desde raw/ y los estandariza. Incluye limpieza de nulos, corrección de tipos, normalización de columnas y generación de particiones por año/mes/día. Finalmente, los datos se convierten a formato Parquet, lo que reduce el peso, acelera las consultas y disminuye costos.

- 3. Crawler 1:** Actualización del Glue Data Catalog.

Una vez generados los datos en la capa processed/, un crawler analiza el esquema y registra las tablas en el Glue Data Catalog. Esto permite que Athena pueda consultarlas inmediatamente.

- 4. Job Glue 2:** Enriquecimiento y modelado processed → curated.

Aquí se construyen las tablas analíticas finales. Se unen diferentes fuentes (órdenes, productos, clientes, sesiones, eventos), se crean nuevas columnas derivadas, se calculan métricas y se generan modelos orientados a las preguntas de negocio definidas. Cada tabla curated está diseñada para análisis directo.

- 5. Crawler 2:** Actualización del Glue Data Catalog para curated.

Igual que en el paso anterior, este crawler detecta las tablas generadas en curated/ y las registra en el catálogo para habilitar su consulta desde Athena.

6. **Athena:** Consultas SQL sobre processed y curated.

Con los datos catalogados, Athena permite ejecutar consultas SQL para análisis inmediato, generar vistas y validar los resultados del pipeline. Esta capa sirve también como fuente de datos para herramientas externas.

7. **QuickSight:** Dashboards conectados mediante Custom SQL.

QuickSight se conecta a Athena y extrae la información necesaria para construir dashboards analíticos, permitiendo visualizar métricas clave como ingresos, márgenes, tendencias temporales y comportamiento de clientes.

8. **Job Glue 3:** Carga de datos a RDS.

En caso de necesitar una base transaccional para consultas rápidas tipo OLTP/OLAP híbrido, los datos procesados pueden enviarse a RDS, donde cumplen funciones operativas adicionales.

Modelo de Datos (Processed y Curated)

El modelo final incluye:

Tablas Processed:

- Copias limpias y tipificadas de las tablas originales.
- Formato Parquet para mejorar la performance.
- Particiones por año/mes/día cuando corresponde.

Tablas Curated:

La capa curated/ contiene las tablas analíticas finales listas para consulta desde Athena y QuickSight. Estas tablas incluyen enriquecimientos, joins entre fuentes, columnas derivadas y métricas orientadas directamente a las preguntas de negocio.

1. fact_order_items_enriched

Tabla de hechos centrada en los ítems de cada orden, con información adicional de productos y métricas de rentabilidad.

Columnas principales:

- order_id
- product_id
- quantity
- unit_price_usd
- line_total_usd
- category
- price_usd
- cost_usd
- profit_per_unit ($unit_price_usd - cost_usd$)
- total_profit ($line_total_usd - cost_usd \times quantity$)
- year (partición)

Uso analítico: análisis de márgenes, ranking de productos, desempeño por categoría.

2. fact_orders_enriched

Tabla de órdenes enriquecida con información del cliente y atributos temporales para análisis de tendencias.

Columnas principales:

- order_id
- customer_id
- order_time
- order_date
- total_usd
- discount_pct
- order_country
- device
- source

- age
- customer_country
- marketing_opt_in
- year, month (*particiones*)

Uso analítico: ingresos mensuales, segmentación por país/canal/dispositivo, ticket promedio, cohortes simples.

3. events_enriched

Tabla granular de eventos utilizada para construir funnels de conversión y analizar el comportamiento del usuario.

Columnas principales:

- event_id
- session_id
- timestamp
- event_date
- event_type
- product_id
- qty
- cart_size
- payment
- discount_pct
- amount_usd
- year, month (*particiones*)

Estas tablas componen el modelo analítico final y permiten responder de forma directa las tres preguntas de negocio planteadas al inicio del informe.

Recursos y configuraciones

Bucket S3

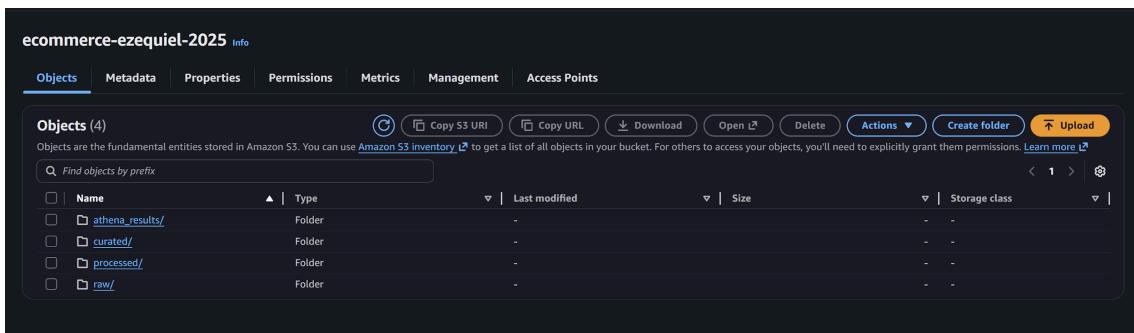
El bucket S3 es el núcleo del Data Lake y cumple la función de almacenamiento centralizado. Su estructura está diseñada siguiendo el modelo por capas recomendado para pipelines analíticos.

Estructura del bucket:

- raw/ → Archivos originales en CSV exactamente como llegan del negocio.
- processed/ → Datos limpios y convertidos a Parquet, con particiones.
- curated/ → Tablas analíticas finales listas para Athena y QuickSight.

Configuraciones clave:

- Versionado habilitado para evitar pérdida de datos.
- Compresión automática mediante Parquet + Snappy.
- Lifecycle Rules para mover histórico a S3 IA.



Glue Jobs

Cada Glue Job automatiza un paso del pipeline. Las configuraciones principales incluyen:

Parámetros técnicos:

- 2 DPUs para equilibrio entre costo y velocidad.
- Uso de scripts en Spark.
- Timeout configurado para evitar cobros innecesarios.
- Conexión a S3 mediante GlueContext.
- Job Bookmarks habilitados para evitar reprocesamiento.
- Logging completo en CloudWatch.

Glue Job 1 - Raw → Processed

- Limpieza de nulos.
- Conversión de formatos.

- Normalización de tipos.
- Generación de particiones por fecha.

job-ecommerce-1

Last modified on 12/2/2025, 12:01:57 PM Actions Save Run

Script Job details Runs Data quality Schedules Version Control

Job runs (1/9) Info Last updated (UTC) December 3, 2025 at 05:14:08 View details Stop job run Troubleshoot with AI Table View Card View

Run status Retries Start time (Local) End time (Local) Duration Capacity (DPU) Worker type Glue version

Succeeded 0 12/03/2025 00:03:04 12/03/2025 00:05:09 1 m 46 s 2 DPU G.1X 5.0

Run details Input arguments (11) Logs Run insights Metrics Troubleshooting analysis - preview Spark UI Rewind job bookmark

Job name: job-ecommerce-1
Id: jr_a2107ef828e7168c9cd19b44ab235bbe9fa3dbb9bb19dcbbd19f11/25/2025 11:43:04
Run status: Succeeded
Retry attempt number: 0
Initial run: 33 minutes 23 seconds
Trigger name: -
Job run queuing: False

Start time (Local): 11/25/2025 11:09:34
End time (Local): 11/25/2025 11:14:08
Duration: 1 m 46 s
Capacity (DPU): 2 DPU
Worker type: G.1X
Glue version: 5.0
Log group name: /aws-glue/jobs
Max capacity: 10 DPU
Execution class: Standard
Cloudwatch logs:

- Output logs
- Error logs

Number of workers: 10
Timeout: 480 minutes
Usage profile: -

Vemos la primera ejecución de 30 minutos donde se leen todos los archivos y la mas reciente (arriba), donde solo se leen los archivos nuevos (ejecutado por la StateMachine), que dura solo 2 minutos .

Glue Job 2 - Processed → Curated

- Joins entre datasets.
- Cálculo de KPIs.
- Deduplicaciones.
- Modelos analíticos.

job-ecommerce-3-test

Last modified on 12/2/2025, 11:57:07 PM Actions Save Run

Script Job details Runs Data quality Schedules Version Control

Job runs (1/1) Info Last updated (UTC) December 3, 2025 at 05:17:50 View details Stop job run Troubleshoot with AI Table View Card View

Run status Retries Start time (Local) End time (Local) Duration Capacity (DPU) Worker type Glue version

Succeeded 0 12/03/2025 00:09:21 12/03/2025 00:28:56 19 m 29 s 2 DPU G.1X 5.0

Run details Input arguments (12) Logs Run insights Metrics Troubleshooting analysis - preview Spark UI Rewind job bookmark

Job name: job-ecommerce-3-test
Last modified on (Local): 12/03/2025 00:28:56
Run status: Succeeded
Retry attempt number: 0
Initial run: 19 minutes 29 seconds
Trigger name: -
Usage profile: -
Job run queuing: False

Start time (Local): 12/03/2025 00:09:21
End time (Local): 12/03/2025 00:28:56
Duration: 19 m 29 s
Capacity (DPU): 2 DPU
Worker type: G.1X
Glue version: 5.0
Log group name: /aws-glue/jobs
Max capacity: 2 DPU
Execution class: Standard
Cloudwatch logs:

- Output logs
- Error logs

Glue Job 3 – Processed → RDS

Se cargan datos procesados a la base de datos en RDS usando una conexión por JDBC.

Se perdió la ejecución exitosa, muestro datos en RDS para validar.

Glue Crawlers

Los crawlers permiten que Glue detecte automáticamente estructuras de tablas.

Configuraciones principales:

- Crawlers independientes para raw, processed y curated.
- Etiquetas y bases de datos separadas en el Catalog.
- Execution schedule: manual o via Step Functions.
- Detección de particiones habilitada.

crawler-ecommerce-processed

Last updated (UTC) December 3, 2025 at 05:02:56 [Run crawler](#) [Edit](#) [Delete](#)

Crawler properties

Name: crawler-ecommerce-processed	IAM role: ecommerce-ezequiel-glue-role	Database: ecommerce-ezequiel-processed	State: READY
Description: -	Security configuration: -	Lake Formation configuration: -	Table prefix: -
Maximum table threshold: -			

Advanced settings

[Crawler runs](#) [Schedule](#) [Data sources](#) [Classifiers](#) [Tags](#)

Crawler runs (6)

The list of crawler runs for this crawler.

Filter data Filter by a date and time range

Start time (UTC)	End time (UTC)	Current/last duration	Status	DPU hours	Table changes
December 2, 2025 at 16:34:56	December 2, 2025 at 16:35:17	41 s	Completed	0.143	-

crawler-ecommerce-curated

Last updated (UTC) December 3, 2025 at 05:02:14 [Run crawler](#) [Edit](#) [Delete](#)

Crawler properties

Name: crawler-ecommerce-curated	IAM role: ecommerce-ezequiel-glue-role	Database: ecommerce-ezequiel-curated	State: READY
Description: -	Security configuration: -	Lake Formation configuration: -	Table prefix: -
Maximum table threshold: -			

Advanced settings

[Crawler runs](#) [Schedule](#) [Data sources](#) [Classifiers](#) [Tags](#)

Crawler runs (1)

The list of crawler runs for this crawler.

Filter data Filter by a date and time range

Start time (UTC)	End time (UTC)	Current/last duration	Status	DPU hours	Table changes
December 2, 2025 at 02:07:03	December 2, 2025 at 02:07:52	48 s	Completed	0.127	2 table changes, 143 partition changes

RDS MySQL

Aunque el Data Lake es la principal fuente analítica, se utilizó una instancia RDS para almacenar información transaccional.

Características:

- Instancia db.t3.micro (free tier compatible).
- Base de datos privada en VPC.
- Security Groups limitando acceso solo a Glue y a la máquina del desarrollador.

database-ecommerce

Summary

DB identifier: database-ecommerce	Status: Stopped temporarily	Role: Instance	Engine: MySQL Community	Recommendations: 2 informational
CPU: -	Class: db.t3.micro	Current activity: -	Region & AZ: us-east-1a	

[Actions](#)

[Connectivity & security](#) [Monitoring](#) [Logs & events](#) [Configuration](#) [Zero-ETL integrations](#) [Maintenance & backups](#) [Data migrations](#) [Tags](#) [Recommendations](#)

Connectivity & security

Endpoint: database-ecommerce.cw7qa2zodino.us-east-1.rds.amazonaws.com
Port: 3306

Networking

Availability Zone: us-east-1b
VPC: vpc-0cccd95ec8368b40
Subnet group: default
Subnets:
subnet-0e802d5a62x2102x8
subnet-098981002380a95139
subnet-0f7717470e460004
subnet-05552934a6f1f543
subnet-05552934a6f1f543
subnet-03094a657e3fb148

Network type: IPv4

Security

VPC security groups: rds-sa-2025 (sg-09cf0fbfa5181926)
Publicly accessible: Yes
Certificate authority info: rds-ca-rsa2048-g1
Certificate creation date: May 25, 2026, 20:34 (UTC-03:00)
DB instance certificate expiration date: November 26, 2026, 11:51 (UTC-03:00)

- Conexión JDBC utilizada para cargar datos en processed/.

The screenshot shows the 'Connection details' section of the AWS Glue Connection configuration. The connection type is JDBC, with the URL set to `jdbc:mysql://database-ecommerce.cw7qa2iodno.us-east-1.rds.amazonaws.com:3306/database_ecommerce`. Other settings include Driver class name (com.mysql.jdbc.Driver), Username (admin), and Subnet (subnet-0a892d5a62a2102e8). The connection was created on 2025-11-27 15:57:31.939000. Below this is a 'Tags (0)' section where users can add key-value pairs. At the bottom, there's a table titled 'Your jobs (1)' showing one job named 'job-commerce-5'.

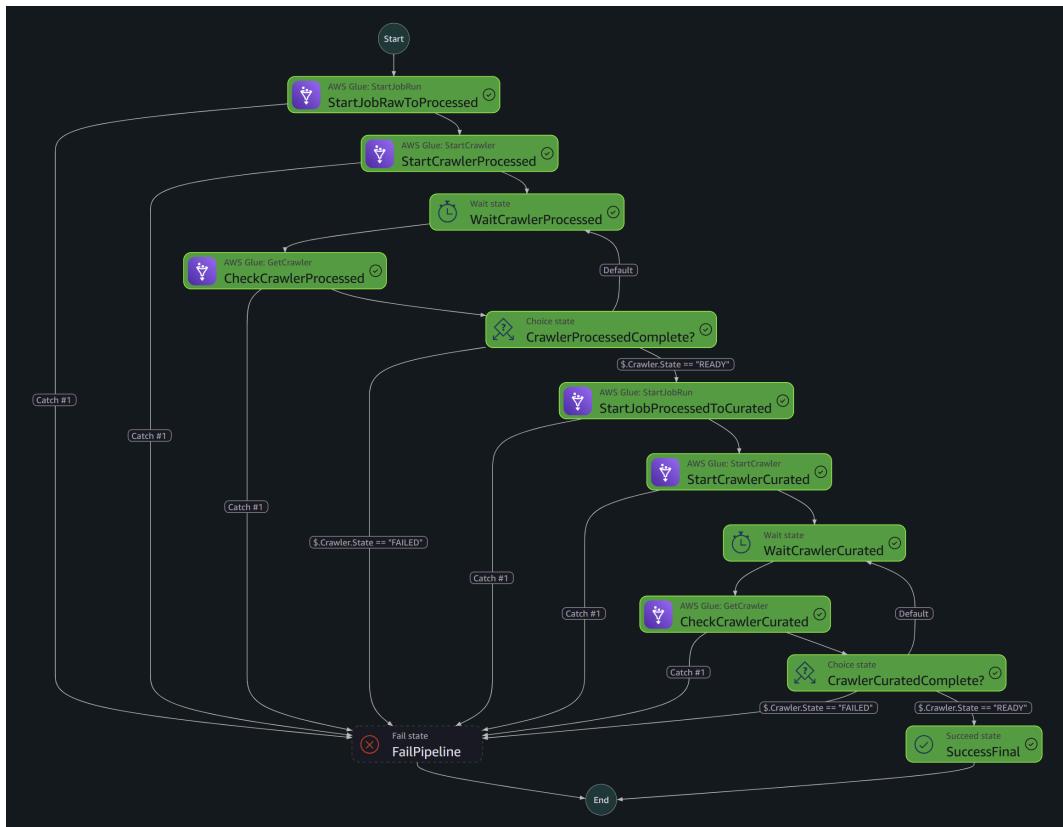
Step Function

Step Functions orquesta la ejecución automatizada del pipeline y garantiza que cada etapa se ejecute en orden, con manejo centralizado de errores y validaciones entre pasos. La definición utilizada en este proyecto sigue un flujo secuencial, con tareas sincrónas y controles explícitos para los crawlers.

Decisiones principales del diseño:

- **Pipeline 100% secuencial:** no se ejecuta ningún paso hasta que el anterior finaliza correctamente.
- **Ejecución sincrona de Glue Jobs:** se usa `glue:startJobRun.sync`, lo que obliga a Step Functions a esperar hasta que el job termine.
- **Control explícito de Crawlers:** cada crawler se inicia, se espera un intervalo fijo y luego se consulta su estado. No se avanza hasta que el crawler esté en estado READY.
- **Manejo de errores centralizado:** cualquier error en cualquier estado envía el pipeline a FailPipeline.

- Sin SNS ni notificaciones automáticas por ahora: la definición actual no incluye un paso final de notificación, aunque puede añadirse.



EventBridge

Regla que dispara la StateMachine cuando detecta un archivo nuevo en la carpeta raw/ del bucket.

IAM Roles

Para garantizar seguridad y buenas prácticas se crearon roles separados:

Roles principales:

GlueRole: Permisos de lectura/escritura en S3 + GlueServiceRole.

ecommerce-ezequiel-glue-role [Info](#)

Summary

Creation date
November 20, 2025, 10:38 (UTC-03:00)

Last activity
7 minutes ago

ARN
arn:aws:iam::371872301872:role/ecommerce-ezequiel-glue-role

Maximum session duration
1 hour

[Edit](#)

Permissions [Trust relationships](#) [Tags](#) [Last Accessed](#) [Revoke sessions](#)

Permissions policies (2) [Info](#)

You can attach up to 10 managed policies.

Policy name	Type	Attached entities
AWSGlueServiceRole	AWS managed	2
ecommerce-ezequiel-glue-extra	Customer inline	0

[Filter by Type](#) [Search](#) [Simulate](#) [Remove](#) [Add permissions](#)

Policy editor

```

1  {
2      "Version": "2012-10-17",
3      "Statement": [
4          {
5              "Effect": "Allow",
6              "Action": [
7                  "s3:PutObject",
8                  "s3:GetObject",
9                  "s3:ListBucket",
10                 "s3:DeleteObject",
11                 "s3:DeleteObjectVersion"
12             ],
13             "Resource": [
14                 "arn:aws:s3:::ecommerce-ezequiel-2025",
15                 "arn:aws:s3:::ecommerce-ezequiel-2025/*"
16             ],
17         },
18         {
19             "Effect": "Allow",
20             "Action": [
21                 "logs:PutLogEvents",
22                 "logs:CreateLogStream",
23                 "logs:CreateLogGroup"
24             ],
25             "Resource": "*"
26         }
27     ]
28 }
```

[Visual](#) [JSON](#) [Actions](#)

Edit statement

Select a statement

Select an existing statement in the policy or add a new statement.

[+ Add new statement](#)

StepFunctionRole: Permisos para ejecutar Glue, leer S3 y publicar en SNS.

StepFunctions-StateMachineCommerce-role-3notg0wmj [Info](#)

Summary

Creation date
December 01, 2025, 10:17 (UTC-03:00)

Last activity
8 minutes ago

ARN
arn:aws:iam::371872301872:role/service-role/StepFunctions-StateMachineCommerce-role-3notg0wmj

Maximum session duration
1 hour

[Edit](#)

Permissions [Trust relationships](#) [Tags](#) [Last Accessed](#) [Revoke sessions](#)

Permissions policies (4) [Info](#)

You can attach up to 10 managed policies.

Policy name	Type	Attached entities
CloudWatchLogsDeliveryFullAccessPolicy-c8bd0240-2d4...	Customer managed	1
GlueJobRunManagementFullAccessPolicy-5d2a5e7d-a45...	Customer managed	1
StepFunctions	Customer inline	0
XRayAccessPolicy-6aa38a63-20ee-4284-ad8c-d6a16484...	Customer managed	1

[Filter by Type](#) [Search](#) [Simulate](#) [Remove](#) [Add permissions](#)

```

StepFunctions
1 [ {
2     "Version": "2012-10-17",
3     "Statement": [
4         {
5             "Effect": "Allow",
6             "Action": [
7                 "glue:StartCrawler",
8                 "glue:GetCrawler",
9                 "glue:GetJobRunMetrics",
10                "glue:StartJobRun",
11                "glue:GetJobRun"
12            ],
13            "Resource": "*"
14        }
15    ]
16 }
17 ]

```

QuickSight access role: Acceso restringido a Athena y al bucket de resultados.

aws-quicksight-service-role-v0 [Info](#)

Summary

Creation date
November 26, 2025, 11:19 (UTC-03:00)

Last activity
[Yesterday](#)

ARN [arn:aws:iam::371872301872:role/service-role/aws-quicksight-service-role-v0](#)

Maximum session duration
1 hour

[Delete](#) [Edit](#)

Permissions [Trust relationships](#) [Tags](#) [Last Accessed](#) [Revoke sessions](#)

Permissions policies (7) [Info](#)

You can attach up to 10 managed policies.

Policy name	Type	Attached entities
AWSQuickSightAthenaAccess	AWS managed	1
AWSQuickSightRAMPolicy	Customer managed	1
AWSQuickSightRDSPolicy	Customer managed	1
AWSQuickSightRedshiftPolicy	Customer managed	1
AWSQuickSights3Policy	Customer managed	1
test	Customer inline	0
test2	Customer inline	0

[Search](#) [Filter by Type](#) All types

[test](#) Customer inline 0 [Copy JSON](#) [Edit](#)

```

test
1 [ {
2     "Version": "2012-10-17",
3     "Statement": [
4         {
5             "Effect": "Allow",
6             "Action": [
7                 "athena:ListWorkGroups",
8                 "athena:GetWorkGroup"
9             ],
10            "Resource": "*"
11        }
12    ]
13 }
14 ]

```

[test2](#) Customer inline 0 [Copy JSON](#) [Edit](#)

```

test2
1 [ {
2     "Version": "2012-10-17",
3     "Statement": [
4         {
5             "Effect": "Allow",
6             "Action": [
7                 "s3:GetBucketLocation",
8                 "s3.GetObject",
9                 "s3>ListBucket",
10                "s3:PutObject"
11            ],
12            "Resource": [
13                "arn:aws:s3:::aws-athena-query-results-",
14                "arn:aws:s3:::aws-athena-query-results-/*"
15            ]
16        }
17 ]
18 }
19 ]

```

Security Groups

Además de los roles IAM, se configuraron **dos Security Groups principales** para controlar estrictamente el tráfico de red dentro de la VPC donde se ejecutan los recursos.

1. Security Group – RDS MySQL

Permite tráfico **únicamente** desde el Security Group de Glue (GlueConnections) y la IP local del desarrollador para pruebas. Puerto habilitado: **3306** (MySQL), no se expone la base de datos a internet.

Name	Security group rule ID	Type	Protocol	Port range	Source
sgr-042151871d6f7646	-	MySQL/Aurora	TCP	3306	sg-09c8fc0fa5181926 / rds-sg-2025
sgr-0792ec1e3a72fd3c5	IPv4	MySQL/Aurora	TCP	3306	181.165.242.144/32
sgr-0352a198dcba0d656	-	MySQL/Aurora	TCP	3306	sg-087183c84aa5aa153 / glue-sg

2. Security Group - Glue

Permite entrada desde el mismo Security Group de Glue (self). Es importante para que la conexión JDBC funcione, RDS y los recursos necesarios de Glue pueden comunicarse dentro de la red privada.

Name	Security group rule ID	Type	Protocol	Port range	Source
sgr-098cc44fc9aea0467	-	All traffic	All	All	sg-087183c84aa5aa153 / glue-sg

Utilización

Esta sección describe cómo un usuario final –por ejemplo, un analista de negocio o un equipo de BI que no interactúa directamente con AWS– puede operar toda la solución sin conocimientos técnicos ni acceso a la consola. El enfoque del diseño fue que el pipeline funcione como una "caja negra" totalmente automatizada.

El flujo completo para el cliente es el siguiente:

0. Desplegar la infraestructura con Terraform (una sola vez).

El desarrollador o el cliente ejecuta el comando terraform apply siguiendo las instrucciones del README. Esto crea automáticamente el bucket S3, los roles IAM, los Glue Jobs, el catálogo de datos, la Step Function, el EventBridge Rule y cualquier recurso necesario. Una vez desplegado, no es necesario repetir este paso.

1. Cargar un nuevo batch de datos mediante el script en Python.

Cada vez que haya datos nuevos -por ejemplo, los archivos del mes- el cliente simplemente ejecuta el script upload_files.py. Este script verifica extensiones, tamaños y nombres, y luego sube los archivos al bucket S3 dentro de raw/.

- o Si se sube un archivo incorrecto, el script lo bloquea.
- o Si se suben archivos válidos, se notifica por consola.

2. EventBridge detecta automáticamente la carga y activa Step Functions.

El cliente no tiene que hacer nada más. Un EventBridge Rule escucha cambios en raw/ y dispara el pipeline.

Step Functions ejecuta en orden:

- o Glue Job 1 (raw → processed)
- o Crawler processed
- o Glue Job 2 (processed → curated)
- o Crawler curated
- o Glue Job 3 (curated → RDS) (si está habilitado)
- o Notificación SNS o log final

3. Visualizar dashboards en QuickSight.

Una vez que Step Functions finaliza, las tablas curated están listas en Athena. QuickSight actualiza automáticamente sus datasets basados en Custom SQL y

el cliente puede ver dashboards con métricas como ingresos, categorías más vendidas, cohortes de clientes, funnel de conversión y más.

¿Qué NO necesita hacer el cliente?

- No debe abrir Glue ni ejecutar Jobs manualmente.
- No debe correr crawlers.
- No debe conectarse a Athena para consultas técnicas.
- No debe manipular IAM ni permisos en AWS.
- No debe manejar Step Functions directamente.

Toda la operación está abstraída detrás del script de carga y la interfaz visual de QuickSight. El cliente no necesita interactuar con la consola de AWS.

Estimación de costos y optimización

Estrategias aplicadas de Optimización de Costos

Durante el desarrollo del proyecto se aplicaron múltiples prácticas orientadas a minimizar el consumo y los costos de los servicios:

- **Uso de Parquet** para reducir el volumen escaneado por Athena (ahorro estimado de hasta 80% respecto de CSV).
- **Particionamiento por fecha** en S3 para reducir costos de escaneo y acelerar consultas.
- **Limitación de datos en desarrollo** mediante LIMIT 100, evitando análisis innecesario de grandes volúmenes.
- **Apagado manual de la instancia RDS** cuando no se utiliza fuera del horario de trabajo.
- **Optimización de Glue Jobs**, usando:
 - Mínimo de DPUs necesarias.
 - Timeouts configurados para evitar ejecuciones prolongadas o colgadas.

Estas prácticas permiten mantener el entorno operativo con costos muy bajos, aun utilizando servicios administrados como Glue y RDS.

Automatización del apagado de RDS

Para reducir costos del motor relacional se propone automatizar su encendido y apagado mediante **Amazon EventBridge** y **AWS Lambda**.

EventBridge ejecuta dos reglas (inicio y fin de la jornada) que invocan una función Lambda encargada de ejecutar StartDBInstance y StopDBInstance.

En este proyecto se considera un uso de **8 horas diarias**, de lunes a viernes, en lugar de mantener la instancia disponible 24/7.

Resumen de Costos del Entorno

A partir de las ejecuciones reales y asumiendo el uso optimizado de RDS (8 h/día de lunes a viernes), se obtiene la siguiente estimación para un uso general:

Glue Jobs

- Ejecuciones: 30 min + 19 min + 25 min con 2 DPU.
- **Costo total estimado: USD 1.09.**

Glue Crawlers

- 2 crawlers de 1.5 min.
- **Costo total estimado: USD 0.02.**

S3

- 133 MB almacenados.
- **Costo mensual estimado: USD 0.003.**

RDS

- 8 h/día × 5 días/semana × 4 semanas = ~160 h/mes.

- Instancia `db.t3.micro` \approx **0.018 USD/h.**
- Costo mensual estimado: **USD 2.88.**

Total estimado de todos los recursos	\approx USD 4.00
--------------------------------------	------------------------------

Este análisis demuestra que un entorno analítico moderno, totalmente automatizado y administrado con servicios serverless, puede operar con costos muy bajos si se aplican buenas prácticas desde el inicio. Uso de Parquet para reducir costo de Athena hasta 80%.

En el desarrollo del proyecto usé un presupuesto bastante más alto por distintas pruebas, testeos y errores que fueron surgiendo, y que explique más adelante. Esta estimación refleja el costo que realmente tendría la **versión final optimizada** del entorno si se usa en producción.

Análisis

Los dashboards creados en Amazon QuickSight permiten analizar de manera simple y actualizable el rendimiento del e-commerce. Como pueden filtrarse por mes, se adaptan fácilmente cuando se incorporan nuevos datos al Data Lake.

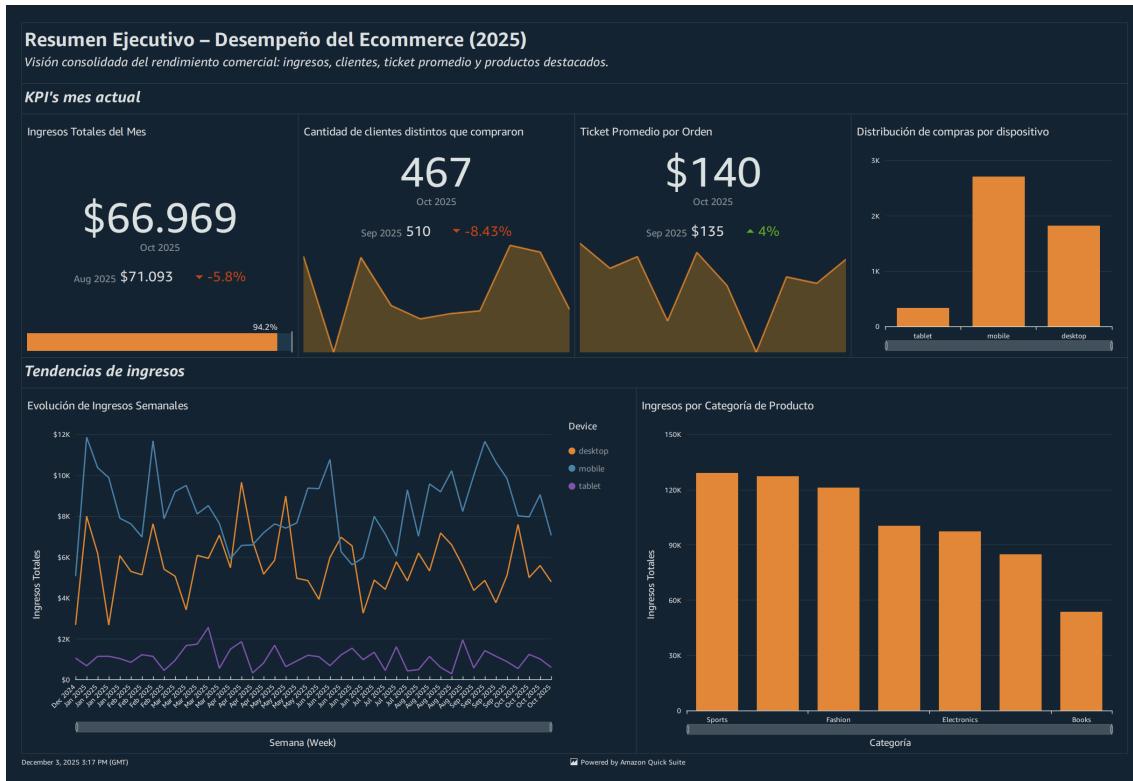
Para ordenar el análisis, todo surge de **tres preguntas de negocio simples**, que resumen lo que un equipo comercial realmente necesita saber.

1) ¿Cómo está funcionando el negocio este mes?

Los KPIs del mes muestran:

- Ingresos totales del mes.
- Comparación con el mes anterior.
- Cantidad de clientes únicos que compraron.
- Ticket promedio.

Dashboard 1

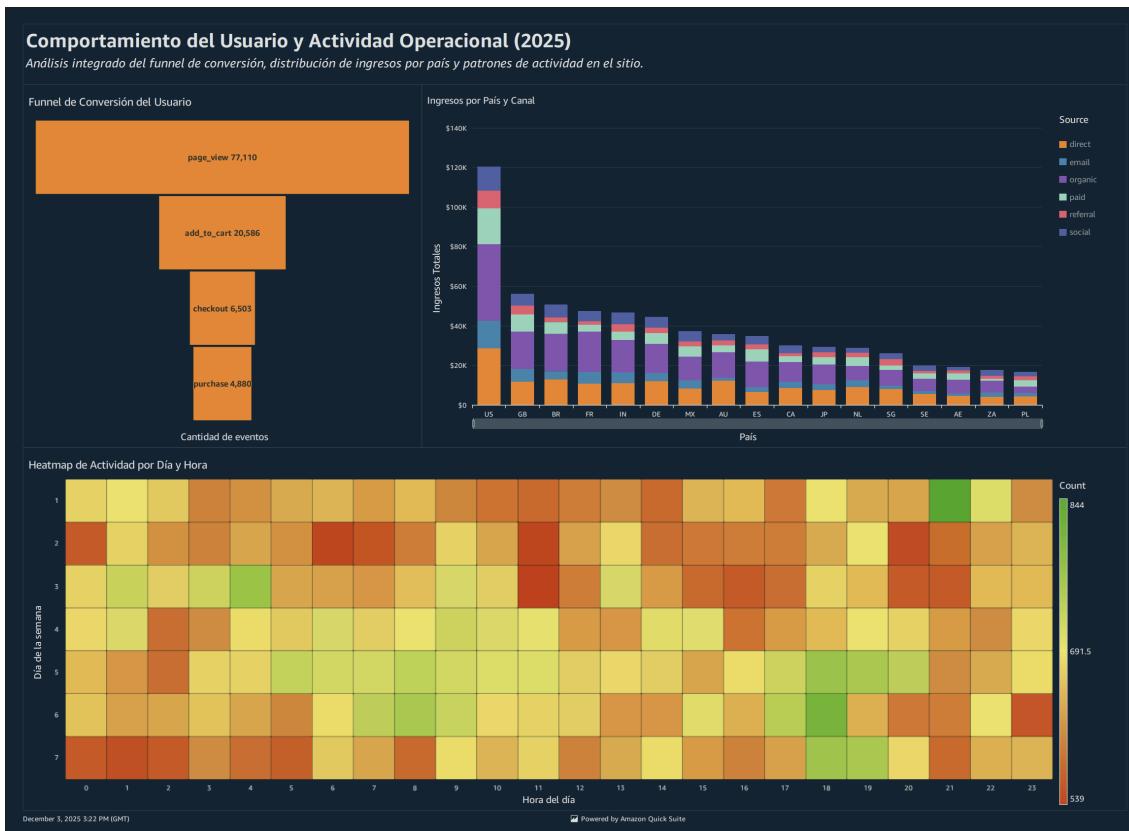


Insight: En el último mes los ingresos bajaron principalmente porque hubo menos clientes, no porque se haya gastado menos por orden. Esto permite enfocar acciones en retención y adquisición, más que en precios.

2) ¿En qué etapa del funnel estamos perdiendo más usuarios?

El funnel muestra el recorrido completo: page views → add to cart → checkout → purchase.

Dashboard 2



Insight: La mayor caída ocurre entre agregar al carrito y checkout. Esto indica que el proceso de compra podría optimizarse o acompañarse con incentivos (descuentos, recordatorios, recuperación de carritos, etc.).

3) ¿Cómo se comportan los usuarios según país, canal y horario?

Los dashboards permiten ver:

- Ingresos por país y por canal.
- Dispositivos con mayor participación.
- Horarios y días con más actividad.

Insight: Estados Unidos aporta la mayor parte de los ingresos y los horarios pico se concentran entre las 10-12 h y 19-21 h, lo que permite planificar campañas automáticas o recomendaciones personalizadas en esos momentos.

Estas tres preguntas resumen la lectura principal del negocio y permiten entender rápidamente qué está pasando, por qué está pasando y dónde conviene actuar para mejorar los resultados.

Estas tres preguntas desglosan la problemática central del negocio y permiten una lectura clara del desempeño comercial, operativo y del comportamiento del usuario. A partir de ellas se pueden definir acciones concretas en marketing, producto y estrategia comercial.

Desafíos en el desarrollo

Durante el desarrollo surgieron múltiples desafíos que impactaron tanto en la estabilidad del pipeline como en el costo total del proyecto. A continuación se detallan los principales problemas encontrados, su impacto y qué soluciones se implementan para resolverlos.

Fallas del Crawler por locations incorrectos

El Crawler detectaba schemas distintos a los esperados debido a errores de inferencia cuando encontraba archivos viejos o con formato inconsistente. Esto obligó a eliminar tablas del catálogo, limpiar particiones manualmente y volver a ejecutar el crawler con rutas corregidas y exclusiones configuradas.

Duplicados en *order_items*

Algunos registros se repetían por errores en la tabla original y por un join mal aplicado en etapas tempranas del pipeline. Se implementó una deduplicación explícita usando `groupBy(order_id, product_id)` y agregando métricas agregadas controladas.

Problemas de permisos con QuickSight

QuickSight no podía acceder a Athena debido a restricciones en el rol asignado y permisos insuficientes sobre el bucket de resultados. Se corrigió creando un rol dedicado con acceso restringido únicamente a los recursos necesarios.

Restricciones de SCP al intentar modificar IAM

La organización tenía una política de control (SCP) que bloqueaba cambios en IAM. Esto impidió modificar roles existentes y obligó a crear nuevos roles en lugar de editar los ya creados. El workaround consistió en solicitar al administrador permisos temporales y trabajar únicamente con recursos permitidos.

Pruebas con Glue que incrementaron el costo del proyecto

En las primeras etapas del desarrollo, se realizaron numerosas pruebas de conectividad con RDS. Varios Glue Jobs quedaban colgados esperando respuesta de la base de datos y seguían consumiendo DPUs sin procesar datos. Esto generó un gasto considerable innecesario.

Un desafío adicional fue **aprender a utilizar correctamente una conexión JDBC hacia RDS desde Glue**, lo cual requirió múltiples intentos, ajustes de parámetros de red, validaciones de credenciales y debugging extensivo en los logs de CloudWatch. Este proceso tomó tiempo y provocó varias ejecuciones de prueba que consumen recursos sin producir datos útiles.

Además, inicialmente los Jobs se configuraron con **10 workers**, lo cual era excesivo para el tamaño real del dataset. Esto aumentó drásticamente los costos hasta identificar que el procesamiento podía realizarse de manera eficiente con solo **2 DPUs**.

Errores al ejecutar el Job curated con archivos nuevos

Otro desafío importante surgió cuando se agregaron archivos nuevos y se re-ejecutó el Glue Job encargado de generar las tablas curated. En estas pruebas, el job quedaba corriendo durante mucho tiempo sin finalizar y sin generar resultados.

Ya con un presupuesto elevado debido a las pruebas previas, se optó por **no seguir ejecutando pruebas costosas hasta revisar completamente la lógica interna del job**, especialmente en las uniones con tablas grandes como events o sessions, que podrían estar generando un shuffle innecesario o un plan de ejecución poco eficiente.

Optimización final

La infraestructura final fue rediseñada para minimizar costos:

- Reducción de DPUs de 10 → 2 por job.
- Timeout agresivo para evitar que jobs colgados sigan consumiendo recursos.
- Conexiones más estables a RDS para evitar retries.
- Uso de particiones y Parquet para reducir el costo de Athena.
- Procesamiento incremental: la primera corrida es más larga y costosa, pero las corridas mensuales siguientes procesan únicamente los nuevos datos.

Estas mejoras permiten que el pipeline sea mucho más económico, predecible y adecuado para cargas reales de un negocio e-commerce.