

2026-02-11 — Phase 3 Information Architecture

Refactor

Alcance implementado:

- `SQL Explorer` se rehizo como explorador de base estilo DBeaver (mock).
- `Pipelines` mantiene submodos `Simple` y `Pro` con semántica invertida según feedback.
- Persistencia de submodo de pipelines en store.
- Compatibilidad de rutas existentes (`/editor`, `/pipelines`) mantenida.

Qué cambió

Top-level navigation

- Se actualizó el header principal para mostrar `SQL Explorer` en lugar de `SQL Editor`.
- Se mantuvo la clave interna `code` para no romper estado persistido ni rutas existentes.

SQL Explorer (DB explorer estilo DBeaver)

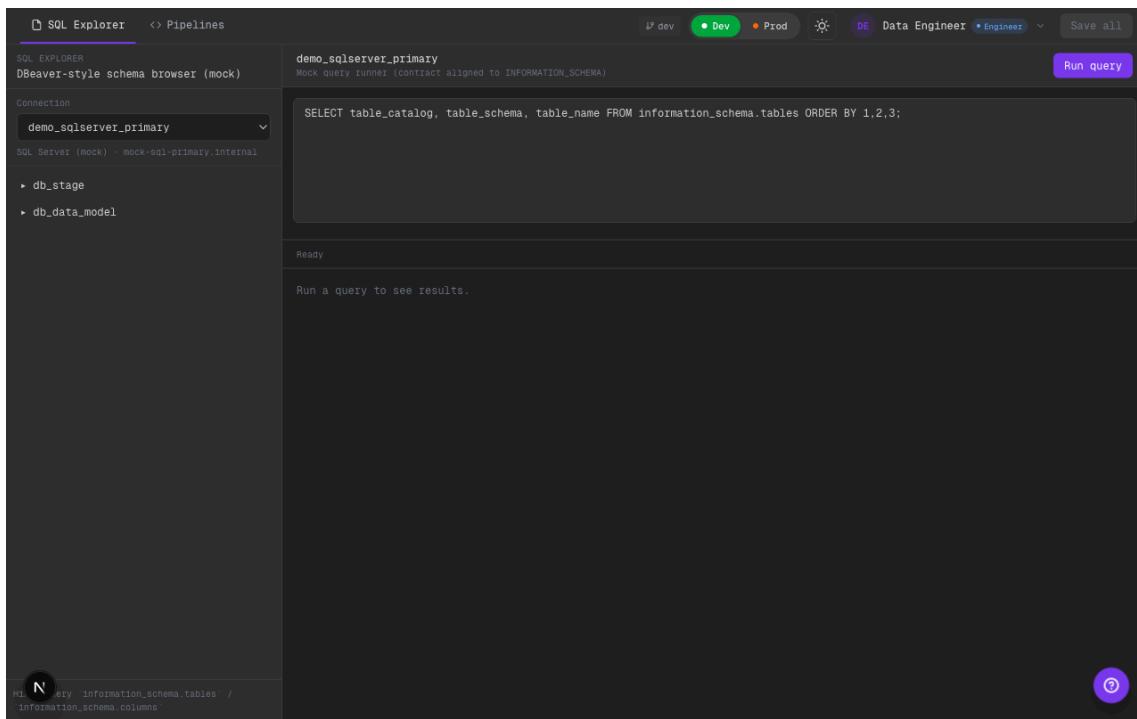
- `CodeView` ahora muestra:
 - conexión mock (`demo_sqldatabase_primary`)
 - árbol `database` → `schema` → `table`
 - query runner SQL
 - resultados tabulares
- El runner soporta contrato mock orientado a `information_schema.tables` / `information_schema.columns`.
- También permite `SELECT * FROM schema.table` con resultados mock de muestra.

Pipelines submodes (Simple / Pro)

- Se agregó `pipelineSubMode` en `workspace-store` con persistencia (`simple` por default).
- `PipelineView` ahora renderiza tabs de submodo:
 - `Simple` : board de pipelines (overview/detail con tareas y config DAG), equivalente al modo "pro" anterior.
 - `Pro` : editor de archivos SQL completo (file tree + edición + diff/git flow), equivalente al SQL editor anterior.
- `Quick Open` ahora abre archivo en `Pipelines` → `Pro` para mantener el flujo de edición.
- Desde el slide-out de tareas en `Simple`, el CTA ahora abre explícitamente `Pipelines` → `Pro` (`Open in Pro Editor`) para mantener continuidad de edición con tree/diff/git.

Evidencia visual

SQL Explorer (dark)



Qué mirar:

- El tab superior dice `SQL Explorer`.
- Se ve el árbol de DB (`db_stage`, schemas, tablas).
- Se ve query SQL ejecutable y resultados de `information_schema`.

Pipelines Simple (dark)

SQL Explorer < Pipelines

IP dev • Dev • Prod ☀ DE Data Engineer • Engineer Save all

Simple Pro Pipeline submode persisted

Search by name, integration, or tag... 5 of 5 DAGs by tag by integration

BEATS_INTEGRATION 1

Pipeline	Type	Schedule	Tasks	Status	Tags
dbo_AccountLogType	snapshot	Daily at 06:00 in 12h 37m	2	Draft	snapshot

CRM_INTEGRATION 3

Pipeline	Type	Schedule	Tasks	Status	Tags
dbo_AccountReference	snapshot	Every 3h in 44m	3	Draft	snapshot
gc_Game	snapshot	Every 3h in 42m	3	Draft	snapshot
tr_GameTransaction	incremental	Every hour in 37m	3	Draft	incremental

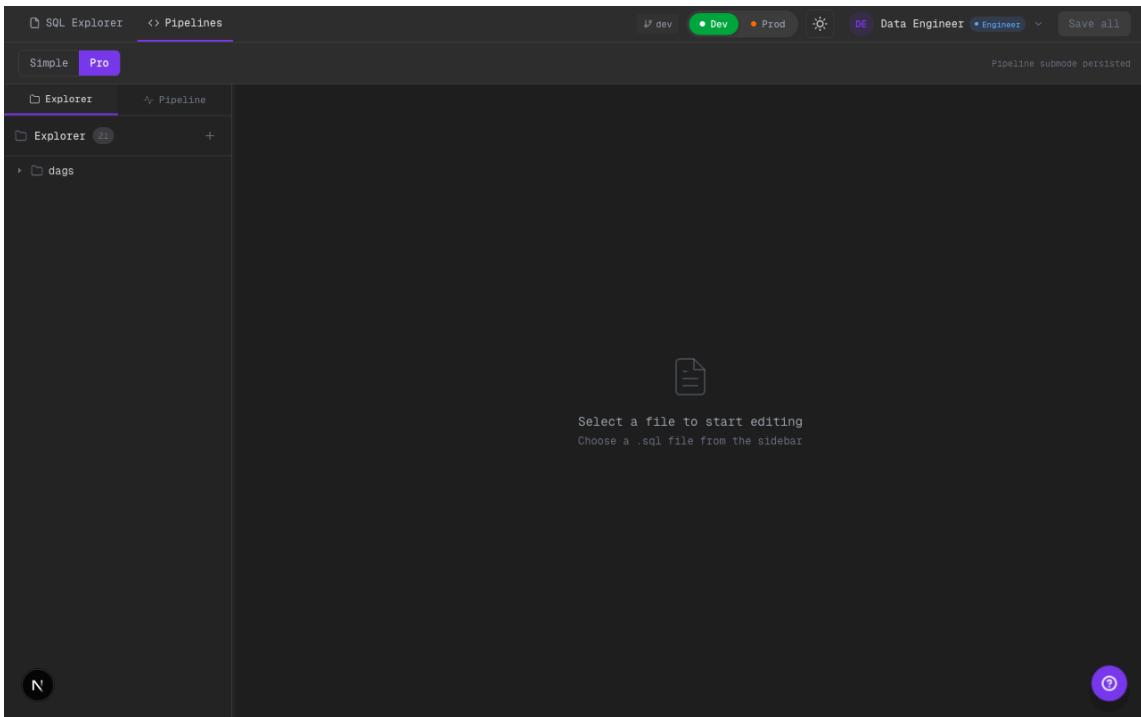
DATA_SOURCES 1

Pipeline	Type	Schedule	Tasks	Status	Tags
tr_DailyTransactionAmount	incremental	Every hour in 37m	3	Draft	incremental

Qué mirar:

- Tab de submodo con Simple activo.
- Se muestra el board de pipelines (search + grupos + filas de pipeline).
- El detalle/orden de tareas se mantiene disponible desde este submodo.

Pipelines Pro (dark)



Qué mirar:

- Tab de submodo con `Pro` activo.
- Se ve file tree SQL + editor.
- Se ve barra de acciones (`Dev/Prod`, `submit`, `diff/changes`), preservando flujo git scaffold.

Simple -> Pro handoff (dark)

The screenshot shows the Azure Data Studio interface. On the left, the 'SQL Explorer' tab is selected, showing a list of pipelines: 'Simple' (selected) and 'Pro'. Under 'Simple', there are several items: 'dbo_AccountReference', 'CRM_Integration', 'gc_Game', 'CRM_Integration', 'tr_GameTransaction', 'CRM_Integration', 'dbo_AccountLogType' (selected), 'BEATS_Integration', and 'tr_DailyTransactionAmount' under 'data_sources'. On the right, the 'Pipelines' tab is selected, showing a pipeline named 'BEATS_integration_dbo_AccountLogType' in draft mode. The pipeline has two tasks: '#1 data_model_task [TRANSFORM]' and '#2 delete_logs [DQA]'. A slide-out panel titled 'data_model_task.sql' displays the SQL code for the first task:

```

1 TRUNCATE TABLE db_data_model.Accounts_dbo_AccountLogType
2 ;
3
4 INSERT INTO db_data_model.Accounts_dbo_AccountLogType
5 SELECT
6     accountLogTypeID,
7     saga_hash,
8     saga_real_run_ts,
9     typeName,
10    rowCreated,
11    rowModified
12 FROM db_stage.Accounts_dbo_AccountLogType
13 WHERE saga_logical_run_ts = '{{ ts | convert_utc_to_et("US/Eastern") }}'
14 ORDER BY saga_real_run_ts ASC
15 ;

```

The status bar at the bottom indicates 'All changes saved + Diff'.

Qué mirar:

- En el slide-out se ve el botón `Open in Pro Editor`.
- El contexto parte desde `Simple` (detalle de pipeline con task list), no desde `SQL Explorer`.

The screenshot shows the Azure Data Studio interface. On the left, the 'SQL Explorer' tab is selected, showing a list of pipelines: 'Simple' (selected) and 'Pro'. Under 'Simple', there are several items: 'Explorer' (selected), 'Pipeline', 'dags' (selected), and '+'. On the right, the 'Pipelines' tab is selected, showing a pipeline named 'data_model_task.sql' in draft mode. The pipeline has one task: '#1 data_model_task [TRANSFORM]'. A slide-out panel titled 'data_model_task.sql' displays the SQL code for the task:

```

1 TRUNCATE TABLE db_data_model.Accounts_dbo_AccountLogType
2 ;
3
4 INSERT INTO db_data_model.Accounts_dbo_AccountLogType
5 SELECT
6     accountLogTypeID,
7     saga_hash,
8     saga_real_run_ts,
9     typeName,
10    rowCreated,
11    rowModified
12 FROM db_stage.Accounts_dbo_AccountLogType
13 WHERE saga_logical_run_ts = '{{ ts | convert_utc_to_et("US/Eastern") }}'
14 ORDER BY saga_real_run_ts ASC
15 ;

```

The status bar at the bottom indicates 'No changes to display'.

Qué mirar:

- Luego del click, queda activo `Pipelines` con submodo `Pro`.
- Se abre el editor completo con file tree y SQL activo para seguir edición/diff.

Límites scaffold

- `SQL Explorer` sigue sin ejecución real contra motor (mock explícito).
- No se agregó backend real de metadata/catalog; resultado SQL es simulado.

TODO hooks

- Conectar `SQL Explorer` a metadata real (`information_schema`) cuando exista backend de conexión.
- Evaluar multi-connection real (actualmente una mock).

Calidad

- `cd ui && npm run lint` (sin errores; warnings existentes no bloqueantes).
- `cd ui && npm run build` (ok).
- Validación visual en Chrome vía Playwright (capturas dark mode).