Dateneo®

Entornos de dbt Core

Una guía completa para configurar y gestionar entornos de desarrollo y producción separados utilizando targets y perfiles en dbt Core.

¿Qué son los entornos de dbt?

dbt facilita el mantenimiento de entornos de producción y desarrollo separados mediante el uso de **targets** dentro de un **perfil**. Un perfil típico tendrá un target llamado dev configurado como predeterminado.

Esto significa que mientras realizas cambios, tus objetos se construirán en tu target de *desarrollo* sin afectar las consultas de producción de tus usuarios finales. Una vez que confíes en tus cambios, puedes desplegar el código a *producción* ejecutando tu proyecto dbt con un target *prod*.

Flexibilidad de implementación

Esquemas separados

Diferentes esquemas dentro de una base de datos (recomendado)

Bases de datos separadas

Bases de datos completamente diferentes para cada entorno

Clusters separados

Clusters completamente diferentes para máximo aislamiento

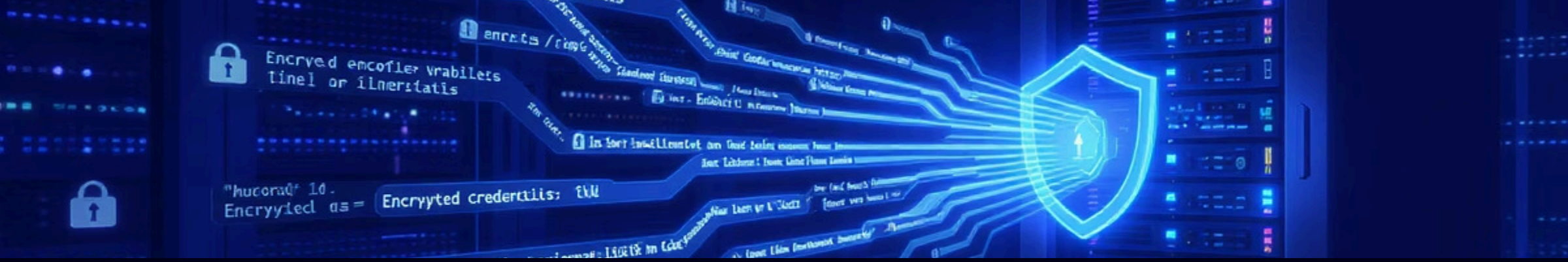
Recomendamos usar **diferentes esquemas dentro de una base de datos** para separar tus entornos. Esta es la solución más fácil de configurar y más rentable en un stack de datos moderno basado en la nube.

Configuración de perfiles de conexión

Cuando invocas dbt desde la línea de comandos, dbt analiza tu `dbt_project.yml` y obtiene el nombre del perfil que necesita para conectarse a tu almacén de datos.

```
# Ejemplo de archivo dbt_project.yml
name: 'jaffle_shop'
profile: 'jaffle_shop'
```

dbt luego verifica tu archivo `profiles.yml` para un perfil con el mismo nombre. Un perfil contiene todos los detalles requeridos para conectarse a tu almacén de datos.



Función env_var para credenciales seguras

La función `env_var` puede usarse para incorporar variables de entorno del sistema en tu proyecto dbt. Está disponible en cualquier lugar donde dbt procese código Jinja.

```
profile:
  target: prod
  outputs:
    prod:
      type: postgres
      host: 127.0.0.1
      user: "{{ env_var('DBT_USER') }}"
      password: "{{ env_var('DBT_PASSWORD') }}"
```

Esto evita poner credenciales directamente en el servidor y mejora la seguridad.

Configuración de targets y esquemas

El **target schema** representa el esquema predeterminado en el que dbt construirá objetos, y a menudo se usa como diferenciador entre entornos separados dentro de un Data Warehouse.

Esquemas de desarrollo

Patrón recomendado: `dbt_<username>`

- Permite que múltiples usuarios desarrollen en dbt
- Evita que los usuarios sobrescriban el trabajo de otros
- Asegura que los permisos y la propiedad de objetos sean consistentes en todo el esquema

Esquemas de producción

- Claro para los usuarios finales
- Indica que está listo para análisis
- No requiere creación previa, dbt lo creará si no existe

Mientras que el target schema representa el esquema predeterminado que dbt utilizará, también se puede dividir los modelos en esquemas separados usando esquemas personalizados.

Mejores prácticas y comandos

1 **Credenciales individuales**

Cada usuario de dbt debe tener sus propias credenciales de base de datos, incluyendo un usuario separado para ejecuciones de producción.

2 **Validar conexión**

Usa `dbt debug` para probar tu conexión desde dentro de un proyecto dbt.

3 **Sobrescribir targets**

Usa `dbt run --profile mi-perfil --target dev` para especificar perfil y target diferentes.