FraudLens - Documentation CI/CD

Version: 1.0.0 | **Date:** Février 2026 | **Auteur:** FraudLens Team

---

Table des Matières

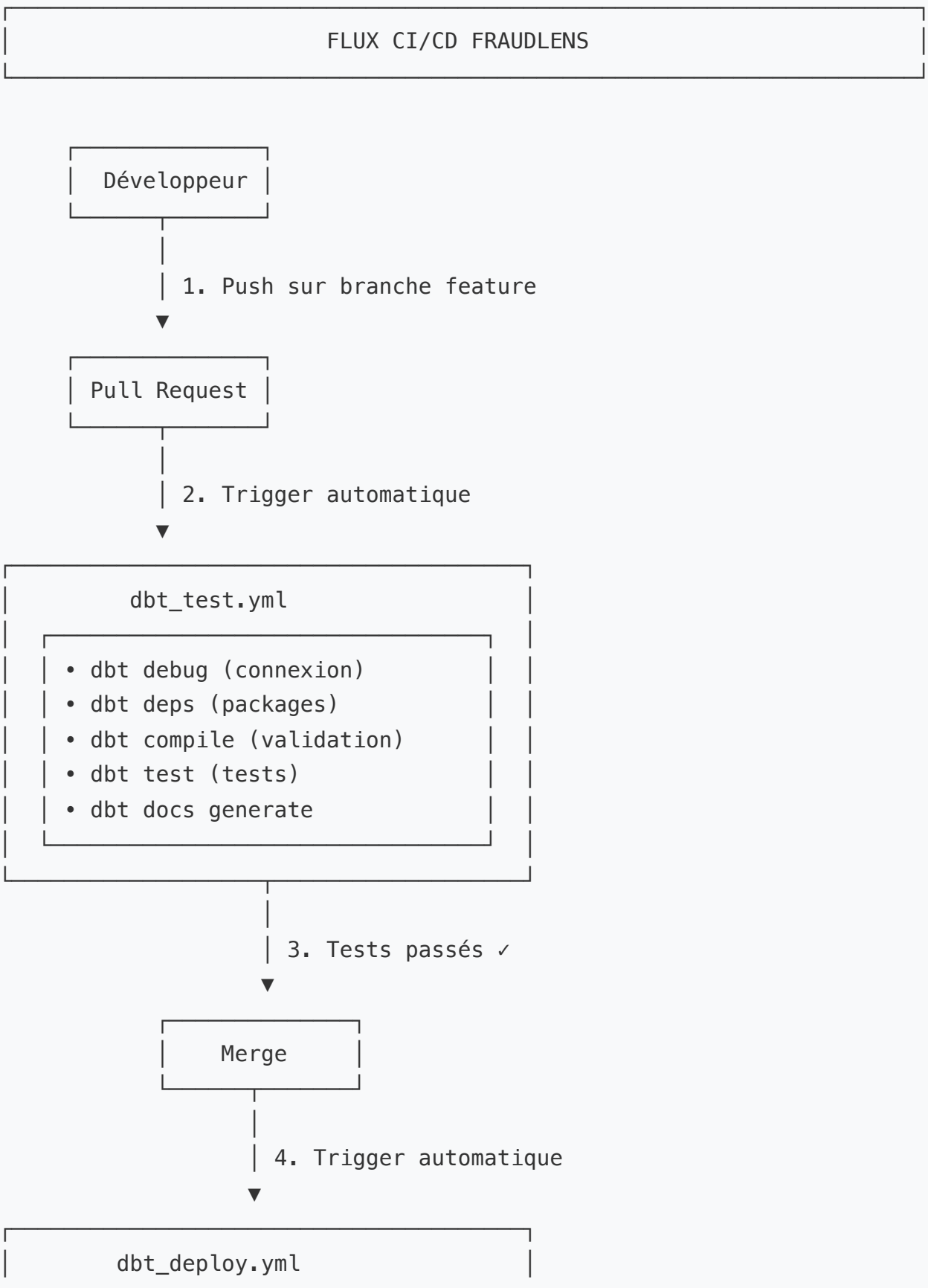
1. Vue d'Ensemble

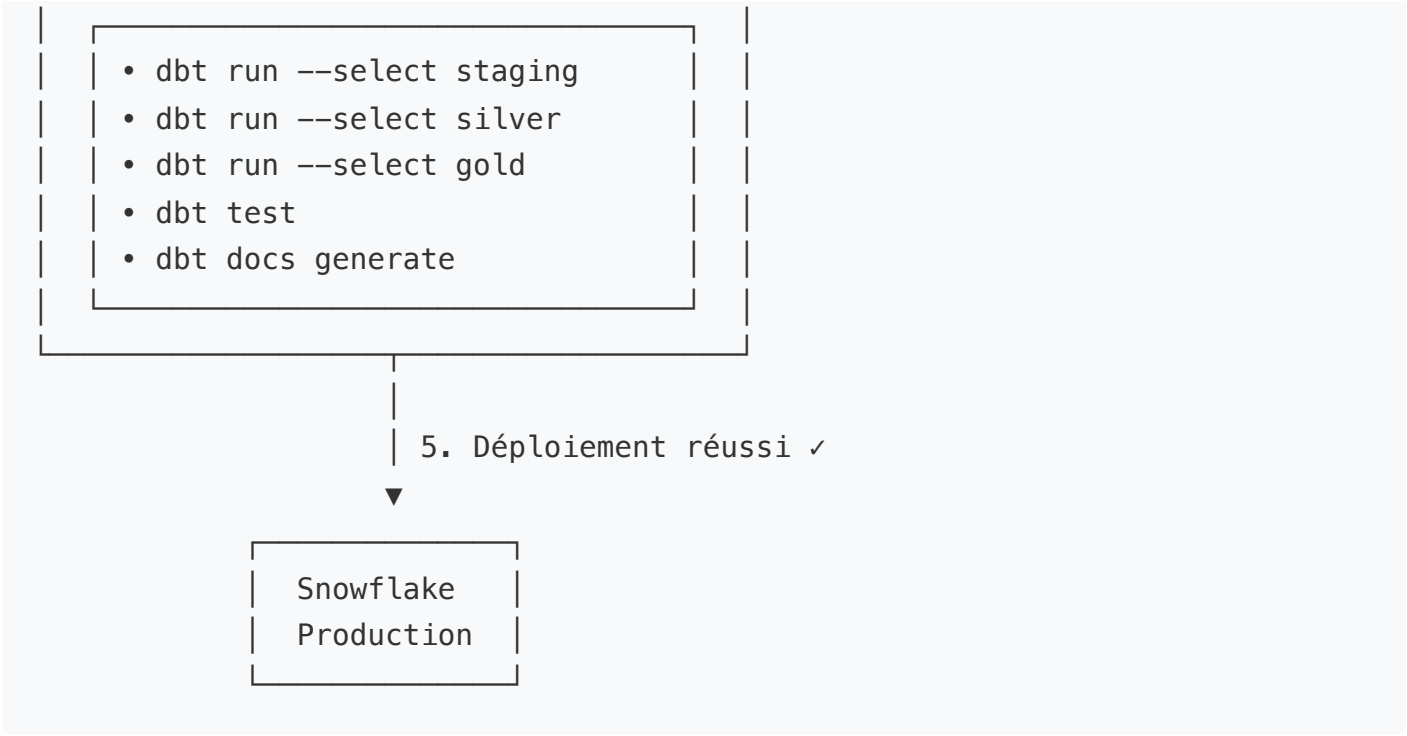
FraudLens utilise **GitHub Actions** pour automatiser les tests et le déploiement des modèles dbt. Cette documentation détaille l'architecture CI/CD mise en place pour garantir la qualité du code et la fiabilité des déploiements.

Objectifs du CI/CD

- **Qualité** : Exécuter les tests dbt sur chaque Pull Request
- **Automatisation** : Déployer automatiquement les modèles après merge
- **Sécurité** : Utiliser l'authentification par clé privée RSA
- **Documentation** : Générer la documentation dbt automatiquement

2. Architecture CI/CD





3. Workflows GitHub Actions

3.1 dbt_test.yml - Tests sur Pull Request

Déclencheur: Pull Request vers `main` avec modifications dans `dbt/**`

Fichier: `.github/workflows/dbt_test.yml`

Étape	Action	Description
1	Checkout	Clone le repository
2	Setup Python	Installe Python 3.11
3	Install dbt	pip install dbt-snowflake==1.7.0
4	Setup Private Key	Crée le fichier .p8 depuis le secret
5	Create profiles.yml	Configure la connexion Snowflake
6	dbt debug	Vérifie la connexion
7	dbt deps	Installe les packages dbt
8	dbt compile	Compile les modèles
9	dbt test	Exécute les tests
10	dbt docs generate	Génère la documentation

```yaml
name: dbt Tests

on:
  pull_request:
    branches: [main]
    paths:
      - 'dbt/**'
      - '.github/workflows/dbt_test.yml'

env:
  SNOWFLAKE_ACCOUNT: ${{ secrets.SNOWFLAKE_ACCOUNT }}
  SNOWFLAKE_USER: ${{ secrets.SNOWFLAKE_USER }}
  SNOWFLAKE_WAREHOUSE: ${{ secrets.SNOWFLAKE_WAREHOUSE }}
  SNOWFLAKE_DATABASE: ${{ secrets.SNOWFLAKE_DATABASE }}

jobs:
  dbt-test:
    name: Run dbt Tests
    runs-on: ubuntu-latest
    steps:
      - name: Checkout code
        uses: actions/checkout@v4

      - name: Setup Snowflake private key
        run: |
          mkdir -p $HOME/.snowflake
          echo "${{ secrets.SNOWFLAKE_PRIVATE_KEY }}" > $HOME/.snowflake/rsa_key.p8
          chmod 600 $HOME/.snowflake/rsa_key.p8

      - name: Run dbt tests
        working-directory: ./dbt
        run: dbt test --select staging silver gold --profiles-dir $HOME/.dbt
```

3.2 dbt_deploy.yml - Déploiement sur main

Déclencheur: Push sur `main` avec modifications dans `dbt/**`

Fichier: `.github/workflows/dbt_deploy.yml`

Étape	Commande	Description
Run Staging	`dbt run --select staging`	Exécute les vues staging
Run Silver	`dbt run --select silver`	Exécute les tables silver
Run Gold	`dbt run --select gold`	Exécute les tables gold
Tests	`dbt test`	Vérifie l'intégrité des données
Docs	`dbt docs generate`	Met à jour la documentation

3.3 lint.yml - Linting SQL (Optionnel)

Déclencheur: Pull Request avec modifications de fichiers SQL

Outil: SQLFluff pour le linting SQL avec le dialecte Snowflake

4. Secrets GitHub

Les secrets doivent être configurés dans **Settings → Secrets and variables → Actions**

Secret	Description	Exemple
`SNOWFLAKE_ACCOUNT`	Identifiant du compte Snowflake	rdsnbdu-gbc86569
`SNOWFLAKE_USER`	Nom d'utilisateur Snowflake	FISHER
`SNOWFLAKE_WAREHOUSE`	Nom du warehouse	FRAUDLENS_WH
`SNOWFLAKE_DATABASE`	Nom de la base de données	FRAUDLENS_DB
`SNOWFLAKE_PRIVATE_KEY`	Contenu de la clé privée RSA	-----BEGIN PRIVATE KEY-----...

> ⚠️ **Important:** Le secret `SNOWFLAKE_PRIVATE_KEY` doit contenir le contenu complet du fichier `.p8`, incluant les lignes BEGIN et END.

5. Authentification Snowflake

Méthode: Clé Privée RSA

L'authentification utilise une **clé privée RSA** au lieu d'un mot de passe pour une sécurité renforcée.

Méthode	Sécurité	Recommandation
Mot de passe	⭐⭐	Non recommandé pour CI/CD
Clé privée (RSA)	⭐⭐⭐	**Recommandé**
Key Pair + MFA	⭐⭐⭐⭐	Production critique

Configuration dans le workflow

```
- name: Setup Snowflake private key
  run: |
    mkdir -p $HOME/.snowflake
    echo "${{ secrets.SNOWFLAKE_PRIVATE_KEY }}" > $HOME/.snowflake/rsa_key.p8
    chmod 600 $HOME/.snowflake/rsa_key.p8

- name: Create profiles.yml for CI
  run: |
    mkdir -p $HOME/.dbt
    cat > $HOME/.dbt/profiles.yml << EOF
    fraudlens:
      target: prod
      outputs:
        prod:
          type: snowflake
          account: ${{ secrets.SNOWFLAKE_ACCOUNT }}
          user: ${{ secrets.SNOWFLAKE_USER }}
          private_key_path: $HOME/.snowflake/rsa_key.p8
          role: ACCOUNTADMIN
          database: ${{ secrets.SNOWFLAKE_DATABASE }}
          warehouse: ${{ secrets.SNOWFLAKE_WAREHOUSE }}
          schema: STAGING
          threads: 4
    EOF
```

ℹ️ **Note:** Le chemin utilise `$HOME` et non `~` car le tilde n'est pas expandé dans les fichiers YAML.

6. Flux de Travail Développeur

Étape 1: Créer une branche

```
git checkout -b feature/ma-nouvelle-feature
```

Étape 2: Faire des modifications

Modifier les fichiers dbt dans `dbt/models/`

Étape 3: Commit et Push

```
git add .
git commit -m "Add new feature"
git push -u origin feature/ma-nouvelle-feature
```

Étape 4: Créer une Pull Request

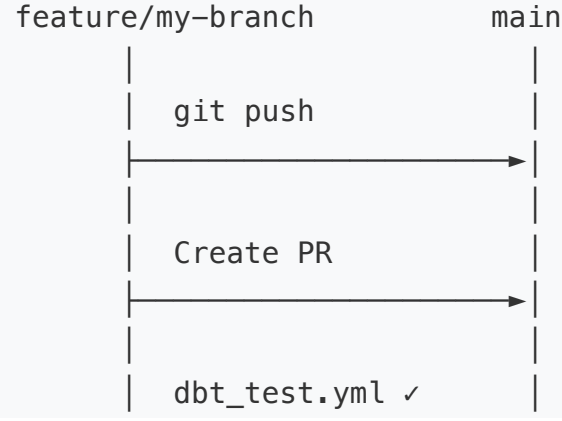
```
gh pr create --title "Ma nouvelle feature" --body "Description..."
```

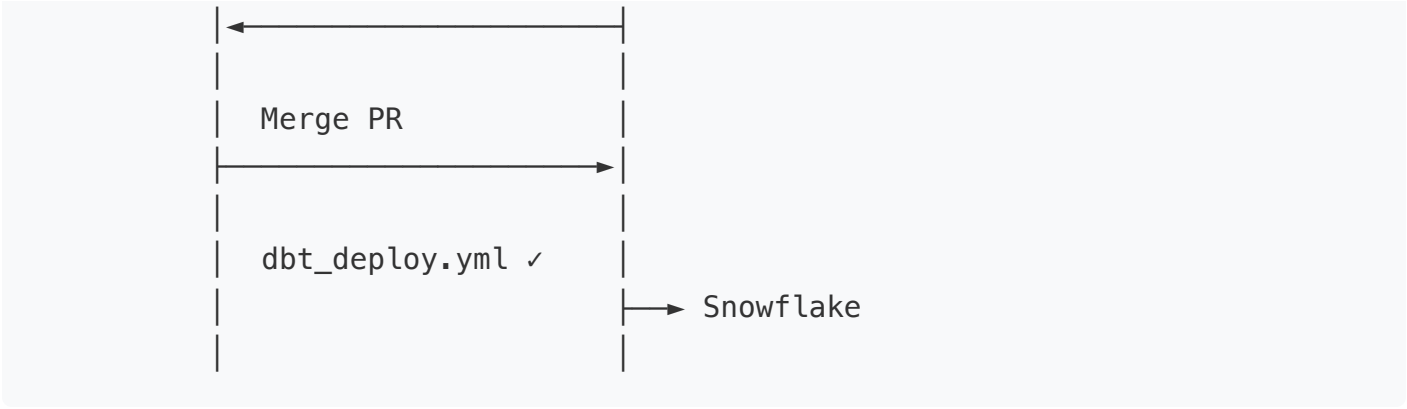
Étape 5: Vérifier les tests CI

Le workflow `dbt_test.yml` se lance automatiquement. Vérifier que tous les tests passent dans l'onglet "Checks" de la PR.

Étape 6: Merge la PR

Une fois les tests passés, merger la PR. Le workflow `dbt_deploy.yml` déploie automatiquement en production.





7. Dépannage

Erreur: "No such file or directory: rsa_key.p8"

Cause: Le chemin `~` n'est pas expandé dans YAML.

Solution: Utiliser `$HOME` au lieu de `~`

```
# Incorrect
private_key_path: ~/.snowflake/rsa_key.p8

# Correct
private_key_path: $HOME/.snowflake/rsa_key.p8
```

Erreur: "profiles.yml not found"

Cause: dbt cherche le profiles.yml dans le mauvais répertoire.

Solution: Ajouter `--profiles-dir $HOME/.dbt` à toutes les commandes dbt.

Erreur: "Connection test failed"

Causes possibles:

- Secrets mal configurés dans GitHub
- Clé privée incorrecte ou incomplète
- Compte Snowflake invalide
- Warehouse suspendu

Vérification:

1. Vérifier les secrets dans GitHub Settings
2. S'assurer que la clé privée est complète (avec BEGIN/END)

3. Tester la connexion localement avec la même clé

Le workflow ne se lance pas

Cause: Les fichiers modifiés ne correspondent pas au filtre `paths:`

Solution: Vérifier que les modifications sont dans `dbt/**`

8. Bonnes Pratiques

Tests avant merge

Toujours attendre que les tests CI passent avant de merger une PR.

Revue de code

Faire reviewer les changements dbt par un pair avant merge.

Commits atomiques

Un commit = une modification logique.

Messages de commit clairs

```
feat: Add new fraud detection model
fix: Correct duplicate handling in provider.sql
docs: Update CI/CD documentation
refactor: Simplify payment aggregation logic
```

Ne jamais commit de secrets

Les fichiers sensibles doivent être dans `.gitignore` :

- `dashboard/.streamlit/secrets.toml`
- `snowflake/keys/*.p8`
- `.env`

> ✓ **Fichiers protégés:** Le fichier `profiles.yml` du repo utilise des variables d'environnement (`env_var`) et non des credentials en clair.

9. Monitoring

Voir les runs GitHub Actions

```
# Lister les derniers runs
gh run list --limit 10

# Voir les détails d'un run
gh run view <run-id>

# Voir les logs complets
gh run view <run-id> --log

# Voir uniquement les logs d'échec
gh run view <run-id> --log-failed
```

Statuts des workflows

Statut	Signification	Action
✅ success	Tous les tests passent	PR peut être mergée
❌ failure	Un ou plusieurs tests échouent	Corriger et re-push
🟡 in_progress	Workflow en cours d'exécution	Attendre la fin
⚪ skipped	Aucun fichier dbt modifié	Normal si pas de changement dbt

---

Ressources

- Documentation dbt
- GitHub Actions
- Snowflake Key Pair Authentication
- Repository FraudLens

Document généré automatiquement - FraudLens v1.0.1 - Février 2026