Documentation Technique

Projet Final : Intégration de Données Ontario 511 via SQL Server

A- Description complète de l'architecture (sans code Python)

Le système est entièrement conçu et implémenté dans SQL Server, sans recours à des scripts externes (tels que Python). L'architecture repose sur une intégration automatisée des données provenant de fichiers CSV générés par un processus externe (non inclus), supposés être déposés régulièrement dans un répertoire cible sur le serveur.

B- Composants clés de l'architecture :

1. Fichiers CSV sources : Quatre fichiers sont attendus dans un dossier partagé :evenements.csv, constructions.csv, cameras.csv, roadcondition.csv.
2. Ces fichiers contiennent les données routières mises à jour par l'API Ontario 511.
3. SQL Server Agent Job : Un job planifié exécute toutes les 2 heures un ensemble de scripts T-SQL. Ce job :
4. Charge les données CSV dans des tables temporaires via BULK INSERT.
5. Appelle des procédures stockées pour valider, transférer et nettoyer les données.
6. Journalise chaque étape dans la table LogsExecution.

C- Processus d'importation :

1. Nettoyage des tables temporaires.
2. Chargement des CSV via BULK INSERT.
3. Appel de la procédure ImporterDonneesDepuisTemp pour chaque type de données.
4. Archivage des événements obsolètes.
5. Mise à jour des indicateurs KPI.
6. Journalisation du résultat.
7. Gestion des erreurs : Chaque étape est encapsulée dans des blocs TRY...CATCH. En cas d'erreur, un message est enregistré dans LogsExecution avec le statut Échec.
8. Analyse et rapports : Des procédures de génération de rapports permettent d'extraire des insights métier (rue la plus surveillée, condition dominante par région, etc.).
9. Audit et historisation : Les modifications des événements sont tracées via un trigger, et les anciens événements sont archivés automatiquement.

D- Schéma relationnel complet

Tables principales

Evènements	Événements routiers (accidents, fermetures, etc.)
Constructions	Travaux routiers en cours ou planifiés
Cameras	Emplacements et détails des caméras de surveillance
RoadCondition	Conditions météorologiques et routières en temps réel
HistoriqueEvenements	Archive des événements supprimés ou modifiés
Statistiques	Indicateurs KPI calculés périodiquement
LogsExecution	Journal des exécutions du job d'importation

Relations et clés

1. Clés primaires : Chaque table a une clé primaire (ID auto-incrémenté ou clé métier).
2. Clés étrangères : Aucune référence entre les tables métiers (modèle dénormalisé pour performance), sauf :

   HistoriqueEvenements.EvenementID → Evenements.ID (pour traçabilité).
3. Contraintes :

   NOT NULL sur champs obligatoires (ex: Titre, DateDebut, Region).

   CHECK sur champs numériques (ex: Latitude entre -90 et 90).

   DEFAULT GETDATE() pour DateCreation.

E- Instructions de configuration du BULK INSERT (répertoire cible)

Prérequis :

1- Le serveur SQL Server doit avoir accès en lecture/écriture au dossier contenant les CSV.
2- Le compte de service SQL Server doit disposer des permissions NTFS sur le dossier.

Répertoire cible :

C:\Ontario511_Data\Imports\

Ce chemin peut être modifié dans les scripts BULK INSERT selon l'environnement.

Format des fichiers CSV :

1- Encodage : UTF-8
2- Séparateur : virgule (,), en-tête présent
3- Valeurs manquantes : chaîne vide ou NULL
4- détail des procédures stockées, fonctions et triggers

F-Procédures stockées

ImporterDonneesDepuisTemp(@Type)	Transfère les données depuis la table temporaire la table cible selon le type (Evenements , Constructions , etc.). Valide les doublons via MERGE ou NOT EXISTS .
ArchiverEvenementsObsoletes()	Déplace les événements dont DateFin < GETDATE() vers HistoriqueEvenements , puis les supprime de Evenements .
CalculerIndicateursKPI()	Calcule les indicateurs clés et met à jour la table Statistiques (voir sectio
RechercherEvenementsFiltres(@Type, @Date, @Region, @MotCle)	Recherche dynamique dans Evenements avec clauses conditionnelles (LIKE , BETWEEN , etc.).
GenererRapportRuePlusCamera()	Retourne la rue (RoadwayName) avec le plus grand nombre de caméras actives.

GenererRapportRuePlusConstruction()	Retourne la rue avec le plus de chantiers actifs (basé sur Status = 'Active').
GenererRapportConditionParRegion()	Pour chaque région, retourne la condition routière la plus fréquente (ex: "Snow", "Wet").
GenererRapportOrganisationTop()	Retourne l'organisation ayant signalé le plus d'événements dans Evenements .

G-Triggers (3 minimum)

TR_HistoriqueEvenements	AFTER UPDATE, DELETE sur Evenements	Insère une copie de l'événement modifié/supprimé dans HistoriqueEvenements avec Action ('MODIFIE', 'SUPPRIME') et DateAction .
TR_UpdateTimestamp	AFTER UPDATE sur Evenements , Constructions , RoadCondition	Met à jour automatiquement le champ DateModification à GETDATE() .
TR_MiseAJourStatistiques	AFTER INSERT sur Evenements , Cameras , Constructions	Déclenche CalculerIndicateursKPI() pour maintenir les statistiques à jour.

H-Fonctions SQL (2 minimum)

fn_NombreEvenementsParJour(@Date DATE)	Fonction scalaire	Retourne le nombre total d'événements actifs à une date donnée (chevauchant @Date).
fn_ConditionDominanteParRue(@RoadwayName NVARCHAR(255))	Fonction scalaire	Retourne la condition météo la plus fréquente pour une

		rue donnée (ex: "Dry", "Icy").

I-. Explication des indicateurs calculés dans Statistiques

a- table `Statistiques` stocke des indicateurs
b- KPI mis à jour quotidiennement (ou à chaque import). Chaque indicateur est
c- identifié par un `NomIndicateur` et une `Valeur` (numérique ou texte).
d- Liste des indicateurs calculés par `CalculerIndicateursKPI()` :

`NombreTotalEvenementsActifs`	`SELECT COUNT(*)` `FROM Evenements WHERE DateFin >=` `GETDATE()`
`NombreMoyenEvenementsParJour`	Moyenne du nombre d'événements actifs par jour sur les 30 derniers jours (via `fn_NombreEvenementsParJour`).
`DureeMoyenneEvenementsHeures`	`AVG(DATEDIFF(HOUR,` `DateDebut, DateFin))` sur tous les événements terminés.
`NombreTotalCameras`	`SELECT COUNT(*)` `FROM Cameras`
`NombreMoyenConstructionsActivesParJour`	Moyenne du nombre de constructions actives par jour sur les 30 derniers jours.

Donc Fréquence de mise à jour : Appelée automatiquement par le job toutes les 2 heures, ou déclenchée par `TR_MiseAJourStatistiques`.

J. Exemple d’utilisation des procédures de rapport

1. Récupérer la rue avec le plus de caméras

```
EXEC GenererRapportRuePlusCamera;
-- Résultat : "Highway 401" (avec 42 caméras)
```

2. Trouver la rue avec le plus de constructions actives

```
EXEC GenererRapportRuePlusConstruction;
-- Résultat : "Yonge Street" (avec 8 chantiers actifs)
```

3- Obtenir la condition dominante par région

```
EXEC GenererRapportConditionParRegion;
-- Résultat :
-- Région: Toronto, Condition: Wet
-- Région: Ottawa, Condition: Snow
-- Région: Hamilton, Condition: Dry
```

4. Identifier l’organisation la plus active

```
EXEC GenererRapportOrganisationTop;
-- Résultat : "Ministère des Transports" (147 événements signalés)
```

5. Rechercher des événements avec filtres

```
EXEC RechercherEvenementsFiltres
    @Type = 'Accident',
    @Date = '2025-04-05',
    @Region = 'Toronto',
    @MotCle = 'collision';
-- Retourne tous les accidents à Toronto le 5 avril 2025 contenant "collision"
```

Conclusion:

Ce projet a permis de concevoir et de mettre en œuvre un système complet d'intégration, de gestion et d'analyse des données routières de l'Ontario, en exploitant exclusivement les fonctionnalités avancées de SQL Server. Sans recourir à aucun script externe (comme Python), nous avons développé une solution entièrement automatisée, robuste et évolutive, conforme aux exigences fonctionnelles et techniques.

Grâce à l'utilisation de BULK INSERT et du SQL Server Agent, le système assure un chargement régulier et fiable des données CSV dans des tables temporaires, suivi d'un transfert contrôlé vers les tables principales. L'architecture mise en place garantit l'intégrité des données grâce à des contraintes rigoureuses, des clés appropriées et une journalisation complète via la table LogsExecution.

La couche de programmation avancée — composée de procédures stockées, fonctions, triggers et indicateurs KPI — permet non seulement de gérer les données, mais aussi de générer des rapports analytiques pertinents pour les décideurs. Des indicateurs tels que le nombre d'événements actifs, la durée moyenne des incidents ou la condition routière dominante par région offrent une vision claire et opérationnelle de l'état du réseau routier.

Enfin, la documentation technique détaillée et les exemples d'utilisation des rapports illustrent la maintenabilité et la clarté du système.