


M1 Informatique AIGLE

HMIN122M

Mini-Projet : entrepôts de données

Rapport

Bachar Rima
Joseph Saba
Tasnim Shaqura Muhammad
Jérémy Bourgin

8 novembre 2018

Table des matières

Chapitre 1

Introduction

1.1 Problématiques

Dans le cadre du mini-projet du module **HMIN122M**, nous avons décidé de modéliser un entrepôt de données pour le réseau de transport publique de Montpellier, *tam-voyages*. Pour ce faire, nous avons proposé des *data marts* formant le *data warehouse* et permettant de réaliser des requêtes analytiques sur un ensemble important de données. Cette modélisation permettra ainsi de mettre en œuvre un outil d'analyse permettant de bien répondre aux problématiques suivantes :

1. Comment *tam-voyages* pourront-ils profiter de la fréquentation de leurs véhicules en se basant sur la circulation du réseau [1] pour améliorer la qualité de leur service (e.g. *nombre de véhicules à envoyer sur une ligne pendant une certaine période de la journée*) ?
2. Comment *tam-voyages* pourront-ils suivre l'évolution et la maintenance de leurs matériaux de manière à réduire les dépenses qui y sont associées ?

Ces problématiques seront ainsi adressées en analyzant les actions et opérations effectuées par *tam-voyages*, notamment en choisissant celles qui paraissent les plus pertinentes et les plus importantes en termes de données intégrées et flexibilité des critères d'analyse.

1.2 Actions et opérations

Les actions/opérations effectuées par *tam-voyages* considérées :
— Les voyages.

1. en particulier en examinant les lignes de tramway et les bus

— La maintenance de véhicules.
— Les ventes de tickets et les abonnements.
— Les amendes.

1.3 Exemples de requêtes analytiques possibles

1. exemples de requêtes analytiques pour l'action « voyages » :
   — `le nombre de voyageurs par bus, utilisant des tickets pour le mois de juillet.`
   — `le nombre moyen de voyageurs abonnés par ligne pour chaque voyage pour les deux derniers mois.`
   — `l'arrêt le plus fréquenté par toutes les lignes de circulation.`
2. exemples de requêtes analytiques pour l'action « maintenance » :
   — `le nombre de bus maintenus pour le mois de septembre 2018.`
   — `les X employés les plus expérimentés convoqués pour la maintenance des bus pour le dernier mois.`
   — `les X premières véhicules nécessitant le plus de maintenance pour les 6 dernier mois.`
3. exemples de requêtes analytiques pour l'action « ventes » :
   — `le nombre d'abonnés ayant plus que 26 ans pour le mois d'août 2018.`
   — `le nombre d'abonnés par date de naissance pour l'année 2018.`
   — `les types d'abonnement les plus fréquents pour l'année 2018.`
4. exemples de requêtes analytiques pour l'action « amendes » :
   — `les lignes qui ont générées le plus d'amendes pour les deux derniers mois.`
   — `les lignes les plus contrôllées de la semaine dernière.`
   — `le nombre des abonnés qui ont reçu des amendes par ligne, l'avant-midi.`
   — `la somme total d'amendes rapportée par type de voyageur par ligne pour le dernier mois.`

Chapitre 2

Modélisation

2.1 Choix des actions à modéliser

Les actions considérées, par ordre d'importance :

1. « voyages ».
2. « ventes ».
3. « maintenance ».
4. « amendes ».

Les actions les plus pertinentes à analyser vis-à-vis les problématiques avancées sont « voyages »et « maintenance »qu'on traitera de la manière suivante :

voyages : modèle en étoile détaillé.

maintenance : modèle en étoile *moins* détaillé, en particulier le modèle intitulé "*periodic snapshot*".

2.2 Modélisation des voyages

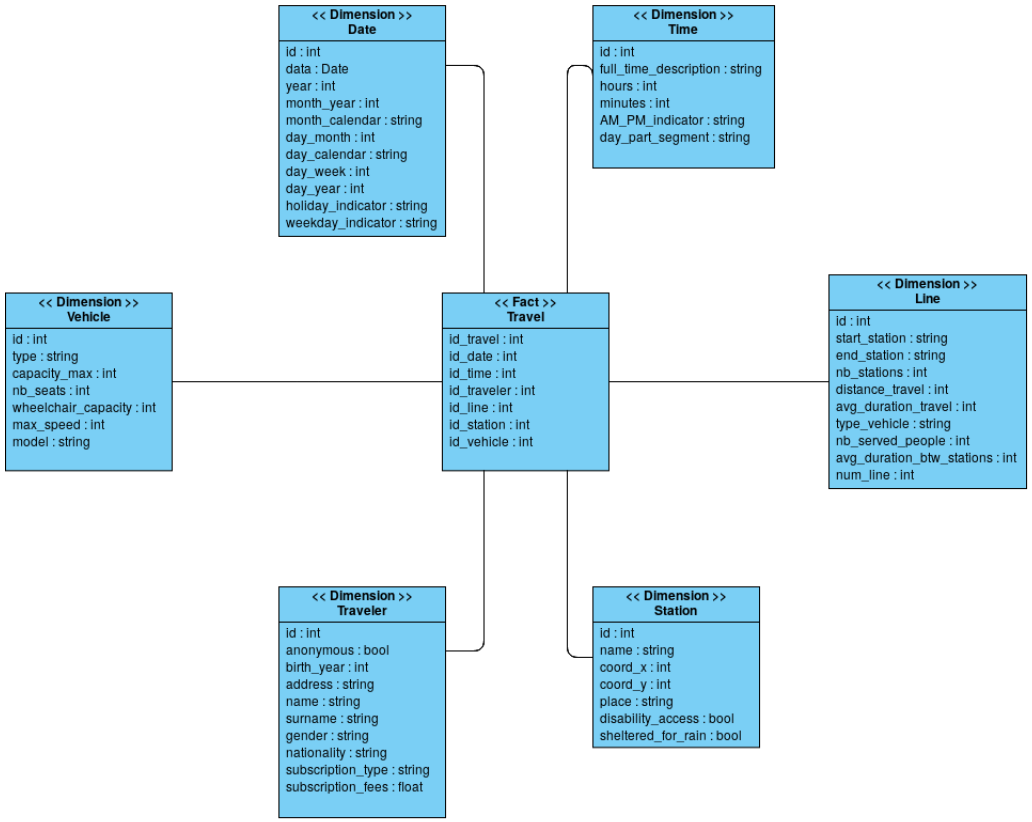

FIGURE 2.1 – modèle en étoile de l'action « voyages »

2.2.1 Discussion

— chaque tuple dans la table `Vehicle` désigne un véhicule possédé par la société.
— chaque tuple dans la table `Ligne` désigne une itinéraire prise par une ligne. Autrement dit, la même ligne peut avoir plusieurs itinéraires différentes (*e.g. la ligne 3 possède 4 itinéraires différentes*)
— chaque tuple dans la table `Station` désigne une station desservie par un véhicule.
— la table `Traveler` est une dimension qui contient deux dimensions corrélées (les abonnés et le voyageur anonyme non abonné utilisant un ticket) :

1. si le voyageur est **abonné**, alors on traite le tuple correspondant en tant qu'un **voyageur concret** dont les informations sont à notre disposition.
2. sinon, tous les **voyageurs non abonnés** seront représentés par

un seul tuple.

3. cette décision de corrélation est utilisée pour éviter la normalisation et l'introduction d'une superclasse abstraite étendue par les classes désignant les voyageurs abonnés et non abonnés.
4. nous utilisons ainsi l'attribut **anonymous** afin de distinguer les deux types de voyageurs. En effet, **anonymous** valera *true* quand le voyageur est non abonné, sinon il valera *true*.
5. le tuple du **voyageur non abonné** contiendra ainsi des **valeurs nulles** pour les attributs décrivant un **voyageur abonné**.

— chaque tuple dans la table `Travel` désigne un voyage effectué par un voyageur à un temps et une date données, en prenant un véhicule d'une ligne depuis un arrêt spécifié.

La table des voyages n'admet aucune mesure ; on se contentera d'utiliser les informations fournies par les dimensions qui suffiront largement pour analyser la fréquentation des différentes lignes et véhicules correspondant.

Remarques

— l'attribut **id_travel** de la table `Travel` est la clé primaire utilisée pour identifier un voyage (*dimension dégénérée*).
— l'attribut **nb_served_people** de la table `Line` désigne le nombre de passagers désservis par la ligne.
— l'attribut **place** de la table `Station` désigne l'endroit où se trouve la station (avenue X, rue Y, ...).
— l'attribut **wheelchair_capacity** de la table `Vehicle` désigne la capacité théorique maximale de personnes handicappées et de leurs fauteuils roulants.

2.3 Modélisation des maintenances

Les mesures de la table des maintenances sont :
— `cost` : mesure additive désignant le coût de maintenance d'un véhicule pour une transaction donnée.

2.3.1 Discussion

— chaque tuple dans la table `Employee` désigne un employé chez *tam-voyages*.
— chaque tuple dans la table `TechnicalArea` désigne un dépôt utilisé par *tam-voyages* pour maintenir des véhicules.

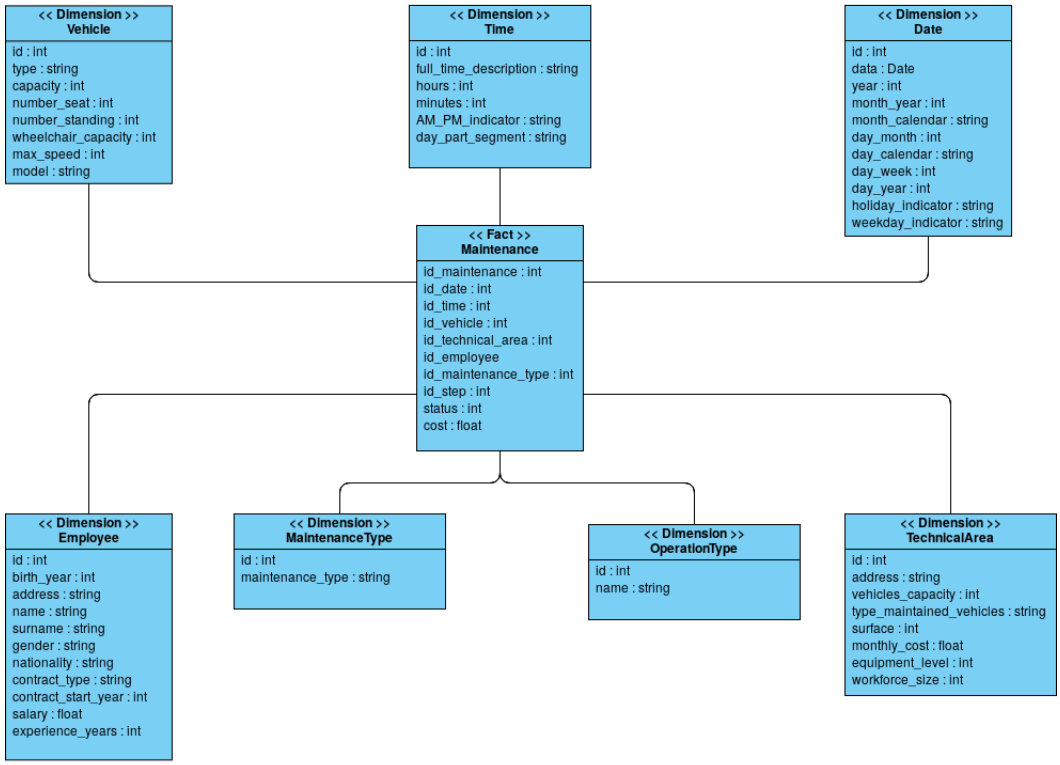

FIGURE 2.2 – modèle en étoile de l'action « maintenance »

— chaque tuple dans la table `MaintenanceType` désigne un type de maintenance effectuée (*e.g. changement de roues, maintenance du moteur, ...*)

— chaque tuple dans la table `OperationType` désigne une opération faite lors de la maintenance d'un véhicule (*i.e. type de transaction effectuée*). Les opérations possibles définies :

1. conduire le véhicule concerné vers un dépôt de maintenance
2. allouer un garage pour la maintenance du véhicule
3. attribuer la maintenance du véhicule à un spécialiste
4. éventuellement attendre l'acquisition de ressources nécessaires à la maintenance du véhicule
5. mise en marche du véhicule dans le réseau après sa maintenance

— chaque tuple dans la table `Maintenance` désigne l'état d'une opération lors de la maintenance d'un véhicule (*i.e. le grain choisi pour la maintenance est la transaction*).

Remarques

l'attribut **equipment_level** de la table `TechnicalArea` désigne le niveau de matériaux disponibles au local et peut prendre une valeur entre 1 (*pas assez équipé*) et 5 (*très bien équipé*).

2.4 Entrepôt de données obtenu

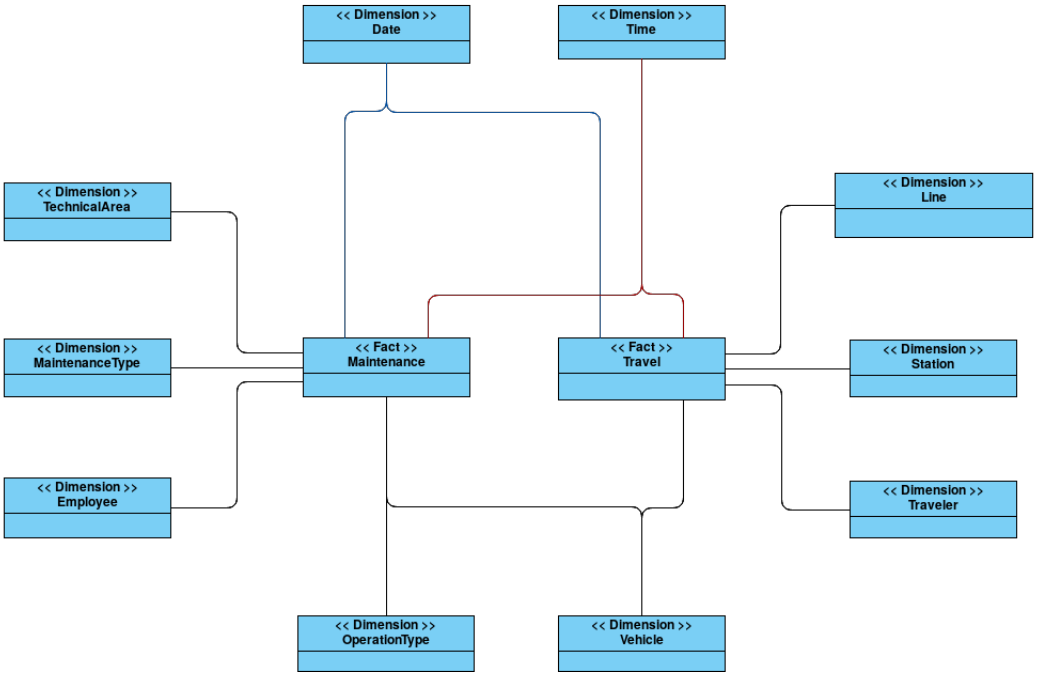

FIGURE 2.3 – le *data warehouse* résultant

Chapitre 3

Implémentation

3.1 Choix des technologies

Pour l'implémentation de notre entrepôt de données, nous avons décidé d'utiliser `Oracle`. Pour chaque dimension partagée entre les deux *data marts* nous avons créé deux vues virtuelles selon le niveau de détails nécessité pour chaque analyse.

Chapitre 4

Conclusion