Mini-Rapport BI - Conception Physique du Data Warehouse

Elaboré par :

Rania Assali

Maryem Amara

L2BI G1

Phase 1: Analyse et conception

1 Introduction

Cette partie présente la phase de modélisation conceptuelle et la conception physique de l'entrepôt de données pour le projet « Analyse des Réservations Hôtelières ».
L'objectif est de structurer les données du dataset « Hotel Booking Demand » dans un modèle multidimensionnel afin de répondre aux besoins analytiques et décisionnels du projet.

2 Définition des Besoins d'Affaires

La définition des besoins d'affaires repose sur l'identification des faits et mesures pertinentes pour les décideurs.

2.1 Faits et indicateurs clés :

Indicateur	Description
Montant des réservations	Total des revenus générés par les réservations
Nombre de réservations annulées	Nombre total de réservations annulées
Taux d'occupation	Ratio du nombre de chambres réservées sur le nombre total de chambres disponibles
Durée moyenne des séjours	Nombre moyen de nuits réservées
Répartition des réservations	Répartition par segment de marché et canal de distribution
Taux de fidélisation	Pourcentage de clients récurrents

2.2 Spécification des besoins fonctionnels

Le système BI devra permettre :

- L'analyse des tendances de réservation et d'annulation.
- La segmentation des clients par origine géographique et type de réservation.
- Le suivi des performances des hôtels.

3 Inspection des Données Sources

L'inspection des données constitue une étape primordiale pour valider l'exploitabilité et l'utilité des données dans la modélisation de l'entrepôt.
Cette étape comprend la vérification de l'accès aux données, la quantification des enregistrements et l'analyse de la qualité de chaque variable.
Source : hotel_bookings.csv (provenant de Kaggle)
Nombre de lignes : 119 390
Nombre de colonnes : 32

3.1 Tableau de synthèse de l'inspection des données

Colonne	Type	Description	Qualité (0-3)	Utilité (0-3)	Commentaires
hotel	Catégorie	Type d'hôtel : Resort / City	3	3	Bonne qualité et directement exploitable
is_canceled	Binaire	Annulation (1=Oui, 0=Non)	3	3	Aucune correction nécessaire
lead_time	Entier	Délai en jours avant l'arrivée	3	3	Utilisable tel quel pour les analyses temporelles
adr	Float	Tarif journalier moyen par réservation	3	3	Exploitable pour la facturation et KPI financiers
market_segment	Catégorie	Segment de marché	3	3	4 **Clé potentielle pour la dimension Marché**
distribution_channel	Catégorie	Canal de distribution utilisé	3	3	Importante pour la stratégie commerciale
country	Catégorie	Pays d'origine (ISO 3)	3	3	Utilisable pour des analyses géographiques
reserved_room_type	Catégorie	Type de chambre réservée	3	3	Peut servir pour la dimension Chambre

assigned_room_type	Catégorie	Type de chambre attribuée	3	2	Nécessite vérification de cohérence avec réservation
stays_in_weekend_nights	Entier	Nuits durant le weekend	3	3	Indicateur clé pour la durée de séjour
stays_in_week_nights	Entier	Nuits en semaine	3	3	Utilisable pour la segmentation des séjours
total_of_special_requests	Entier	Nombre de demandes spéciales	3	3	Pertinent pour la qualité de service

Évaluation : la majorité des variables sont exploitables sans traitement supplémentaire et présentent un intérêt certain pour l'analyse décisionnelle.
Les colonnes 'assigned_room_type' nécessitent une vérification pour s'assurer de la cohérence entre la réservation et l'attribution réelle de la chambre.

L'inspection des données débute par la vérification de l'accès à la source et se poursuit par une analyse des données disponibles.
Elle se conclut par une évaluation de la qualité et de l'utilité des données.

Colonne	*Type*	*Description*	*Qualité*	*Utilité*	*Commentaire*
hotel	Catégorie	Type d'hôtel (Resort / City)	3	3	Exploitable directement
is_canceled	Binaire	Indicateur d'annulation	3	3	Exploitable sans traitement
lead_time	Entier	Délai de réservation	3	3	Bonne qualité
adr	Float	Tarif journalier moyen	3	3	OK pour l'analyse
market_segment	Catégorie	Segment de marché	3	3	Exploitable
distribution_channel	Catégorie	Canal de distribution	3	3	Bonne qualité
country	Catégorie	Pays d'origine	3	3	Norme ISO-3

La qualité des données est jugée globalement très bonne et leur utilité élevée pour la modélisation.

3.2 Identification des Tables des Faits et des Dimensions

Après inspection, la table de faits suivante est définie :

Table de faits : Fait_Reservations

- id_fact_reservation (PK)
- id_temps (FK)
- id_client (FK)
- id_hotel (FK)
- id_chambre (FK)
- id_market_segment (FK)
- id_distribution_channel (FK)
- adr
- total_of_special_requests
- is_canceled
- weekend_nights
- week_nights

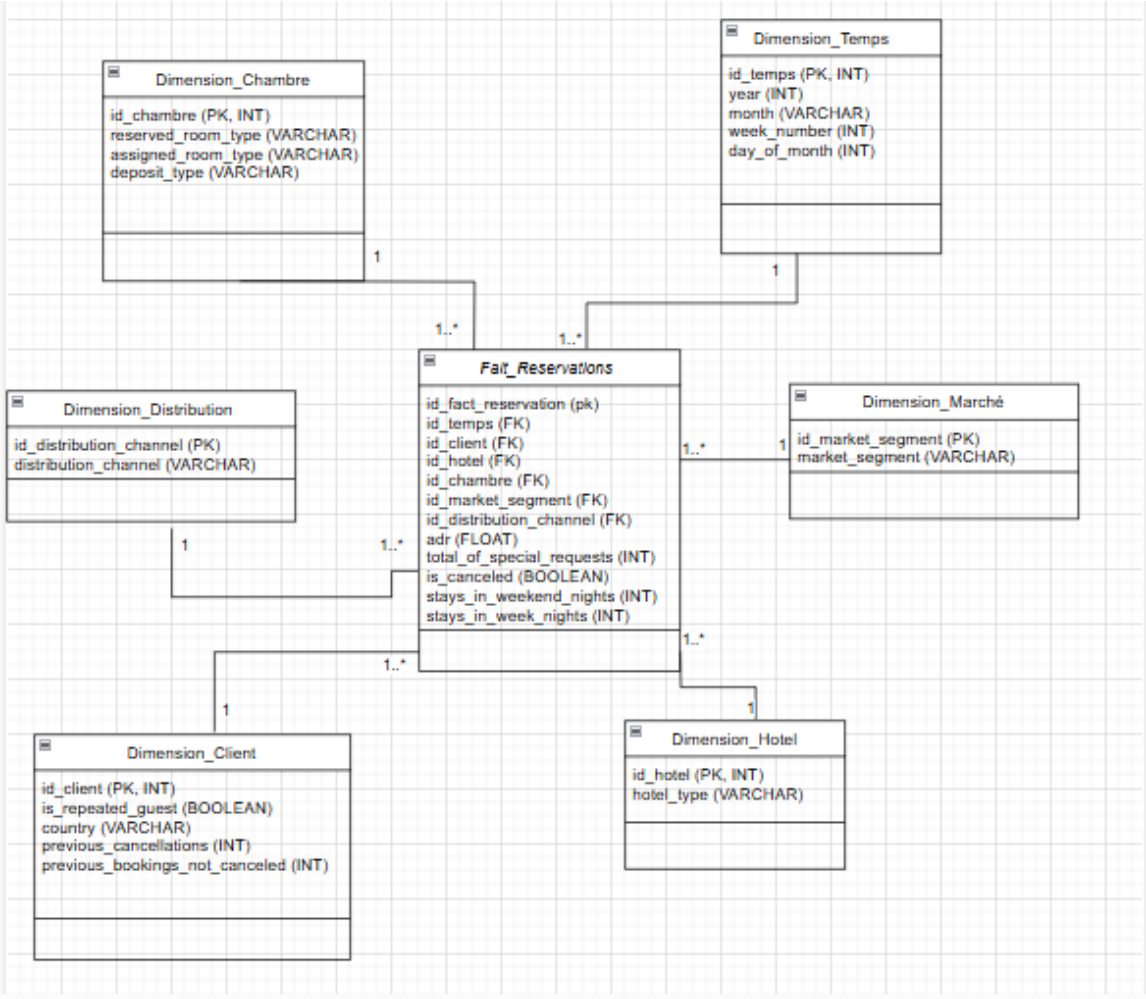
Dimensions identifiées : Dimension_Temps, Dimension_Client, Dimension_Hotel, Dimension_Chambre, Dimension_Marché, Dimension_Distribution.

4 Conception du Modèle Physique

Le modèle retenu est un schéma en Étoile (Star Schema).
Justification :

- Simplicité et lisibilité
- Optimisation des performances
- Facilité d'extension pour d'autres dimensions futures

Les relations entre la table de faits et les dimensions sont définies par des clés étrangères.



5 Conclusion

Cette modélisation conceptuelle et physique garantit un modèle robuste et performant pour les futures analyses BI et machine learning.
La prochaine étape consistera à développer les flux ETL et charger les données dans le data warehouse.

Phase 2: ETL et implémentation du data warehouse

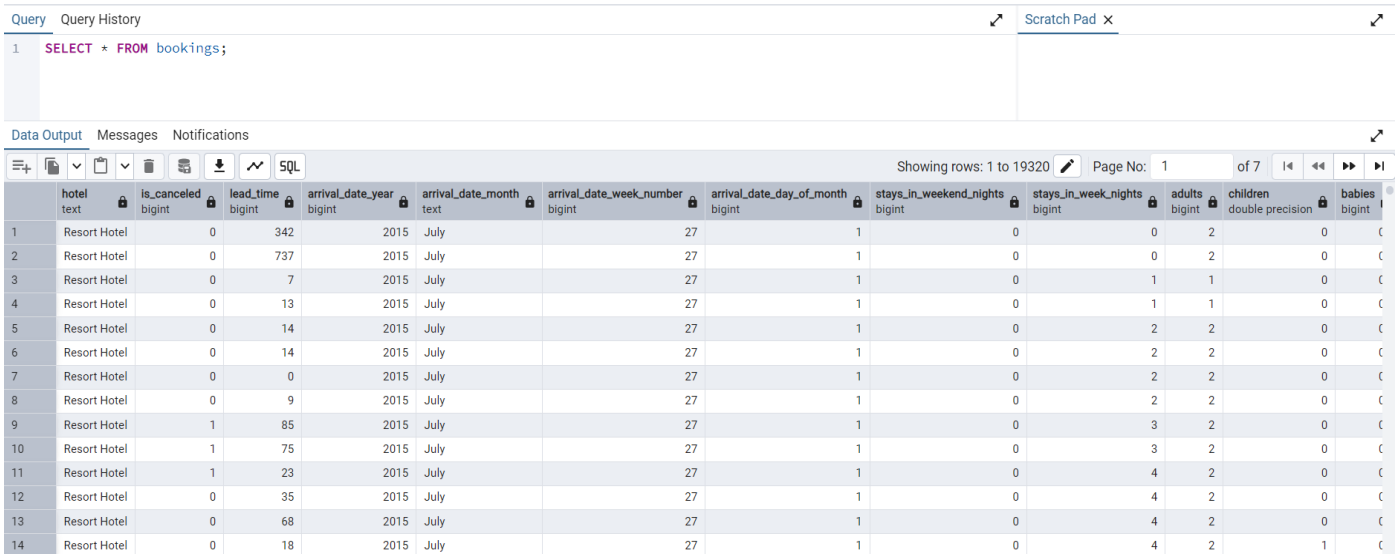
L'objectif de cette phase est de **charger les données brutes** issues du fichier CSV "Hotel Booking Demand" dans un **data warehouse relationnel** en suivant la modélisation en étoile (modèle dimensionnel).

🛠️ Étapes réalisées :

1. **Création du schéma relationnel** dans PostgreSQL :
   - Tables de dimensions : `DIM_HOTEL`, `DIM_GUEST`, `DIM_MEAL`, `DIM_TIME`, etc.
   - Table de faits : `FACT_BOOKING`
   - Ajout de clés primaires et étrangères
2. **Nettoyage et transformation des données** :
   - Traitement des valeurs nulles (`NULL` remplacés, types ajustés)
   - Uniformisation des formats de dates et de chaînes
   - Création de colonnes dérivées (`is_rep`, `meal_type`, etc.)
3. **Chargement des données (Load)** :
   - Insertion dans chaque table de dimension à partir du fichier source
   - Jointures et insertion des lignes dans la table de faits
4. **Vérifications** :
   - Contrôle des clés étrangères
   - Contrôle de la qualité des données
   - Tests sur les doublons et les incohérences temporelles

🔍 Outils utilisés :

- PostgreSQL
- SQL (DDL, DML, fonctions analytiques)
- Table "RAW_DATA" chargée à partir du CSV



Query | Query History | Scratch Pad

```
SELECT * FROM bookings;
```

Data Output | Messages | Notifications

Showing rows: 1 to 19320 | Page No: 1 of 7

	hotel text	is_canceled bigint	lead_time bigint	arrival_date_year bigint	arrival_date_month text	arrival_date_week_number bigint	arrival_date_day_of_month bigint	stays_in_weekend_nights bigint	stays_in_week_nights bigint	adults bigint	children double precision	babies bigint
1	Resort Hotel	0	342	2015	July	27	1	0	0	2	0	0
2	Resort Hotel	0	737	2015	July	27	1	0	0	2	0	0
3	Resort Hotel	0	7	2015	July	27	1	0	1	1	0	0
4	Resort Hotel	0	13	2015	July	27	1	0	1	1	0	0
5	Resort Hotel	0	14	2015	July	27	1	0	2	2	0	0
6	Resort Hotel	0	14	2015	July	27	1	0	2	2	0	0
7	Resort Hotel	0	0	2015	July	27	1	0	2	2	0	0
8	Resort Hotel	0	9	2015	July	27	1	0	2	2	0	0
9	Resort Hotel	1	85	2015	July	27	1	0	3	2	0	0
10	Resort Hotel	1	75	2015	July	27	1	0	3	2	0	0
11	Resort Hotel	1	23	2015	July	27	1	0	4	2	0	0
12	Resort Hotel	0	35	2015	July	27	1	0	4	2	0	0
13	Resort Hotel	0	68	2015	July	27	1	0	4	2	0	0
14	Resort Hotel	0	18	2015	July	27	1	0	4	2	1	0

1. Nettoyage des données

Le nettoyage des données a été réalisé directement dans la table source `bookings` via plusieurs requêtes SQL :

- Les valeurs `NULL` dans les colonnes `children`, `country`, `agent`, et `company` ont été remplacées par des valeurs par défaut (`0` ou `'Unknown'`).
- Les chaînes de caractères dans des colonnes importantes (`meal`, `hotel`, `market_segment`, etc.) ont été standardisées grâce à des fonctions SQL telles que `UPPER()`, `INITCAP()`, et `TRIM()` afin d'éviter les incohérences dues à la casse ou aux espaces.

Query | Query History

```sql
UPDATE public.bookings SET children = 0 WHERE children IS NULL;
UPDATE public.bookings SET country = 'Unknown' WHERE country IS NULL;
UPDATE public.bookings SET agent = '0' WHERE agent IS NULL;
UPDATE public.bookings SET company = '0' WHERE company IS NULL;

-- Nettoyage des chaînes
UPDATE public.bookings SET meal = UPPER(TRIM(meal));
UPDATE public.bookings SET hotel = INITCAP(TRIM(hotel));
UPDATE public.bookings SET market_segment = INITCAP(TRIM(market_segment));

-- Colonnes calculées
ALTER TABLE public.bookings ADD COLUMN IF NOT EXISTS total_guests INT;
ALTER TABLE public.bookings ADD COLUMN IF NOT EXISTS total_stays INT;

UPDATE public.bookings
SET total_guests = adults + children + babies,
    total_stays = stays_in_week_nights + stays_in_weekend_nights;

-- Regrouper les pays rares
```

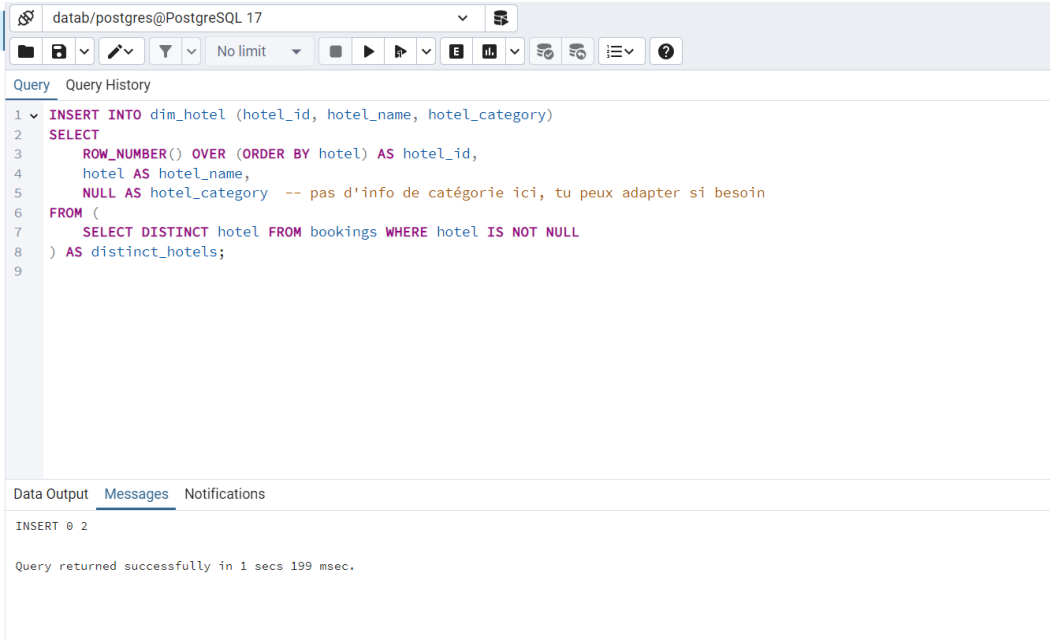
Data Output | Messages | Notifications

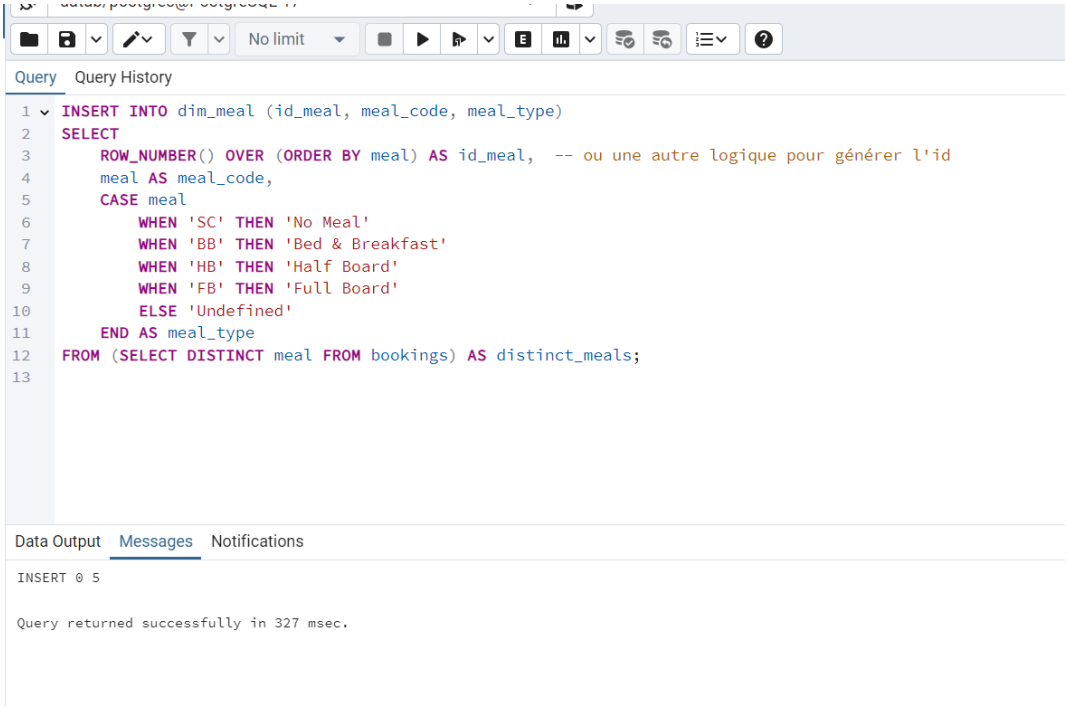
```
UPDATE 2390

Query returned successfully in 13 secs 673 msec.
```

2. Chargement des tables de dimensions

Les tables de dimensions `dim_hotel` et `dim_meal` ont été créées puis alimentées à partir des données nettoyées de la table `bookings` :

- **dim_hotel** : insertion des hôtels distincts avec génération d'identifiants uniques (`hotel_id`) via la fonction `ROW_NUMBER()`.
- **dim_meal** : insertion des codes repas distincts (`meal_code`) avec attribution d'un identifiant (`meal_id`) et classification du type de repas (`meal_type`) selon la valeur du code.

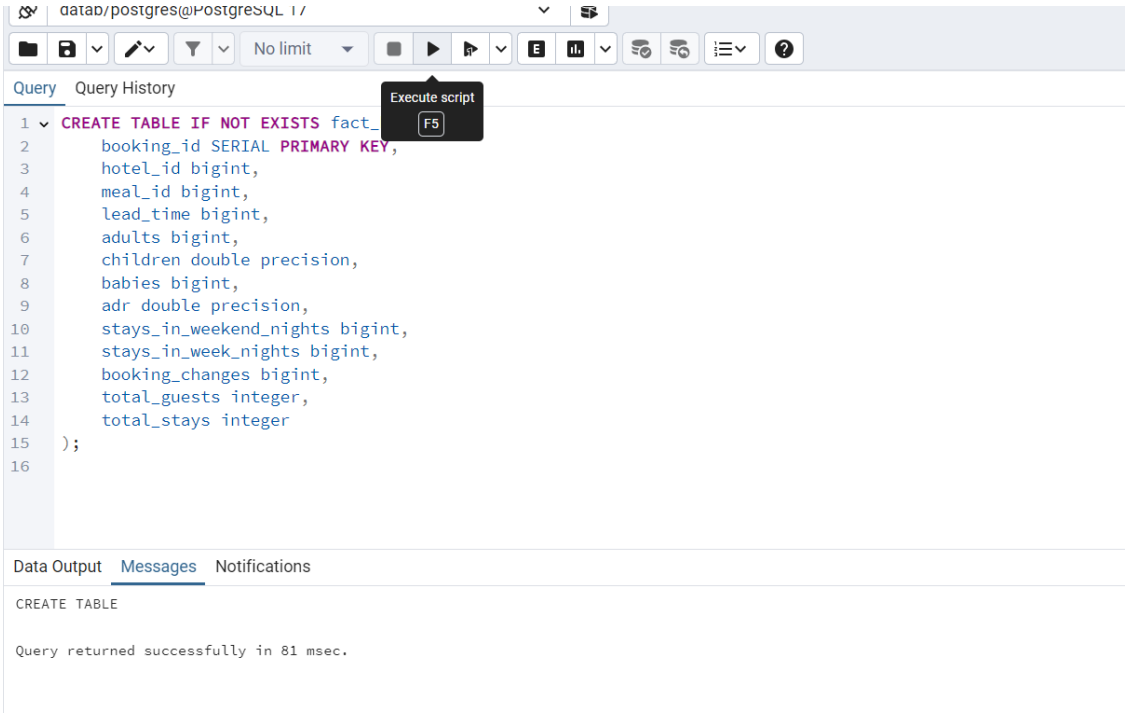


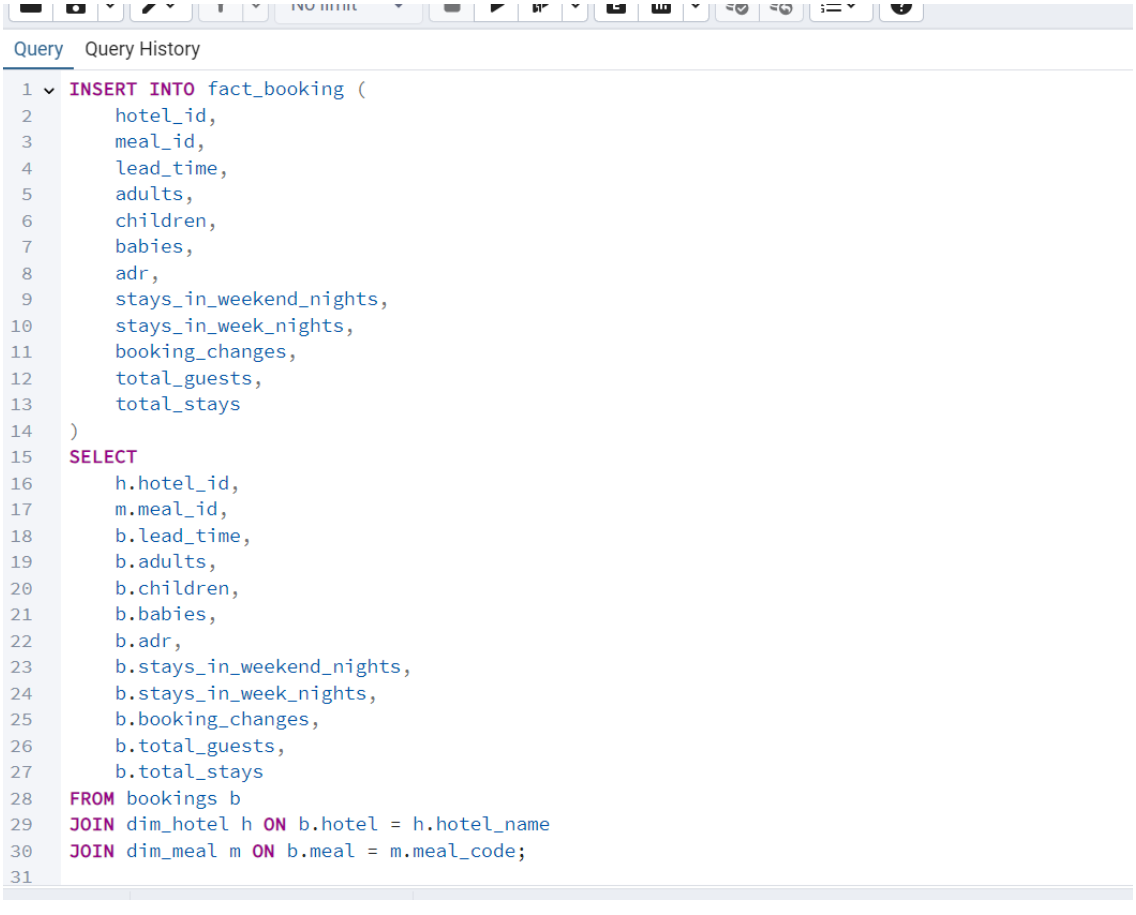


3. Création et chargement de la table de faits

La table de faits `fact_booking` a été créée pour contenir les données transactionnelles des réservations avec les clés étrangères vers les dimensions :

- Les données ont été chargées par une requête `INSERT INTO ... SELECT ... JOIN` qui lie la table source `bookings` aux tables `dim_hotel` et `dim_meal` sur les attributs `hotel_name` et `meal_code`.
- Ce chargement permet d'insérer dans la table de faits les mesures quantitatives (lead_time, adultes, adr, etc.) en garantissant la cohérence via les clés des dimensions.


No limit
Query
Query History
Execute script
F5
CREATE TABLE IF NOT EXISTS fact_
booking_id SERIAL PRIMARY KEY,
hotel_id bigint,
meal_id bigint,
lead_time bigint,
adults bigint,
children double precision,
babies bigint,
adr double precision,
stays_in_weekend_nights bigint,
stays_in_week_nights bigint,
booking_changes bigint,
total_guests integer,
total_stays integer
);
Data Output
Messages
Notifications
CREATE TABLE
Query returned successfully in 81 msec.


Query
Query History
INSERT INTO fact_booking (
hotel_id,
meal_id,
lead_time,
adults,
children,
babies,
adr,
stays_in_weekend_nights,
stays_in_week_nights,
booking_changes,
total_guests,
total_stays
)
SELECT
h.hotel_id,
m.meal_id,
b.lead_time,
b.adults,
b.children,
b.babies,
b.adr,
b.stays_in_weekend_nights,
b.stays_in_week_nights,
b.booking_changes,
b.total_guests,
b.total_stays
FROM bookings b
JOIN dim_hotel h ON b.hotel = h.hotel_name
JOIN dim_meal m ON b.meal = m.meal_code;

Phase 3: Analyse et visualisation

3.1 Objectif de la phase :

L'objectif principal de cette phase est de développer **un tableau de bord interactif** à partir des données intégrées lors de la phase précédente. L'enjeu est de transformer les données brutes en informations décisionnelles à travers des **indicateurs clés de performance (KPIs).**
Cette visualisation permettra une compréhension rapide des tendances, une analyse comparative et un pilotage plus précis de l'activité hôtelière.

3.2 Visualisations développées

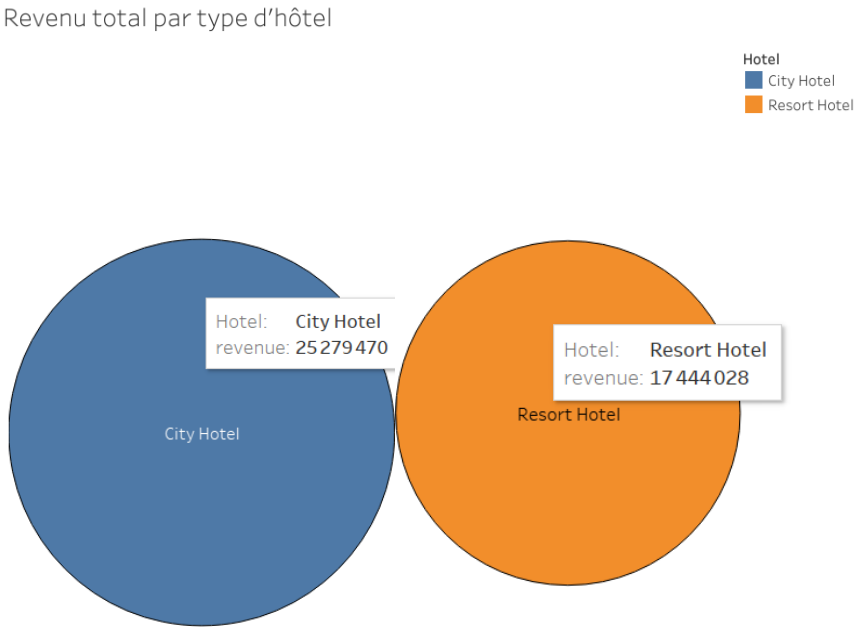
Trois visualisations ont été conçues pour répondre à des besoins spécifiques :

3.2.1 Revenu total par type d'hôtel

- **Objectif** : mesurer la rentabilité entre hôtels urbains et balnéaires.

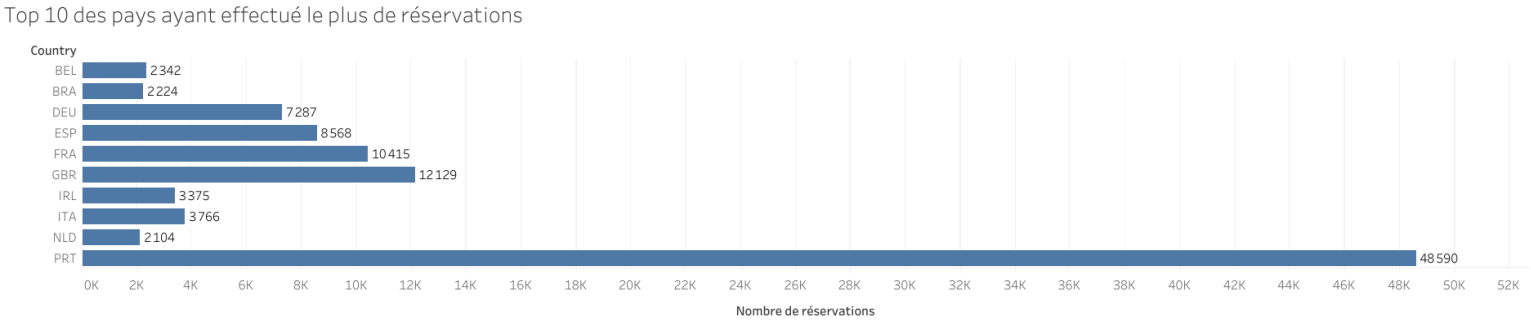
  KPI : `SUM(revenue)` **avec des champs calculés** `revenue = adr × (stays_in_week_nights + stays_in_weekend_nights)`

- **Résultat** : le City Hotel génère généralement plus de revenus sur l'ensemble de l'année.



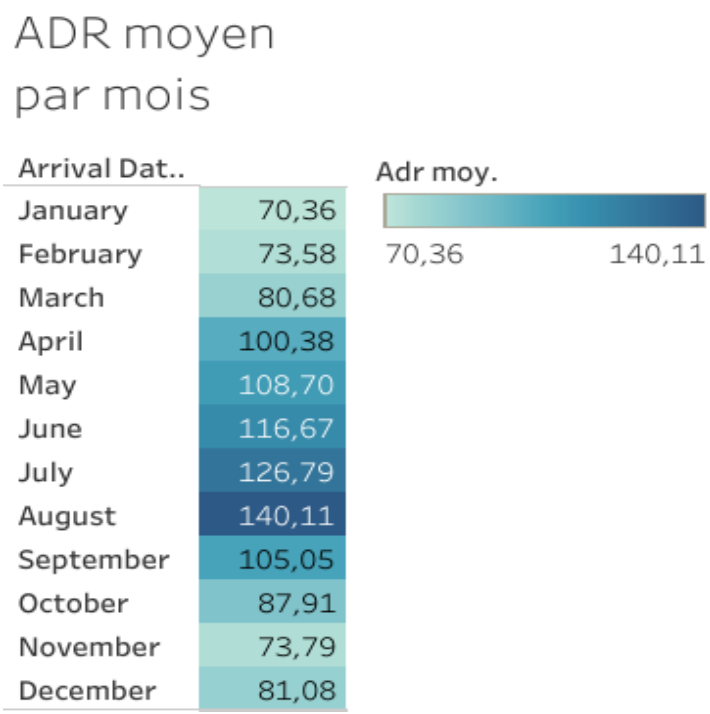
3.2.2 Réservations par pays – Top 10

- **Objectif** : identifier les marchés les plus actifs en nombre de réservations.
- **KPI** : `COUNT(number of records)`
- **Résultat** : le Portugal, le Royaume-Uni et la France sont les principales sources de clientèle.



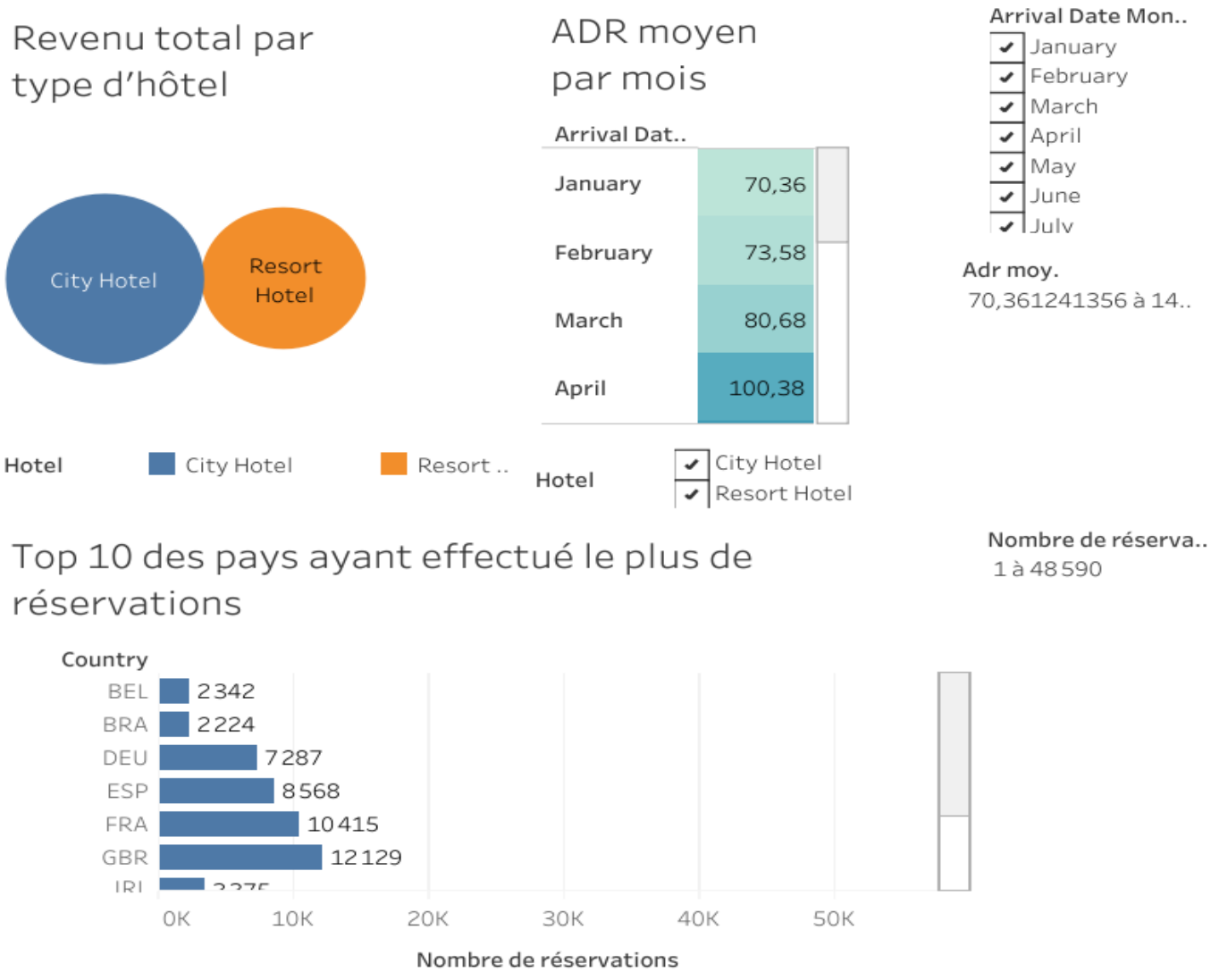
3.2.3 ADR moyen par mois

- **Objectif** : analyser les fluctuations saisonnières du tarif moyen par nuitée (`adr`).
- **KPI** : `AVG(adr)`
- **Résultat** : hausse significative en été, montrant une forte demande touristique.



3.3 Conception du tableau de bord

Le tableau de bord regroupe l'ensemble des visualisations dans une interface unifiée. Il est structuré comme suit :



Le dashboard permet à l'utilisateur de :

- **Filtrer dynamiquement** les données
- **Explorer les tendances** par segment ou période
- **Afficher les infos bulles** détaillées
- **Comparer les performances** par pays, mois ou type d'hôtel

3.4 Résultat et interprétation :

L'analyse visuelle met en lumière plusieurs **insights stratégiques** :

- Le **City Hotel** domine en revenu, mais subit moins d'annulations que le Resort Hotel.
- Certains pays, comme le Portugal et le Royaume-Uni, concentrent une large part des réservations.
- Les mois d'été affichent une **hausse notable de l'ADR**, signe d'une saisonnalité forte.

3.5 Conclusion :

Cette phase a permis de valoriser les données intégrées précédemment à travers un **tableau de bord professionnel et interactif**.
Les visualisations offrent une lecture immédiate des tendances et facilitent l'**aide à la décision** dans le contexte hôtelier.
Grâce à Tableau Public, les résultats sont **accessibles, partagés et visuellement impactants**.

Phase 4: Machine Learning

4.1 Contexte et justification :

Le secteur hôtelier fait face à un défi majeur lié aux **annulations de réservations**, qui peuvent engendrer des pertes financières significatives, désorganiser la gestion des chambres et compromettre la prévision de la demande réelle.
Dans ce contexte, **l'anticipation du comportement des clients** devient un enjeu stratégique.

Pouvoir identifier à l'avance la probabilité qu'un client revienne ou non permettrait de :

- Réduire les pertes liées aux annulations inattendues
- Mettre en place des **stratégies de surbooking maîtrisées**
- Adapter les actions marketing et les politiques tarifaires pour maximiser la fidélisation

4.2 Problématique business :

Peut-on prédire la fidélité d'un client — c'est-à-dire, déterminer s'il est un client récurrent (`is_repeated_guest = 1`) ou non (`= 0`) — en se basant sur les données liées à sa réservation ?

Il s'agit d'un **problème de classification binaire**, à la fois simple à modéliser et directement exploitable en contexte hôtelier pour :

- Segmenter les clients
- Optimiser les programmes de fidélisation
- Personnaliser les offres et les campagnes de communication

4.3 Données et préparation :

Source des données :

Les données utilisées proviennent du jeu de données *Hotel Booking Demand*, sous forme d'un fichier CSV (`booking.csv`).
Chaque ligne représente une réservation, avec des informations sur le client, le séjour, le type de chambre, les canaux de réservatio n, les repas, etc.

Nettoyage et prétraitement :

Plusieurs étapes ont été nécessaires pour préparer les données à l'entraînement d'un modèle de machine learning :

- **Suppression des colonnes non pertinentes ou redondantes** :
  - `reservation_status`, `reservation_status_date`, `agent`, `company`, `reserved_room_type`, `assigned_room_type`
- **Traitement des valeurs manquantes** :
  - Remplacement des valeurs manquantes dans `children` par `0`
  - Remplissage de la variable `country` avec la valeur `'Unknown'` en cas d'absence

- **Regroupement des modalités rares** :
    - Les pays ayant un faible nombre de réservations ont été regroupés sous une catégorie générique `'Other'` pour limiter la cardinalité

Création de nouvelles variables :

Afin d'enrichir la base et d'améliorer la qualité prédictive du modèle, des variables dérivées ont été créées :

- `total_guests` = `adults` + `children` + `babies`
- `total_stays` = `stays_in_week_nights` + `stays_in_weekend_nights`
- `total_previous_bookings` = `previous_cancellations` + `previous_bookings_not_canceled`

Encodage et normalisation :

- Les variables **numériques** ont été normalisées à l'aide de `StandardScaler`
- Les variables **catégorielles** ont été encodées avec `OneHotEncoder`
- Un `ColumnTransformer` a été utilisé pour combiner les deux types de transformations dans un pipeline cohérent

```
PS C:\Users\LENOVO> & C:/Users/LENOVO/AppData/Local/Programs/Python/Python313/python.exe d:/Phase4_Repeated_Guest.py
Taille entraînement : (95512, 23) (95512,)
Taille test : (23878, 23) (23878,)
Répartition de is_repeated_guest (entraînement) :
 is_repeated_guest
0    0.968088
1    0.031912
1    0.031912
Name: proportion, dtype: float64
```

4.4 Modélisation et évaluation :

Modèles utilisés :

Deux modèles de classification ont été développés pour prédire la fidélité des clients (is_repeated_guest) :

- **Random Forest Classifier** : un modèle d'ensemble robuste basé sur des arbres de décision
- **Régression Logistique** : un modèle linéaire simple et interprétable

Ces modèles ont été intégrés dans un **pipeline complet**, combinant à la fois les étapes de

```
PS C:\Users\LENOVO> & C:/Users/LENOVO/AppData/Local/Programs/Python/Python313/python.exe d:/Phase4_Repeated_Guest.py
Taille entraînement : (95512, 23) (95512,)
Taille test : (23878, 23) (23878,)
Répartition de is_repeated_guest (entraînement) :
 is_repeated_guest
0    0.968088
1    0.031912
Name: proportion, dtype: float64
Modèles définis : Random Forest et Logistic Regression.
```

préparation (encodage, normalisation) et l'entraînement.

Traitement du déséquilibre :

La variable cible étant fortement déséquilibrée (~3 % de clients fidèles), un rééquilibrage a été appliqué à l'aide de **SMOTE** (Synthetic Minority Over-sampling Technique), uniquement sur l'ensemble d'entraînement.

Évaluation des modèles :

Les deux modèles ont été évalués à l'aide de plusieurs métriques :

Métrique	**Random Forest**	**Régression Logistique**
Accuracy	0.99	0.95
Rappel (classe fidèle)	0.93	0.90
Précision (classe fidèle)	0.80	0.93
F1-score (classe fidèle)	0.86	0.54
AUC - ROC	0.99	0.97

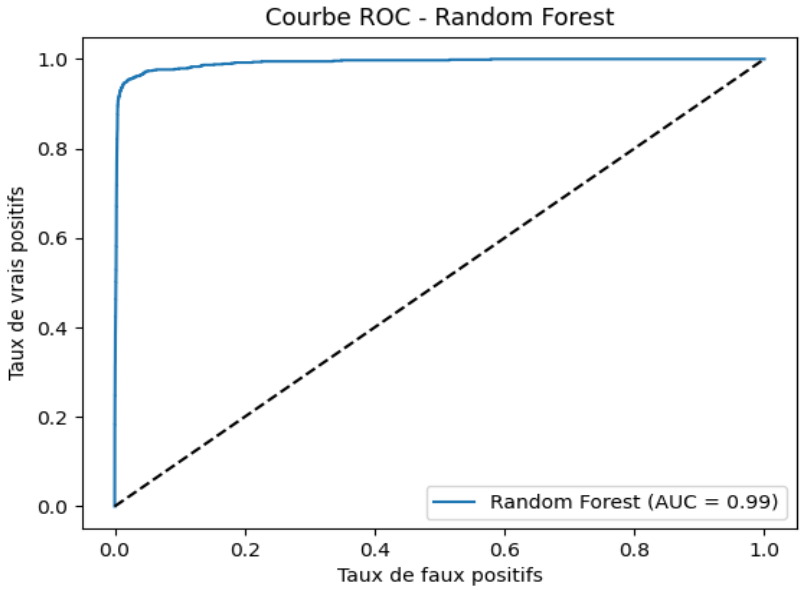
Matrices de confusion :

Les performances des modèles ont été représentées visuellement à l'aide de **matrices de confusion** :

```
Classification Report (Random Forest) :
              precision    recall  f1-score   support

           0       1.00      0.99      0.99     23116
           1       0.80      0.93      0.86       762

    accuracy                           0.99     23878
   macro avg       0.90      0.96      0.93     23878
weighted avg       0.99      0.99      0.99     23878

AUC-ROC (Random Forest) : 0.9920396060221666
Matrice de confusion (Random Forest) :
[[22937   179]
 [   57   705]]
```

```
=== Évaluation de Logistic Regression ===
Classification Report (Logistic Regression) :
              precision    recall  f1-score   support

           0       1.00      0.95      0.97     23116
           1       0.39      0.90      0.54       762

    accuracy                           0.95     23878
   macro avg       0.69      0.93      0.76     23878
weighted avg       0.98      0.95      0.96     23878

AUC-ROC (Logistic Regression) : 0.9732208469074607
Matrice de confusion (Logistic Regression) :
[[22021  1095]
 [   75   687]]
```

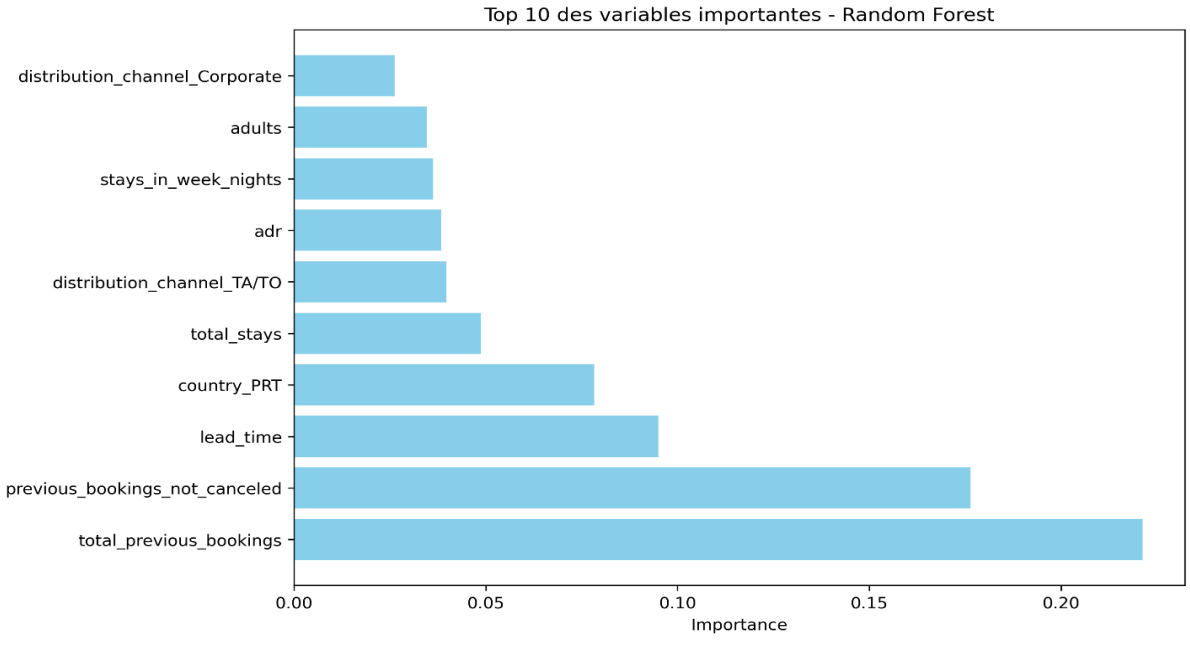
Courbe ROC – Random Forest :



Importance des variables

Le modèle Random Forest permet d'identifier les **variables les plus influentes** dans la prédiction de la fidélité :

- `lead_time`
- `total_guests`
- `adr` (prix moyen journalier)
- `total_previous_bookings`
- `market_segmentdistribution_channel`



Validation finale

Le modèle **Random Forest** a été retenu comme le plus performant grâce à :

- Son **équilibre entre rappel et précision**
- Une **courbe ROC proche de la perfection**
- Une **capacité explicative claire** (feature importance)

Le pipeline complet a été **sauvegardé** pour un futur déploiement ou intégration (`rf_pipeline.joblib`).

4.5 Conclusion et perspectives :

L'objectif de cette phase était de démontrer la capacité à appliquer des techniques de machine learning pour résoudre une problématique métier réelle dans le domaine hôtelier : **prédire la fidélité d'un client** à partir de ses caractéristiques de réservation.

À travers une démarche rigoureuse comprenant la **préparation des données**, la **gestion du déséquilibre de classes** avec SMOTE, et la **comparaison de deux modèles supervisés**, nous avons pu mettre en évidence la pertinence de cette approche.

Le modèle **Random Forest** s'est révélé le plus performant avec une précision globale de 99 %, un rappel de 93 % pour les clients fidèles, et une AUC de 0.99. De plus, il a permis d'identifier les facteurs les plus influents dans la prédiction, tels que le **lead time**, le **nombre total de clients**, et les **réservations précédentes**.

Bénéfices pour l'entreprise hôtelière :

- Meilleure connaissance des clients fidèles
- Ciblage marketing plus efficace
- Anticipation des comportements de réservation
- Personnalisation des offres et programmes de fidélisation