Propriétaire	Société THE QUANTIC FACTORY
Auteur	Morgan CARON
Doc Release	v1.0
Date	30/08/2022
Usage	test technique recrutement développeur

Table des matières

I/ cadre du test

Proposer au candidat la réalisation d'un algorithme s'inscrivant dans un environnement technique :

- local / linux
- mySql
- GoLang

Niveau :

intermédiaire Durée :

5 jours

Lieu : at home

NE PAS HÉSITER A CONTACTER MORGAN CARON :
morgan@quanticfy.io / +33 6 65 04 28 94 (what's app)
(cela est même vivement conseillé pour bien comprendre l'énoncé du test).

Il est possible que vous ne parveniez pas au bout du test dans le temps imparti.
Pas de panique !
Le fait que toutes les features ne soient pas implémentées n'est pas discriminant.

Seront considéré avant tout :

- le respect des consignes (§ suivant)
- le bon dosage entre : l'abstraction et le 'hard coding'
- la structuration, la simplicité et la lisibilité du code
- le stratégies et la simplicité employées pour résoudre des problèmes 'complexes'
- le stabilité du code
- sa réalisation cohérente et end-to-end : certes toutes les features peuvent ne pas être implémentées mais le binaire doit pouvoir s'exécuter en local sur le poste du candidat, avoir un début et une fin => aussi, toutes les fonctions dont le développement est initié doivent être achevées.

Rendu :

- le code (archive contenant tous les .go)
- un meet / visio pour :
    - debrieffing
    - un partage d'écran pour montrer l'exécution du code (live logs)

quanticfy

II/ consignes de développement

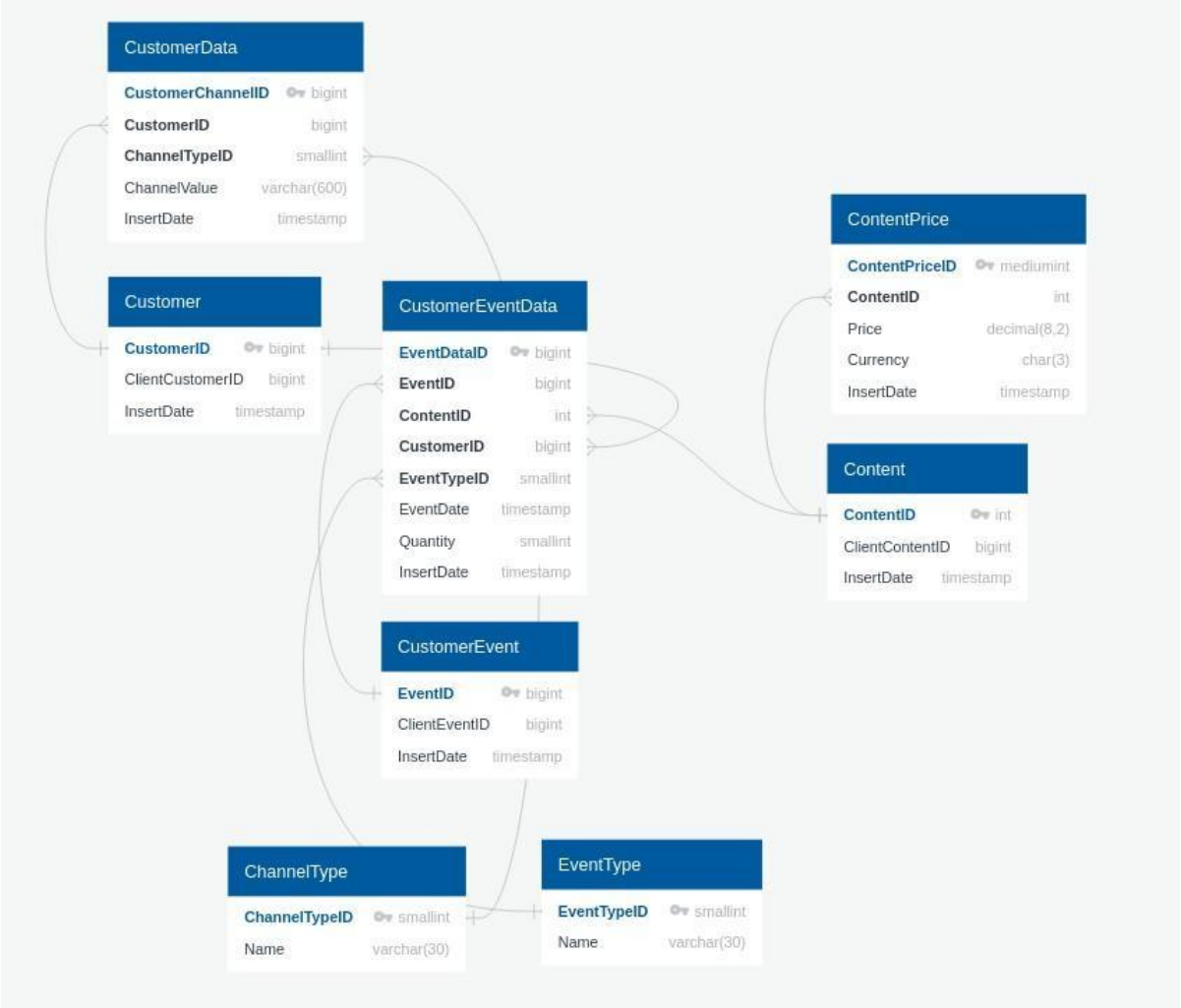
L'algorithme devra être construit et structuré en respectant le schéma BigData standard : Load - Compute in Memory - Export.

Un petit nombre de bonnes pratiques devront être respectées :

- documentation ‘lite’ + code commenté
- log : verbose mode, horodatage et classification standard (info, warning…), étapes de traitement
- recette unitaire ‘lite’
- sécurité : credentials en variable d’environnement si possible
- pour tous les traitement de données, une barre (ou système) indiquant la progression du process doit être utilisé
- politique d’utilisation des packages : natifs ou réputés stables
- ‘mass insert’ pour les exports (INSERT) de données en base
- jointures non autorisées pour les SELECT

III/ énoncé du test

La bdd présente le schéma suivant :



ChannelType :

ChannelTypeID	Name
1	Email
2	PhoneNumber
3	Postal
4	MobileID
5	Cookie

EventType :

EventTypeID	Name
1	sent
2	view
3	click
4	visit
5	cart
6	purchase

Accès (fake) :

- IP : 44.333.11.22
- login : candidat2020
- pwd : dfskj_878$*=

A des fins de recette - conseillé mais non obligatoire, il est possible très facilement de créer cette base en local en utilisant l'outil https://www.quickdatabasediagrams.com/
et en y intégrant ce code :

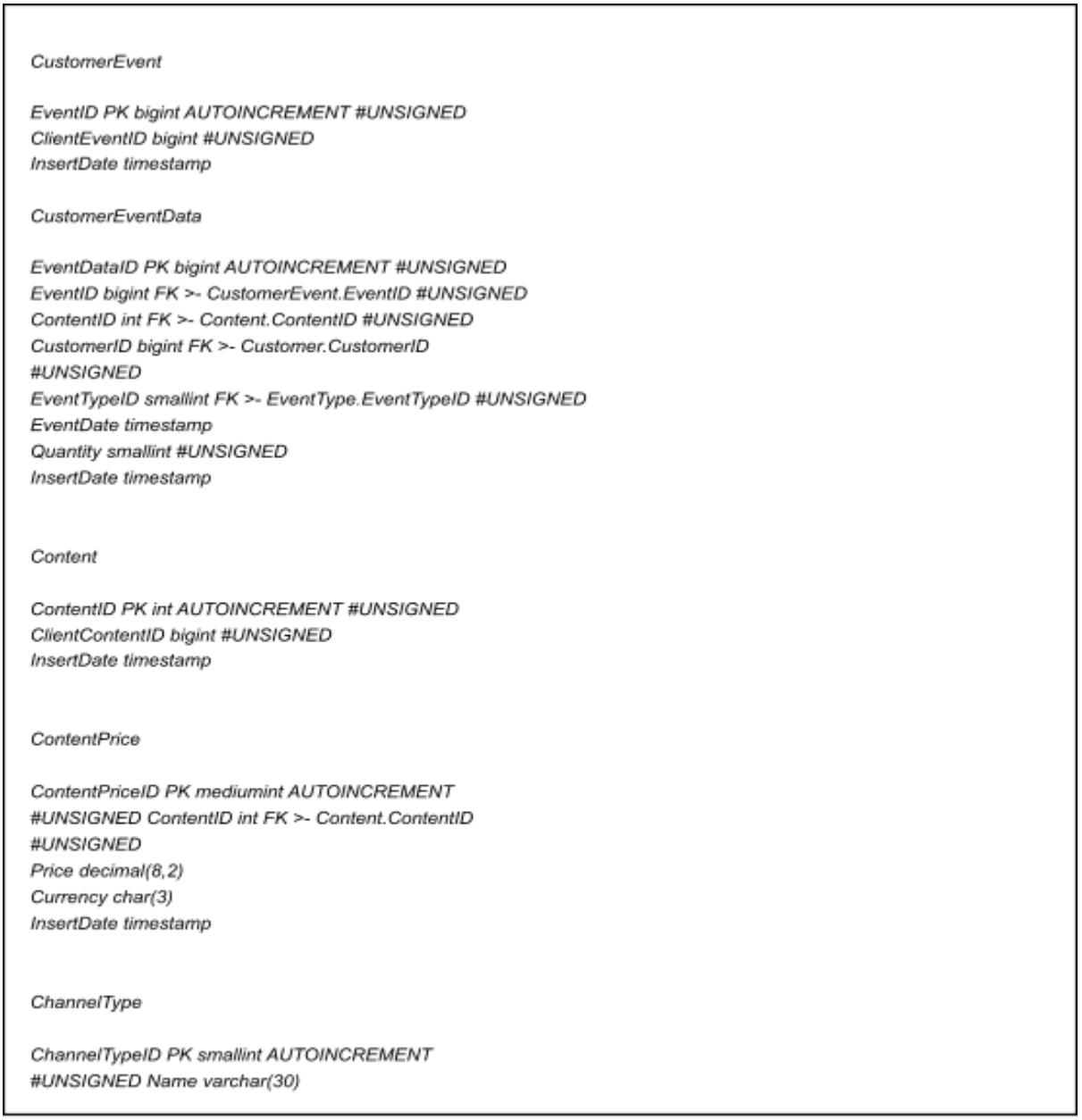
```
# Modify this code to update the DB schema
diagram. # To reset the sample schema, replace
everything with # two dots ('..' - without quotes).

Customer

CustomerID PK bigint AUTOINCREMENT
#UNSIGNED ClientCustomerID bigint #UNSIGNED
InsertDate timestamp


CustomerData

CustomerChannelID PK bigint AUTOINCREMENT
#UNSIGNED CustomerID bigint FK >-
Customer.CustomerID #UNSIGNED ChannelTypeID
smallint FK >- ChannelType.ChannelTypeID ChannelValue
varchar(600)
InsertDate timestamp
```



puis en utilisant la fonction EXPORT -> MySQL.
Il faut ensuite en revanche feeder les tables à la main ou par script (random)
et corriger/adapter/ajuster les ‘types’ de donnée.

Il s’agit d’une base de données e-commerce standard structurée autour des contacts (Customer).

Les Customers sont ‘décrits’ avec leur méta donnée (CustomerData) comme leur numéro de téléphone, leur email…
Pour chaque Customer, leurs actions (events) sont enregistrées dans CustomerEventData, comme leurs achats (Purchase).
Les events sont eux mêmes décrits par leur méta data : quels produits ont été achetés (les Contents) en quelle quantité (Quantity) à quelle date (EventDate)...
Chaque Content est également décrit par sa méta donné (comme le prix de chaque Content = article = produit) dans la table ContentPrice

L’objectif global :

- déterminer les meilleurs clients, c’est à dire ceux ayant apporté le plus de chiffre d’affaires à l’ecommerçant
- ces meilleurs clients font parti du premier quantile (si le CA est trié par ordre décroissant, sinon c’est le dernier), quantile qui est une variable fixée à 2.5%
- exporter ces meilleurs clients dans une table mySQL (mêmes accès)
- analyser comment sont répartis les clients par chiffre d’affaires client

NB : le chiffre d’affaires (CA) pour un client est :
somme pour tous ses EventData de type 6 et tous les contents de chaque Eventdata de quantity * price(content)

L’algorithme doit :

- LOAD
    - load in memory de la donnée nécessaire pour la suite
- TREAT. :
    - déterminer et enregistrer dans une map, Customer par Customer le total Chiffre d’Affaires (c’est à dire la somme de tous ses paniers) depuis le 01/04/2020 => il est conseillé de créer également des ‘objets’ associés (des slices par ex) qui permettront de trier/ordonner les résultats
        - print, dans le log, de 10 entrées de cette map (au hasard)
    - top_customer :
        - variable ‘quantile’, qui, multipliée par 100 est un multiple de 100 (entier) et qui représentera les quantiles
            - ex : quantile = 0.025 => 40 quantiles : 0-2.5% | 2.5 - 5% | … | 97.5% - 100%
        - créer une map avec les N clients du premier quantile (=les meilleurs clients si on considère les chiffres d’affaire classés par ordre décroissant) et print leur nombre
        - créer une map associée à une ‘struct’ pour enregistrer quantile par quantile de chiffres d’affaire (les 2.5% CA les plus élevés, les 2.5% suivants…, jusqu’au dernier quantile 97.5-100%) :
            - nb de clients dans le quantile
            - CA max du quantile (ie. le client du quantile qui a généré le plus de CA)

- CA min du quantile (ie. le client du quantile qui a généré le moins de CA)

- EXPORT :
    - sauvegarder (mêmes accès sql) dans une table ‘test_export_DATE’ la donnée :
        - structure : CustomerID # Email # CA
        - DATE : YYYYMMDD
        - ce qui signifie donc :
            - si plusieurs exports le même jour =>
                - update du champ CA si le customerID existe déjà
                - nouvelle ligne pour un nouveau customer
            - 1er export d’un nouveau jour => le script doit créer la table puis la feed

BON COURAGE,
BONNE CHANCE !