1. Obtenir l'utilisateur ayant le prénom "Muriel" et le mot de passe "test11", sachant que l'encodage du mot de passe est effectué avec l'algorithme Sha1.

```
store=# SELECT * FROM client WHERE prenom = 'Muriel' AND password = encode(digest('test11', 'sha1'), 'hex');
 id | prenom |  nom   |       email        | ville |                password
----+--------+--------+--------------------+-------+-----------------------------------------
 11 | Muriel | Dupuis | muriel@example.com | Paris | 100c4e57374fc998e57164d4c0453bd3a4876a58
(1 row)
```

2. Obtenir la liste de tous les produits qui sont présent sur plusieurs commandes.

```
store=# SELECT nom, COUNT(*) AS nombre_produit FROM commande_ligne GROUP BY nom
 HAVING COUNT(*) > 1 ORDER BY nombre_produit desc;
    nom     | nombre_produit
------------+----------------
 Produit 6D |              4
 Produit 67 |              3
 Produit DD |              2
 Produit 52 |              2
 Produit DE |              2
 Produit D9 |              2
 Produit 95 |              2
 Produit D6 |              2
 Produit 2E |              2
 Produit FC |              2
 Produit 00 |              2
 Produit 3C |              2
 Produit E1 |              2
 Produit 8A |              2
 Produit 12 |              2
 Produit 78 |              2
 Produit 93 |              2
 Produit C4 |              2
 Produit 07 |              2
(19 rows)
```

3. Obtenir la liste de tous les produits qui sont présent sur plusieurs commandes et y ajouter une colonne qui liste les identifiants des commandes associées.

```
store=# SELECT nom, COUNT(*), array_to_string(array_agg(commande_id ORDER BY commande_id ), ',')
FROM commande_ligne GROUP BY nom HAVING COUNT(*) > 1 ORDER BY COUNT(*) DESC;
    nom     | count | array_to_string
------------+-------+-----------------
 Produit 6D |     4 | 23,29,40,41
 Produit 67 |     3 | 15,17,26
 Produit 12 |     2 | 12,18
 Produit 2E |     2 | 16,46
 Produit 3C |     2 | 31,36
 Produit 52 |     2 | 15,42
 Produit 78 |     2 | 2,4
 Produit 8A |     2 | 23,41
 Produit 93 |     2 | 28,47
 Produit 95 |     2 | 22,32
 Produit C4 |     2 | 20,46
 Produit D6 |     2 | 7,33
 Produit D9 |     2 | 3,7
 Produit DD |     2 | 20,25
 Produit DE |     2 | 32,48
 Produit E1 |     2 | 6,22
 Produit 00 |     2 | 5,14
 Produit FC |     2 | 10,21
 Produit 07 |     2 | 4,10
(19 rows)
```

4. Enregistrer le prix total à l'intérieur de chaque ligne des commandes, en fonction du prix unitaire et de la quantité

```
store=# UPDATE commande_ligne SET prix_total = (quantite * prix_unitaire);
UPDATE 120
store=# SELECT prix_total FROM commande_ligne;
     prix_total
--------------------
             148.71
             324.96
              34.96
             334.76
              35.94
             132.37
             612.56
              861.4
             323.84
              237.6
              54.78
              864.5
              89.72
  764.3700000000001
             100.14
             808.85
 202.64999999999998
            1000.08
 1129.3000000000002
             222.62
             488.75
                348
                3.8
                 97
              94.72
             387.73
             225.56
             226.38
             632.38
  590.9399999999999
  912.8700000000001
:
```

5. Obtenir le montant total pour chaque commande et y voir facilement la date associée à cette commande ainsi que le prénom et nom du client associé

```
store=# SELECT client.prenom, client.nom, commande.date_achat, SUM(prix_total) AS prix_commande
FROM commande_ligne LEFT JOIN commande ON commande.id = commande_ligne.commande_id LEFT JOIN cli
ent ON client.id = commande.client_id GROUP BY client.nom, client.prenom, commande.date_achat OR
DER BY nom;
   prenom   |    nom    | date_achat |   prix_commande
------------+-----------+------------+-------------------
 Maris      | Buisson   | 2019-01-18 |             136.4
 Maris      | Buisson   | 2019-01-25 |            1928.59
 Emilien    | Camus     | 2019-01-14 |                97
 Emilien    | Camus     | 2019-01-27 |            995.76
 Emilien    | Camus     | 2019-02-13 |  719.5400000000001
 Gustave    | Collin    | 2019-01-03 |             370.7
 Gustave    | Collin    | 2019-01-17 | 1646.3100000000002
 Gustave    | Collin    | 2019-01-21 |             907.2
 Gustave    | Collin    | 2019-02-06 |            700.96
 Muriel     | Dupuis    | 2019-01-04 |            132.37
 Muriel     | Dupuis    | 2019-02-02 |            362.81
 Muriel     | Dupuis    | 2019-02-03 |            673.65
 Manon      | Durand    | 2019-01-15 | 482.45000000000005
 Manon      | Durand    | 2019-01-19 | 1285.8100000000002
 Manon      | Durand    | 2019-02-01 |            472.82
 Fabrice    | Foucher   | 2019-01-13 |            1063.17
 Fabrice    | Foucher   | 2019-02-09 |             554.7
 Maurice    | Huet      | 2019-02-02 |               784
 Maurice    | Huet      | 2019-02-11 |            1398.06
 Lucas      | Jung      | 2019-01-09 |  764.3700000000001
 Lucas      | Jung      | 2019-01-11 |            1000.08
 Lucas      | Jung      | 2019-02-05 |  751.6400000000001
 Lucas      | Jung      | 2019-02-16 |  592.3199999999999
 Jacinthe   | Langlois  | 2019-02-14 |  620.6800000000001
 Jacinthe   | Langlois  | 2019-02-15 | 1321.9099999999999
 Joachim    | Leon      | 2019-02-05 |             114.4
:
```

6. (difficulté très haute) Enregistrer le montant total de chaque commande dans le champ intitulé “cache_prix_total”

```
store=# UPDATE commande SET cache_prix_total = t2.prix_total FROM (SELECT commande_id, SUM(commande_ligne.prix_total)
AS prix_total FROM commande_ligne GROUP BY commande_id) AS t2 WHERE commande.id = t2.commande_id;
UPDATE 48
store=# SELECT * FROM commande ORDER BY client_id;
 id | client_id | date_achat | reference |  cache_prix_total
----+-----------+------------+-----------+--------------------
 34 |         2 | 2019-02-08 | 001074    |  810.1999999999999
  7 |         2 | 2019-01-10 | 000214    | 1111.6399999999999
 48 |         2 | 2019-02-19 | 001496    |            2637.18
 40 |         2 | 2019-02-12 | 008590    |             856.14
 41 |         2 | 2019-02-12 | 001639    |             573.02
 15 |         3 | 2019-01-17 | 001167    | 1646.3100000000002
 19 |         3 | 2019-01-21 | 005510    |              907.2
  2 |         3 | 2019-01-03 | 007120    |              370.7
 32 |         3 | 2019-02-06 | 001858    |  700.9599999999999
 11 |         4 | 2019-01-14 | 003757    |                 97
 23 |         4 | 2019-01-27 | 005217    |  995.7599999999999
 42 |         4 | 2019-02-13 | 002426    |  719.5400000000001
 35 |         5 | 2019-02-08 | 005379    |              93.68
  4 |         6 | 2019-01-07 | 003425    |            2090.18
 14 |         6 | 2019-01-16 | 002286    |            1223.32
 31 |         7 | 2019-02-05 | 008653    |  751.6400000000001
  6 |         7 | 2019-01-09 | 000996    |  764.3700000000001
  8 |         7 | 2019-01-11 | 008084    |            1000.08
 45 |         7 | 2019-02-16 | 002213    |  592.3199999999999
 26 |         8 | 2019-02-02 | 007277    |                784
 39 |         8 | 2019-02-11 | 003770    |            1398.06
 25 |         9 | 2019-02-01 | 007879    |             472.82
 12 |         9 | 2019-01-15 | 004939    | 482.45000000000005
 17 |         9 | 2019-01-19 | 001369    |            1285.81
 37 |        10 | 2019-02-09 | 002220    |             185.28
 30 |        10 | 2019-02-05 | 000469    |              114.4
 27 |        11 | 2019-02-02 | 002745    |             362.81
 28 |        11 | 2019-02-03 | 001893    |             673.65
  3 |        11 | 2019-01-04 | 002957    |             132.37
  9 |        12 | 2019-01-11 | 009773    | 1129.3000000000002
 24 |        12 | 2019-01-29 | 000706    |             238.99
 46 |        12 | 2019-02-17 | 004759    |            1518.11
 44 |        13 | 2019-02-15 | 008768    | 1321.9099999999999
 43 |        13 | 2019-02-14 | 007209    |  620.6800000000001
 33 |        14 | 2019-02-07 | 003330    |             441.85
 13 |        14 | 2019-01-16 | 003421    |             451.94
 16 |        15 | 2019-01-18 | 008974    |              136.4
 22 |        15 | 2019-01-25 | 008459    |            1928.59
 36 |        16 | 2019-02-09 | 003672    |              554.7
 10 |        16 | 2019-01-13 | 004616    | 1063.1699999999998
 18 |        17 | 2019-01-20 | 009924    | 1061.9199999999998
 21 |        17 | 2019-01-23 | 002359    |             510.07
  5 |        17 | 2019-01-08 | 008255    |             954.22
 20 |        17 | 2019-01-22 | 007778    |            1169.15
 47 |        19 | 2019-02-18 | 007155    |             611.52
 38 |        19 | 2019-02-10 | 000086    |             567.13
:
```

7. Obtenir le montant global de toutes les commandes, pour chaque mois

```
store=# SELECT EXTRACT(MONTH FROM date_achat) AS mois, SUM(cache_prix_total) FROM commande GROUP BY mois;
 mois |   sum
------+----------
    1 | 21259.57
    2 | 18616.68
(2 rows)
```

8. Obtenir la liste des 10 clients qui ont effectué le plus grand montant de commandes, et obtenir ce montant total pour chaque client.

```
store=# SELECT client.nom, client.prenom, SUM(commande.cache_prix_total) AS client_montant FROM commande
LEFT JOIN client ON client.id = commande.client_id GROUP BY client.nom, client.prenom, commande.client_id
 ORDER BY client_montant DESC LIMIT 10;
    nom    |   prenom   |  client_montant
-----------+------------+------------------
 Vespasien | Valentin   | 5988.179999999999
 Saunier   | Patrick    |           3695.36
 Collin    | Gustave    |           3625.17
 Riou      | Olivier    |            3313.5
 Jung      | Lucas      |           3108.41
 Riou      | Christiane |            2886.4
 Durand    | Manon      |           2241.08
 Huet      | Maurice    |           2182.06
 Buisson   | Maris      |           2064.99
 Langlois  | Jacinthe   |           1942.59
(10 rows)
```

9. Obtenir le montant total des commandes pour chaque date

```
store=# SELECT date_achat, SUM(cache_prix_total) AS montant_date FROM commande GROUP BY date_achat
ORDER BY date_achat asc;
 date_achat |    montant_date
------------+--------------------
 2019-01-01 | 508.62999999999994
 2019-01-03 |              370.7
 2019-01-04 |             132.37
 2019-01-07 |            2090.18
 2019-01-08 |             954.22
 2019-01-09 |  764.3700000000001
 2019-01-10 | 1111.6399999999999
 2019-01-11 |            2129.38
 2019-01-13 | 1063.1699999999998
 2019-01-14 |                 97
 2019-01-15 | 482.45000000000005
 2019-01-16 |            1675.26
 2019-01-17 | 1646.3100000000002
 2019-01-18 |              136.4
 2019-01-19 |            1285.81
 2019-01-20 | 1061.9199999999998
 2019-01-21 |             907.2
 2019-01-22 |            1169.15
 2019-01-23 |             510.07
 2019-01-25 |            1928.59
 2019-01-27 |  995.7599999999999
 2019-01-29 |             238.99
 2019-02-01 |             472.82
 2019-02-02 |            1146.81
 2019-02-03 |             673.65
 2019-02-04 |            1255.08
 2019-02-05 |  866.0400000000001
 2019-02-06 |  700.9599999999999
 2019-02-07 |             441.85
 2019-02-08 |  903.8799999999999
 2019-02-09 |             739.98
 2019-02-10 |             567.13
 2019-02-11 |            1398.06
 2019-02-12 | 1429.1599999999999
 2019-02-13 |  719.5400000000001
 2019-02-14 |  620.6800000000001
 2019-02-15 | 1321.9099999999999
 2019-02-16 |  592.3199999999999
 2019-02-17 |            1518.11
 2019-02-18 |             611.52
 2019-02-19 |            2637.18
(41 rows)
```

10. Ajouter une colonne intitulée “category” à la table contenant les commandes. Cette colonne contiendra une valeur numérique

```
store=# ALTER TABLE commande ADD category INT;
ALTER TABLE
store=# SELECT * FROM commande;
 id | client_id | date_achat | reference |  cache_prix_total  | category
----+-----------+------------+-----------+--------------------+----------
  1 |        20 | 2019-01-01 | 004214    | 508.62999999999994 |
  2 |         3 | 2019-01-03 | 007120    |              370.7 |
  3 |        11 | 2019-01-04 | 002957    |             132.37 |
  4 |         6 | 2019-01-07 | 003425    |            2090.18 |
  5 |        17 | 2019-01-08 | 008255    |             954.22 |
  6 |         7 | 2019-01-09 | 000996    |  764.3700000000001 |
  7 |         2 | 2019-01-10 | 000214    | 1111.6399999999999 |
  8 |         7 | 2019-01-11 | 008084    |            1000.08 |
  9 |        12 | 2019-01-11 | 009773    | 1129.3000000000002 |
 10 |        16 | 2019-01-13 | 004616    | 1063.1699999999998 |
 11 |         4 | 2019-01-14 | 003757    |                 97 |
 12 |         9 | 2019-01-15 | 004939    | 482.45000000000005 |
 13 |        14 | 2019-01-16 | 003421    |             451.94 |
 14 |         6 | 2019-01-16 | 002286    |            1223.32 |
 15 |         3 | 2019-01-17 | 001167    | 1646.3100000000002 |
 16 |        15 | 2019-01-18 | 008974    |              136.4 |
 17 |         9 | 2019-01-19 | 001369    |            1285.81 |
 18 |        17 | 2019-01-20 | 009924    | 1061.9199999999998 |
 19 |         3 | 2019-01-21 | 005510    |              907.2 |
:
```

11. Enregistrer la valeur de la catégorie, en suivant les règles suivantes :

- ○ “1” pour les commandes de moins de 200€
- ○ “2” pour les commandes entre 200€ et 500€
- ○ “3” pour les commandes entre 500€ et 1.000€
- ○ “4” pour les commandes supérieures à 1.000€

```
store=# update commande set category = ( case when cache_prix_total<200 then 1 when cache_prix_total<500
 then 2 when cache_prix_total<1000 then 3 else 4 end);
UPDATE 48
store=# SELECT * FROM commande;
 id | client_id | date_achat | reference |  cache_prix_total  | category
----+-----------+------------+-----------+--------------------+----------
  1 |        20 | 2019-01-01 | 004214    | 508.62999999999994 |        3
  2 |         3 | 2019-01-03 | 007120    |              370.7 |        2
  3 |        11 | 2019-01-04 | 002957    |             132.37 |        1
  4 |         6 | 2019-01-07 | 003425    |            2090.18 |        4
  5 |        17 | 2019-01-08 | 008255    |             954.22 |        3
  6 |         7 | 2019-01-09 | 000996    |  764.3700000000001 |        3
  7 |         2 | 2019-01-10 | 000214    | 1111.6399999999999 |        4
  8 |         7 | 2019-01-11 | 008084    |            1000.08 |        4
  9 |        12 | 2019-01-11 | 009773    | 1129.3000000000002 |        4
 10 |        16 | 2019-01-13 | 004616    | 1063.1699999999998 |        4
 11 |         4 | 2019-01-14 | 003757    |                 97 |        1
 12 |         9 | 2019-01-15 | 004939    | 482.45000000000005 |        2
 13 |        14 | 2019-01-16 | 003421    |             451.94 |        2
 14 |         6 | 2019-01-16 | 002286    |            1223.32 |        4
 15 |         3 | 2019-01-17 | 001167    | 1646.3100000000002 |        4
 16 |        15 | 2019-01-18 | 008974    |              136.4 |        1
 17 |         9 | 2019-01-19 | 001369    |            1285.81 |        4
 18 |        17 | 2019-01-20 | 009924    | 1061.9199999999998 |        4
 19 |         3 | 2019-01-21 | 005510    |              907.2 |        3
 20 |        17 | 2019-01-22 | 007778    |            1169.15 |        4
:
```

12. Créer une table intitulée “commande_category” qui contiendra le descriptif de ces catégories

```
store=# CREATE TABLE commande_category(id SERIAL, nom VARCHAR(255) NOT NULL, PRIMARY KEY(id));
CREATE TABLE
```

```
store=# SELECT * FROM commande_category;
 id | nom
----+-----
(0 rows)
```

13. Insérer les 4 descriptifs de chaque catégorie au sein de la table précédemment créée

```
store=# INSERT INTO commande_category(id, nom) VALUES (1, 'commandes de moins de 200€'),
(2, 'commandes entre 200€ et 500€'), (3, 'commandes entre 500€ et 1.000€'), (4, 'commande
s supérieures à 1.000€');
INSERT 0 4
```

```
store=# SELECT * FROM commande_category;
 id |              nom
----+-------------------------------
  1 | commandes de moins de 200€
  2 | commandes entre 200€ et 500€
  3 | commandes entre 500€ et 1.000€
  4 | commandes supérieures à 1.000€
(4 rows)
```

14. Supprimer toutes les commandes (et les lignes des commandes) inférieur au 1er février 2019. Cela doit être effectué en 2 requêtes maximum

```
store=# DELETE FROM commande_ligne WHERE commande_id IN (SELECT id FROM commande WHERE date_achat < '2019-02-01');
DELETE 58
store=# DELETE FROM commande WHERE date_achat < '2019-02-01';
DELETE 24
```