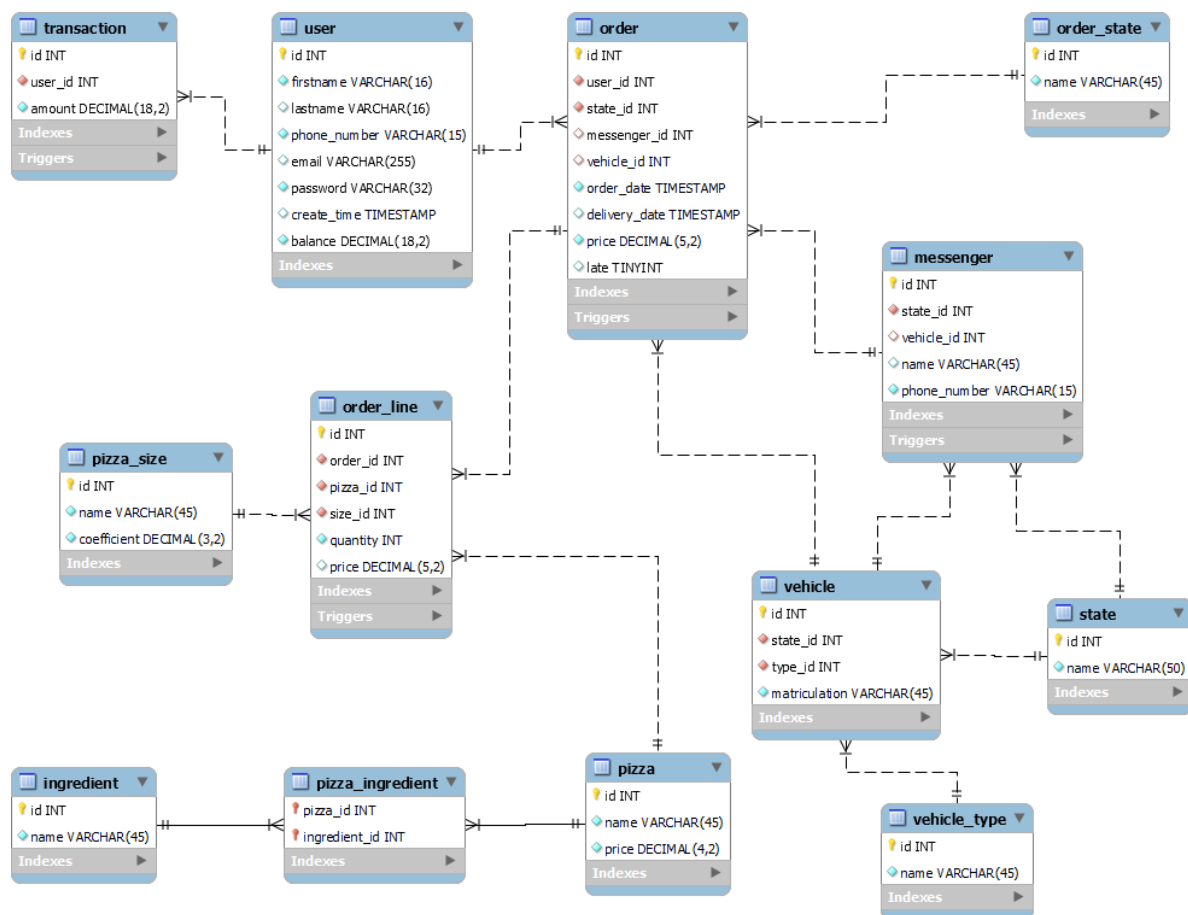


# Rapport du projet de gestion de l'entreprise de pizzas à domicile : FlashPizza.

## I. Conception de la base de données

### A. Modèle entité-association



## B. Modèle relationnel

USER(id, firstname, lastname, balance, phone\_number, email, password)

id : clé primaire

TRANSACTION(id, user\_id, amount)

id : clé primaire

user\_id : clé étrangère ref à id dans USER

STATE(id, name)

id : clé primaire

VEHICLE\_TYPE(id, name)

id : clé primaire

VEHICLE(id, state\_id, type\_id, matriculation)

id : clé primaire

state\_id : clé étrangère ref à id dans STATE

type\_id : clé étrangère ref à id dans VEHICLE\_TYPE

MESSENGER(id, state\_id, name, phone\_number, vehicle\_id)

id : clé primaire

state\_id : clé étrangère ref à id dans STATE

ORDER\_STATE(id, name)

id : clé primaire

ORDER(id, user\_id, state\_id, messenger\_id, vehicle\_id, order\_date, delivery\_date, price, late)

id : clé primaire

user\_id : clé étrangère ref à id dans USER

state\_id : clé étrangère ref à id dans ORDER\_STATE

messenger\_id : clé étrangère ref à id dans MESSENGER

vehicle\_id : clé étrangère ref à id dans VEHICLE

PIZZA(id, name, price)

id : clé primaire

PIZZA\_SIZE(id, name, coefficient)

id : clé primaire

ORDER\_LINE(id, order\_id, pizza\_id, size\_id, quantity, price)

id : clé primaire

order\_id : clé étrangère ref à id dans ORDER

pizza\_id : clé étrangère ref à id dans PIZZA

size\_id : clé étrangère ref à id dans PIZZA\_SIZE

INGREDIENT(id, name)

id : clé primaire

PIZZA\_INGREDIENT(pizza\_id, ingredient\_id)

pizza\_id, ingredient\_id : clé primaire

pizza\_id : clé étrangère ref à id dans PIZZA

ingredient\_id : clé étrangère ref à id dans INGREDIENT

## C. Script de la création des tables

```
-- MySQL Script generated by MySQL Workbench
-- Sun Jun 19 20:05:10 2022
-- Model: New Model    Version: 1.0
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_
DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';

-- -----
-- Schema flashpizza
-- -----

-- -----
-- Schema flashpizza
-- -----

CREATE SCHEMA IF NOT EXISTS `flashpizza` DEFAULT CHARACTER SET utf8 ;
USE `flashpizza` ;

-- -----
-- Table `flashpizza`.`ingredient`
-- -----

CREATE TABLE IF NOT EXISTS `flashpizza`.`ingredient` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb3;

-- -----
-- Table `flashpizza`.`state`
-- -----

CREATE TABLE IF NOT EXISTS `flashpizza`.`state` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(50) NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb3;

-- -----
-- Table `flashpizza`.`vehicle_type`
-- -----

CREATE TABLE IF NOT EXISTS `flashpizza`.`vehicle_type` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb3;
```

```
-- Table `flashpizza`.`vehicle`
```

```
CREATE TABLE IF NOT EXISTS `flashpizza`.`vehicle` (  
  `id` INT NOT NULL AUTO_INCREMENT,  
  `state_id` INT NOT NULL DEFAULT 1,  
  `type_id` INT NOT NULL,  
  `matriculation` VARCHAR(45) NOT NULL,  
  PRIMARY KEY (`id`),  
  CONSTRAINT `fk_vehicle_state1`  
    FOREIGN KEY (`state_id`)  
      REFERENCES `flashpizza`.`state` (`id`),  
  CONSTRAINT `fk_vehicle_vehicle_type1`  
    FOREIGN KEY (`type_id`)  
      REFERENCES `flashpizza`.`vehicle_type` (`id`))  
ENGINE = InnoDB  
DEFAULT CHARACTER SET = utf8mb3;  
  
CREATE INDEX `fk_vehicle_state1_idx` ON `flashpizza`.`vehicle` (`state_id` ASC) VISIBLE;  
  
CREATE INDEX `fk_vehicle_vehicle_type1_idx` ON `flashpizza`.`vehicle` (`type_id` ASC)  
VISIBLE;
```

```
-- Table `flashpizza`.`messenger`
```

```
CREATE TABLE IF NOT EXISTS `flashpizza`.`messenger` (  
  `id` INT NOT NULL AUTO_INCREMENT,  
  `state_id` INT NOT NULL DEFAULT '1',  
  `vehicle_id` INT NULL DEFAULT NULL,  
  `name` VARCHAR(45) NULL DEFAULT NULL,  
  `phone_number` VARCHAR(15) NOT NULL,  
  PRIMARY KEY (`id`),  
  CONSTRAINT `fk_messenger_state1`  
    FOREIGN KEY (`state_id`)  
      REFERENCES `flashpizza`.`state` (`id`),  
  CONSTRAINT `fk_vehicle_id1`  
    FOREIGN KEY (`vehicle_id`)  
      REFERENCES `flashpizza`.`vehicle` (`id`)  
      ON DELETE RESTRICT  
      ON UPDATE RESTRICT)  
ENGINE = InnoDB  
DEFAULT CHARACTER SET = utf8mb3;  
  
CREATE INDEX `fk_messenger_state1_idx` ON `flashpizza`.`messenger` (`state_id` ASC)  
VISIBLE;  
  
CREATE INDEX `fk_vehicle_id_idx` ON `flashpizza`.`messenger` (`vehicle_id` ASC) VISIBLE;
```

```
-- Table `flashpizza`.`order_state`
```

```
CREATE TABLE IF NOT EXISTS `flashpizza`.`order_state` (  
  `id` INT NOT NULL AUTO_INCREMENT,
```

```

    `name` VARCHAR(45) NOT NULL,
    PRIMARY KEY (`id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb3;

-----
-- Table `flashpizza`.`user`
-----
CREATE TABLE IF NOT EXISTS `flashpizza`.`user` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `firstname` VARCHAR(16) NOT NULL,
  `lastname` VARCHAR(16) NULL,
  `phone_number` VARCHAR(15) NOT NULL,
  `email` VARCHAR(255) NULL DEFAULT NULL,
  `password` VARCHAR(32) NOT NULL,
  `create_time` TIMESTAMP NULL DEFAULT CURRENT_TIMESTAMP,
  `balance` DECIMAL(18,2) NOT NULL DEFAULT '0.00',
  PRIMARY KEY (`id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb3;

-----
-- Table `flashpizza`.`order`
-----
CREATE TABLE IF NOT EXISTS `flashpizza`.`order` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `user_id` INT NOT NULL,
  `state_id` INT NOT NULL DEFAULT '1',
  `messenger_id` INT NULL DEFAULT NULL,
  `vehicle_id` INT NULL DEFAULT NULL,
  `order_date` TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `delivery_date` TIMESTAMP NULL DEFAULT NULL,
  `price` DECIMAL(5,2) NOT NULL DEFAULT '0.00',
  `late` TINYINT NULL DEFAULT 0,
  PRIMARY KEY (`id`),
  CONSTRAINT `fk_order_messenger1`
    FOREIGN KEY (`messenger_id`)
      REFERENCES `flashpizza`.`messenger` (`id`),
  CONSTRAINT `fk_order_order_state1`
    FOREIGN KEY (`state_id`)
      REFERENCES `flashpizza`.`order_state` (`id`),
  CONSTRAINT `fk_order_user`
    FOREIGN KEY (`user_id`)
      REFERENCES `flashpizza`.`user` (`id`),
  CONSTRAINT `fk_order_vehicle1`
    FOREIGN KEY (`vehicle_id`)
      REFERENCES `flashpizza`.`vehicle` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb3;

CREATE INDEX `fk_order_user_idx` ON `flashpizza`.`order` (`user_id` ASC) VISIBLE;

```

```

CREATE INDEX `fk_order_order_state1_idx` ON `flashpizza`.`order` (`state_id` ASC)
VISIBLE;

CREATE INDEX `fk_order_messenger1_idx` ON `flashpizza`.`order` (`messenger_id` ASC)
VISIBLE;

CREATE INDEX `fk_order_vehicle1_idx` ON `flashpizza`.`order` (`vehicle_id` ASC) VISIBLE;

-----
-- Table `flashpizza`.`pizza`
-----
CREATE TABLE IF NOT EXISTS `flashpizza`.`pizza` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  `price` DECIMAL(4,2) NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb3;

-----
-- Table `flashpizza`.`pizza_size`
-----
CREATE TABLE IF NOT EXISTS `flashpizza`.`pizza_size` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  `coefficient` DECIMAL(3,2) NOT NULL DEFAULT '1.00',
  PRIMARY KEY (`id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb3;

-----
-- Table `flashpizza`.`order_line`
-----
CREATE TABLE IF NOT EXISTS `flashpizza`.`order_line` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `order_id` INT NOT NULL,
  `pizza_id` INT NOT NULL,
  `size_id` INT NOT NULL,
  `quantity` INT NOT NULL DEFAULT '1',
  `price` DECIMAL(5,2) NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `fk_order_line_order1`
    FOREIGN KEY (`order_id`)
    REFERENCES `flashpizza`.`order` (`id`),
  CONSTRAINT `fk_order_line_pizza1`
    FOREIGN KEY (`pizza_id`)
    REFERENCES `flashpizza`.`pizza` (`id`),
  CONSTRAINT `fk_order_line_pizza_size1`
    FOREIGN KEY (`size_id`)
    REFERENCES `flashpizza`.`pizza_size` (`id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb3;

```

```
CREATE INDEX `fk_order_line_order1_idx` ON `flashpizza`.`order_line` (`order_id` ASC)
VISIBLE;
```

```
CREATE INDEX `fk_order_line_pizza1_idx` ON `flashpizza`.`order_line` (`pizza_id` ASC)
VISIBLE;
```

```
CREATE INDEX `fk_order_line_pizza_size1_idx` ON `flashpizza`.`order_line` (`size_id`
ASC) VISIBLE;
```

```
-- -----
-- Table `flashpizza`.`pizza_ingredient`
-- -----
```

```
CREATE TABLE IF NOT EXISTS `flashpizza`.`pizza_ingredient` (
  `pizza_id` INT NOT NULL,
  `ingredient_id` INT NOT NULL,
  PRIMARY KEY (`pizza_id`, `ingredient_id`),
  CONSTRAINT `fk_pizza_ingredient_ingredient1`
    FOREIGN KEY (`ingredient_id`)
      REFERENCES `flashpizza`.`ingredient` (`id`),
  CONSTRAINT `fk_pizza_ingredient_pizza1`
    FOREIGN KEY (`pizza_id`)
      REFERENCES `flashpizza`.`pizza` (`id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb3;
```

```
CREATE INDEX `fk_pizza_ingredient_pizza1_idx` ON `flashpizza`.`pizza_ingredient`
(`pizza_id` ASC) VISIBLE;
```

```
CREATE INDEX `fk_pizza_ingredient_ingredient1_idx` ON `flashpizza`.`pizza_ingredient`
(`ingredient_id` ASC) VISIBLE;
```

```
-- -----
-- Table `flashpizza`.`transaction`
-- -----
```

```
CREATE TABLE IF NOT EXISTS `flashpizza`.`transaction` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `user_id` INT NOT NULL,
  `amount` DECIMAL(18,2) NOT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `fk_transaction_user1`
    FOREIGN KEY (`user_id`)
      REFERENCES `flashpizza`.`user` (`id`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

```
CREATE INDEX `fk_transaction_user1_idx` ON `flashpizza`.`transaction` (`user_id` ASC)
VISIBLE;
```

```
USE `flashpizza` ;
```

```
-- -----
-- Placeholder table for view `flashpizza`.`available_messenger`
-- -----
```

```

CREATE TABLE IF NOT EXISTS `flashpizza`.`available_messenger` (`Nom` INT, `Status` INT,
`Téléphone` INT, `Vehicule` INT, `Type vehicule` INT);

-- -----
-- Placeholder table for view `flashpizza`.`menu`
-- -----
CREATE TABLE IF NOT EXISTS `flashpizza`.`menu` (`id` INT, `Pizza` INT, `Ingredient` INT,
`Price` INT);

-- -----
-- Placeholder table for view `flashpizza`.`messenger_status`
-- -----
CREATE TABLE IF NOT EXISTS `flashpizza`.`messenger_status` (`Nom` INT, `Status` INT,
`Téléphone` INT, `Vehicule` INT, `Type vehicule` INT);

-- -----
-- Placeholder table for view `flashpizza`.`pizza_price`
-- -----
CREATE TABLE IF NOT EXISTS `flashpizza`.`pizza_price` (`pizza_id` INT, `Pizza` INT,
`size_id` INT, `Taille` INT, `Price` INT);

-- -----
-- Placeholder table for view `flashpizza`.`revenue`
-- -----
CREATE TABLE IF NOT EXISTS `flashpizza`.`revenue` (`"revenue"` INT);

-- -----
-- Placeholder table for view `flashpizza`.`customer_expenses`
-- -----
CREATE TABLE IF NOT EXISTS `flashpizza`.`customer_expenses` (`firstname` INT,
`"expenses"` INT);

-- -----
-- Placeholder table for view `flashpizza`.`best_customer`
-- -----
CREATE TABLE IF NOT EXISTS `flashpizza`.`best_customer` (`id` INT);

-- -----
-- Placeholder table for view `flashpizza`.`available_vehicle`
-- -----
CREATE TABLE IF NOT EXISTS `flashpizza`.`available_vehicle` (`matriculation` INT,
`"type"` INT, `"state"` INT);

-- -----
-- Placeholder table for view `flashpizza`.`total_pizza_order`
-- -----
CREATE TABLE IF NOT EXISTS `flashpizza`.`total_pizza_order` (`"Pizza"` INT, `"Total
commandé"` INT);

-- -----
-- Placeholder table for view `flashpizza`.`best_pizza`
-- -----
CREATE TABLE IF NOT EXISTS `flashpizza`.`best_pizza` (`id` INT);

-- -----
-- Placeholder table for view `flashpizza`.`worst_pizza`

```



```

-----
CREATE TABLE IF NOT EXISTS `flashpizza`.`worst_pizza` (`id` INT);

-----
-- View `flashpizza`.`available_messenger`
-----

DROP TABLE IF EXISTS `flashpizza`.`available_messenger`;
USE `flashpizza`;
CREATE OR REPLACE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL SECURITY DEFINER
VIEW `available_messenger` AS SELECT `messenger_status`.`Nom` AS `Nom`,
`messenger_status`.`Status` AS `Status`, `messenger_status`.`Téléphone` AS `Téléphone`,
`messenger_status`.`Vehicule` AS `Vehicule`, `messenger_status`.`Type vehicule` AS `Type
vehicule` FROM `messenger_status` WHERE (`messenger_status`.`Status` = 'Disponible');

-----
-- View `flashpizza`.`menu`
-----

DROP TABLE IF EXISTS `flashpizza`.`menu`;
USE `flashpizza`;
CREATE OR REPLACE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL SECURITY DEFINER
VIEW `menu` AS SELECT DISTINCT `p`.`id` AS `id`, `p`.`name` AS `Pizza`, `i`.`name` AS
`Ingredient`, `p`.`price` AS `Price` FROM ((`pizza` `p` join `pizza_ingredient` `pi`
on((`pi`.`pizza_id` = `p`.`id`))) join `ingredient` `i` on((`pi`.`ingredient_id` =
`i`.`id`)));

-----
-- View `flashpizza`.`messenger_status`
-----

DROP TABLE IF EXISTS `flashpizza`.`messenger_status`;
USE `flashpizza`;
CREATE OR REPLACE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL SECURITY DEFINER
VIEW `messenger_status` AS SELECT `m`.`name` AS `Nom`, `s`.`Name` AS `Status`,
`m`.`phone_number` AS `Téléphone`, `v`.`matriculation` AS `Vehicule`, `vt`.`name` AS
`Type vehicule` FROM (((`messenger` `m` join `state` `s` on((`s`.`id` =
`m`.`state_id`))) left join `vehicle` `v` on((`v`.`id` = `m`.`vehicle_id`))) left join
`vehicle_type` `vt` on((`vt`.`id` = `v`.`type_id`)));

-----
-- View `flashpizza`.`pizza_price`
-----

DROP TABLE IF EXISTS `flashpizza`.`pizza_price`;
USE `flashpizza`;
CREATE OR REPLACE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL SECURITY DEFINER
VIEW `pizza_price` AS SELECT `p`.`id` AS `pizza_id`, `p`.`name` AS `Pizza`, `s`.`id` AS
`size_id`, `s`.`name` AS `Taille`, (`s`.`coefficient` * `p`.`price`) AS `Price` FROM
(`pizza` `p` join `pizza_size` `s`) ORDER BY `p`.`id` ASC, (`s`.`coefficient` *
`p`.`price`) ASC;

-----
-- View `flashpizza`.`revenue`
-----

DROP TABLE IF EXISTS `flashpizza`.`revenue`;
USE `flashpizza`;
CREATE OR REPLACE VIEW revenue as
      SELECT SUM(price) as "revenue"
      FROM `order`

```

```

WHERE state_id > 0;

-----
-- View `flashpizza`.`customer_expenses`
-----
DROP TABLE IF EXISTS `flashpizza`.`customer_expenses`;
USE `flashpizza`;
CREATE OR REPLACE VIEW customer_expenses as
    SELECT u.firstname, SUM(o.price) as "expenses"
    FROM `order` o
    JOIN user u on u.id = o.user_id
    GROUP BY user_id;

-----
-- View `flashpizza`.`best_customer`
-----
DROP TABLE IF EXISTS `flashpizza`.`best_customer`;
USE `flashpizza`;
CREATE OR REPLACE VIEW best_customer as
    SELECT *
    FROM customer_expenses as c
    WHERE c.expenses = (SELECT Max(expenses) FROM customer_expenses);

-----
-- View `flashpizza`.`available_vehicle`
-----
DROP TABLE IF EXISTS `flashpizza`.`available_vehicle`;
USE `flashpizza`;
CREATE OR REPLACE VIEW `available_vehicle` AS
select v.matriculation, t.Name "type", s.Name "state"
From vehicle v
JOIN state s on s.id = v.state_id
JOIN vehicle_type t on t.id = v.type_id
where state_id = 2;

-----
-- View `flashpizza`.`total_pizza_order`
-----
DROP TABLE IF EXISTS `flashpizza`.`total_pizza_order`;
USE `flashpizza`;
CREATE OR REPLACE VIEW `total_pizza_order` AS
SELECT p.name "Pizza", SUM(ol.quantity) "Total commandé"
FROM flashpizza.order_line ol
JOIN flashpizza.pizza p on p.id = ol.pizza_id
GROUP BY p.id;

-----
-- View `flashpizza`.`best_pizza`
-----
DROP TABLE IF EXISTS `flashpizza`.`best_pizza`;
USE `flashpizza`;
CREATE OR REPLACE VIEW `best_pizza` AS
SELECT *
FROM total_pizza_order
WHERE `total ordered` = (SELECT MAX(`total ordered`) from total_pizza_order);

```

```

-- -----
-- View `flashpizza`.`worst_pizza`
-- -----

DROP TABLE IF EXISTS `flashpizza`.`worst_pizza`;
USE `flashpizza`;
CREATE OR REPLACE VIEW `worst_pizza` AS
SELECT *
FROM total_pizza_order
WHERE `total ordered` = (SELECT MIN(`total ordered`) from total_pizza_order);
USE `flashpizza`;

DELIMITER $$
USE `flashpizza`$$
CREATE DEFINER = CURRENT_USER TRIGGER `flashpizza`.`messenger_AFTER_UPDATE` AFTER UPDATE
ON `messenger` FOR EACH ROW
BEGIN
if not new.vehicle_id <=> old.vehicle_id
then
    if new.vehicle_id is not NULL
    then
        update vehicle
        set state_id = 1
        where id = new.vehicle_id;
    end if;

    if old.vehicle_id is not NULL
    then
        update vehicle
        set state_id = 2
        where id = old.vehicle_id;
    end if;
end if;
END$$

USE `flashpizza`$$
CREATE TRIGGER `before_update_order` BEFORE UPDATE ON `order` FOR EACH ROW BEGIN
if (SELECT user.balance FROM user WHERE user.id = NEW.user_id)< NEW.price AND
NEW.state_id = 2
then
    SIGNAL sqlstate '45001' set message_text = "Not enough money";
end if;

if NEW.messenger_id is NULL AND NEW.state_id = 5 then
    SIGNAL sqlstate '45001' set message_text = "Assign messenger before delivery";
end if;

if NEW.messenger_id is not NULL then
    set NEW.vehicle_id = (SELECT m.vehicle_id FROM messenger m WHERE m.id =
NEW.messenger_id);
end if;
END$$

USE `flashpizza`$$
CREATE TRIGGER `after_update_order` AFTER UPDATE ON `order` FOR EACH ROW BEGIN
if NEW.state_id = 2 THEN
    INSERT INTO `transaction` (user_id, amount) VALUES (NEW.user_id, -NEW.price);

```

```

        end if;

if NEW.state_id = 5    THEN
    UPDATE messenger set messenger.state_id = 1 WHERE messenger.id = NEW.messenger_id;
end if;

if NEW.state_id = 6    THEN
    UPDATE messenger set messenger.state_id = 2 WHERE messenger.id = NEW.messenger_id;
end if;
END$$

USE `flashpizza`$$
CREATE TRIGGER `before_insert_order_line` BEFORE INSERT ON `order_line` FOR EACH ROW SET
NEW.price =
(SELECT p.price from `pizza_price` as p
WHERE pizza_id = NEW.pizza_id AND size_id = NEW.size_id) * NEW.quantity$$

USE `flashpizza`$$
CREATE TRIGGER `after_insert_order_line` AFTER INSERT ON `order_line` FOR EACH ROW
UPDATE `order` as o
SET o.price = o.price + NEW.price
WHERE o.id = NEW.order_id$$

USE `flashpizza`$$
CREATE DEFINER = CURRENT_USER TRIGGER `flashpizza`.`transaction_AFTER_INSERT` AFTER
INSERT ON `transaction` FOR EACH ROW
BEGIN
    UPDATE `flashpizza`.`user`
    set balance = balance + NEW.amount
    WHERE id = NEW.user_id;
END$$

DELIMITER ;

SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;

```

## D. Script d'insertion des données

```

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_
DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';
USE `flashpizza` ;

--
-- Déchargement des données de la table `state`
--
INSERT INTO `state` (`Name`) VALUES
('Non disponible'),
('Disponible');
--

```

```

-- Déchargement des données de la table `pizza_size`
--
INSERT INTO `pizza_size` (`name`, `coefficient`) VALUES
('Humaine', '1.00'),
('Naine', '0.75'),
('Ogresse', '1.25');
--
-- Déchargement des données de la table `user`
--
INSERT INTO `user` (`firstname`, `phone_number`, `email`, `password`) VALUES
('Antonin', '+33641941449', 'anto@gmail.com', 'pwd'),
('Raj Porus', '+33699887766', 'raj@gmail.com', 'pwd');

INSERT INTO `transaction` (`user_id`, `amount`) VALUES
(2, 100),
(2, 45),
(1, 15);
--
-- Déchargement des données de la table `vehicle_type`
--
INSERT INTO `vehicle_type` (`name`) VALUES
('Car'),
('Motorcycle');
--
-- Déchargement des données de la table `vehicle`
--
INSERT INTO `vehicle` (`type_id`, `matriculation`) VALUES
(1, "AB-737-AT"),
(2, "AA-572-BF"),
(1, "AC-613-ET"),
(1, "AA-229-AA");
--
-- Déchargement des données de la table `pizza`
--
INSERT INTO `pizza` (`name`, `price`) VALUES
('Queen', '15.30'),
('Végétarienne', '15.00'),
('Regina', '9.95'),
('Hawaïenne', '17.30'),
('4 Fromages', '17.30'),
('Campagnarde', '17.30');
--
-- Déchargement des données de la table `ingredient`
--
INSERT INTO `ingredient` (`name`) VALUES
('Sauce Tomate'),
('Mozzarella'),
('Champignon'),
('Jambon'),
('Oignon'),
('Poivron'),
('Olive'),

```

```

('Roquette'),
('Fromage de chèvre'),
('Emmental'),
('Poulet'),
('Ananas'),
('Lardon'),
('Tomate fraîche'),
('Crème fraîche');

--
-- Déchargement des données de la table `pizza_ingredient`
--
INSERT INTO `pizza_ingredient` (`pizza_id`, `ingredient_id`) VALUES
(1, 1),
(1, 2),
(1, 3),
(1, 4),
(2, 1),
(2, 2),
(2, 3),
(2, 5),
(2, 6),
(3, 1),
(3, 2),
(3, 3),
(3, 4),
(3, 5),
(3, 7),
(3, 8),
(4, 1),
(4, 2),
(4, 11),
(4, 12),
(5, 1),
(5, 2),
(5, 9),
(5, 10),
(6, 2),
(6, 13),
(6, 15);

--
-- Déchargement des données de la table `messenger`
--
INSERT INTO `messenger` (`state_id`, `vehicle_id`, `name`, `phone_number`) VALUES
(2, NULL, 'Jean Pierre', '+33611223344'),
(2, NULL, 'Arnaud', '+33611251344'),
(1, NULL, 'Mehdi', '+33695371344'),
(1, NULL, 'Mamadou', '+3361612374'),
(2, NULL, 'Rick', '+3361612361');

--
-- Déchargement des données de la table `order_state`
--
INSERT INTO `order_state` (`name`) VALUES
('Basket'),
('Ordered'),
('Cooking'),

```

```

('Ready'),
('Delivering'),
('Delivered');
--
-- Déchargement des données de la table `order`
--
INSERT INTO `order` (`user_id`, `state_id`, `messenger_id`) VALUES
(1, 1, 1);
--
-- Déchargement des données de la table `order_line`
--
INSERT INTO `order_line` (`order_id`, `pizza_id`, `size_id`, `quantity`) VALUES
(1, 1, 1, 4),
(1, 2, 2, 1);

```

## II. Interrogation de la base de données

### A. La vue Menu

```

SELECT p.name as "Pizza", p.price as "Prix", i.name as "Ingredient"
FROM flashpizza.pizza p
JOIN flashpizza.pizza_ingredient pi on pi.pizza_id = p.id
JOIN flashpizza.ingredient i on i.id = pi.ingredient_id

```

### B. La vue Fiche de livraison

```

SELECT m.name as "Livreur", vt.name as "Type vehicule", u.username as "Nom
client", o.order_date as "Date de commande", p.name as "Pizza", ol.quantity as
"Quantité", ol.price as "Prix", o.price as "Prix total"
FROM flashpizza.order o
LEFT JOIN flashpizza.messenger m on m.id = o.messenger_id
LEFT JOIN flashpizza.user u on u.id = o.user_id
LEFT JOIN flashpizza.vehicle v on v.id = m.vehicle_id
LEFT JOIN flashpizza.vehicle_type vt on vt.id = v.type_id
LEFT JOIN flashpizza.order_line ol on ol.order_id = o.id
LEFT JOIN flashpizza.pizza p on p.id = ol.pizza_id

```

### C. Question diverses

- Calcul du nombre de commandes par client ?

```

SELECT u.username as "Nom client", COUNT(*) as "Nombre de commandes",
SUM(o.price) as "Total dépensé"
FROM flashpizza.order o
JOIN flashpizza.user u on u.id = o.user_id
GROUP BY o.user_id;

```

- Calcul de la moyenne des commandes ?

```

SELECT AVG(o.price) as "Moyenne des commandes"

```

```
FROM flashpizza.order o
```

- Extraction des clients ayant commandé plus que la moyenne ?

```
SELECT u.username as "Nom client", COUNT(*) as "Nombre de commandes",  
SUM(o.price) as "Total dépensé"  
FROM flashpizza.order o  
JOIN flashpizza.user u on u.id = o.user_id  
GROUP BY o.user_id  
HAVING SUM(o.price) >  
(SELECT AVG(o.price) as "Moyenne des commandes"  
FROM flashpizza.order o);
```

- Quel est le meilleur client ?

```
SELECT *  
FROM customer_expanses as c  
WHERE c.expanses = (SELECT Max(expanses) FROM customer_expanses)
```

- Quel est le chiffre d'affaires ?

```
SELECT SUM(price) as "revenue"  
FROM `order`  
WHERE state_id > 0
```

- Quel est le plus mauvais livreur ?

```
SELECT id, name, MAX(`late_count`) late_count  
FROM messenger_late_count  
GROUP BY id
```

- Quelle est la meilleure/pire pizza ?

```
SELECT *  
FROM total_pizza_order  
WHERE total_ordered = (SELECT MAX(`total_ordered`) FROM total_pizza_order);
```

```
SELECT *  
FROM total_pizza_order  
WHERE total_ordered = (SELECT MIN(`total_ordered`) FROM total_pizza_order);
```