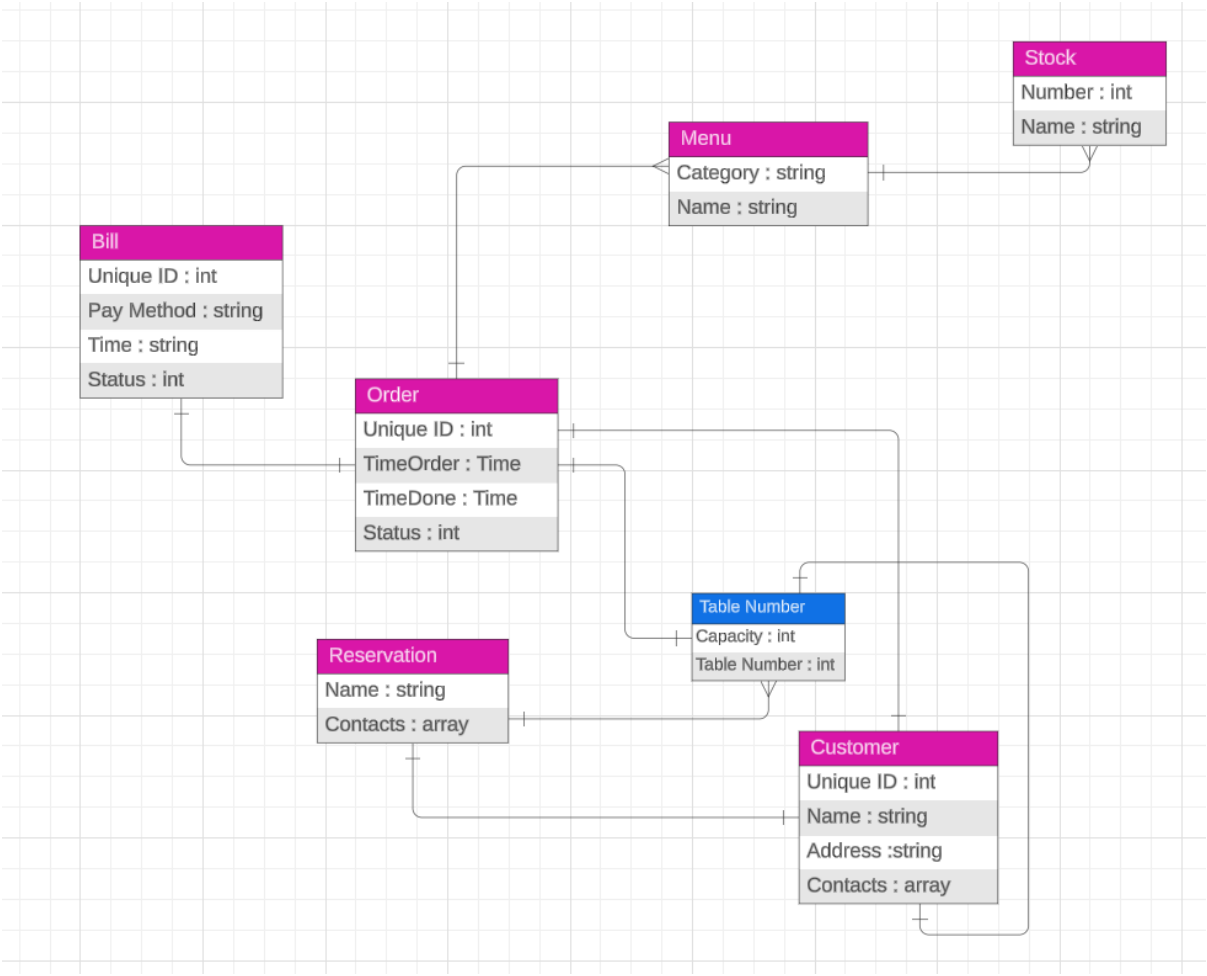
Nama : Jhon Samuel Kudadiri
NIM : 22/503772/TK/55066

Database Technology : Assignment 1

I. Management Rules

1. A customer is characterized by:
    - Unique ID
    - Name
    - Address
    - Contacts
2. A menu (per cuisine) is characterized by:
    - Category (main course, desert, beverage, etc.)
    - Name
3. An order is characterized by:
    - Unique ID
    - TimeOrder
    - TimeDone
    - Status
4. A bill is characterized by:
    - Unique ID
    - Pay method
    - Time
    - Status
5. A reservation is characterized by:
    - Name
    - Contacts
6. Stock is characterized by
    - (connected to menu)
    - Number of stock(s)

II. Conceptual Database Diagram



Soal :

Melanjutkan dan menggunakan hasil dari tugas sebelumnya, buatlah model logikanya dahulu, lalu buatlah database dan tabel-tabelnya dengan sintak DDL SQL :

1. Isilah data di tiap tabel masing masing 5 baris. Tulis SQL dan hasilnya

2. Buatlah 5 query untuk menampilkan data yang ada pada masing-masing tabel (harus berbeda variasi masing-masing query, dan hanya melibatkan 1 tabel saja). Tulis query dan hasil dari querynya (Query dengan pengkondisian, Query dengan pengelompokkan, Query dengan pengkondisian pengelompokkan, Query dengan aggregat function, Query dengan pengurutan)

3. Buatlah 5 query menggunakan metode subquery yang melibatkan 2 tabel. Tulis query dan hasil dari querynya

4. Buatlah 5 query menggunakan metode join tabel yang melibatkan 2 tabel. Tulis query dan hasil dari querynya

III. Logical-Relational DIagram

Customer

id	name	address	contacts

Reservation

Name	contacts	Table_number	Customer_id

Table

number	capacity	Order_id	Reservation_name	Customer_id

Stock

Id	number	string	Menu_name

Menu

category	**name**	Order_id

Order

Id	Timeorder	timedone	status	Customer_id

Bill

id	Pay_method	time	status	Order_id

IV. Creating database and table

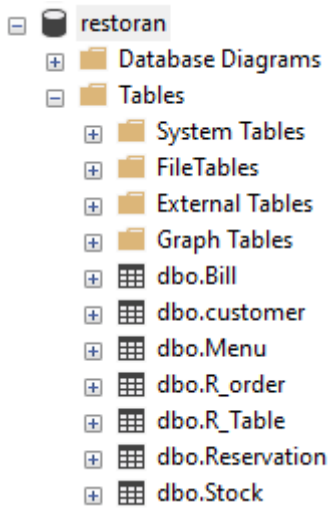
Query :

```
create database restoran;
use restoran;


create table customer(
	id int PRIMARY KEY,
	name varchar(255) NOT NULL,
	address text,
	contacts varchar(255)
);
create table R_order(
	id int UNIQUE,
	time_order DATETIME NOT NULL,
	time_done DATETIME,
	status_of_order varchar(255),
	customer_id int,
	FOREIGN KEY (customer_id) REFERENCES Customer(id)
 );
 CREATE TABLE Reservation (
  name VARCHAR(255) PRIMARY KEY,
  contacts VARCHAR(255),
  table_number INT,
  customer_id INT,
  FOREIGN KEY (Customer_id) REFERENCES Customer(id)
);
CREATE TABLE R_Table (
  number INT PRIMARY KEY ,
  capacity INT,
  order_id INT,
  reservation_name VARCHAR(255),
  customer_id INT,
  FOREIGN KEY (Order_id) REFERENCES R_order(Id),
  FOREIGN KEY (Customer_id) REFERENCES Customer(id)
);
CREATE TABLE Menu (
  name VARCHAR(255) PRIMARY KEY ,
  category VARCHAR(255),
  order_id INT,
  FOREIGN KEY (Order_id) REFERENCES R_Order(Id)
);
CREATE TABLE Stock (
  id INT PRIMARY KEY,
  number INT NOT NULL,
  item_type VARCHAR(255),  -- Changed 'string' to 'item_type' for clarity
  Menu_name VARCHAR(255),
  FOREIGN KEY (Menu_name) REFERENCES Menu(name)
);
CREATE TABLE Bill(
	id INT PRIMARY KEY,
	pay_method varchar(255) NOT NULL,
	time DATETIME,
	status varchar(10) check(status IN ('paid','not paid')),
	order_id int,
	FOREIGN KEY(order_id) references R_order(id),

	);
```

HASIL :



V. Inserting data into the tables

```
INSERT INTO Customer (id, name, address, contacts)
VALUES (1, 'John Doe', '123 Main St', '555-123-4567'),
       (2, 'Jane Smith', '456 Elm St', '555-789-0123'),
       (3, 'Michael Lee', '789 Oak Ave', '555-456-7890'),
       (4, 'Sarah Jones', '1011 Pine Blvd', '555-098-7654'),
       (5, 'David Miller', '1213 Spruce Ln', '555-321-9087');
```

Hasil:

	id	name	address	contacts
1	1	John Doe	123 Main St	555-123-4567
2	2	Jane Smith	456 Elm St	555-789-0123
3	3	Michael Lee	789 Oak Ave	555-456-7890
4	4	Sarah Jones	1011 Pine Blvd	555-098-7654
5	5	David Miller	1213 Spruce Ln	555-321-9087

```
INSERT INTO R_order (id, time_order, time_done, customer_id)
VALUES (1, '2024-04-29 20:13','2024-04-29 22:13', 'pending', 1),
       (2, '2024-04-29 20:13','2024-04-29 22:13', 'completed', 2),
       (3, '2024-04-29 20:13','2024-04-29 22:13', 'in progress', 3),
       (4, '2024-04-29 20:13','2024-04-29 22:13', 'cancelled', 4),
       (5, '2024-04-29 20:13','2024-04-29 22:13', 'pending', 5);
```

Hasil:

	id	time_order	time_done	status_of_order	customer_id
1	1	2024-04-29 20:13:00.000	2024-04-29 22:13:00.000	pending	1
2	2	2024-04-29 20:13:00.000	2024-04-29 22:13:00.000	completed	2
3	3	2024-04-29 20:13:00.000	2024-04-29 22:13:00.000	in progress	3
4	4	2024-04-29 20:13:00.000	2024-04-29 22:13:00.000	cancelled	4
5	5	2024-04-29 20:13:00.000	2024-04-29 22:13:00.000	pending	5

```
INSERT INTO Reservation (name, contacts, table_number, customer_id)
VALUES ('John Doe Reservation', '555-123-4567', 1, 1),
       ('Jane Smith Reservation', '555-789-0123', 2, 2),
       ('Group Reservation', '555-456-7890', 3, 3),
       ('Walk-in Party', 'N/A', 4, 4),
       ('Surprise Dinner', '555-098-7654', 5, 5);
```

Hasil:

	name	contacts	table_number	customer_id
1	Group Reservation	555-456-7890	3	3
2	Jane Smith Reservation	555-789-0123	2	2
3	John Doe Reservation	555-123-4567	1	1
4	Surprise Dinner	555-098-7654	5	5
5	Walk-in Party	N/A	4	4

```
INSERT INTO R_Table (number, capacity, order_id, reservation_name, customer_id)
VALUES (1, 4, 1, 'John Doe Reservation', 1),
       (2, 6, 2, 'Jane Smith Reservation', 2),
       (3, 8, 3, 'Group Reservation', 3),
       (4, 2, 4, 'Walk-in Party', 4),
       (5, 4, 5, 'Surprise Dinner', 5);
```

Hasil:

	number	capacity	order_id	reservation_name	customer_id
1	1	4	1	John Doe Reservation	1
2	2	6	2	Jane Smith Reservation	2
3	3	8	3	Group Reservation	3
4	4	2	4	Walk-in Party	4
5	5	4	5	Surprise Dinner	5

```
INSERT INTO Menu (name, category, order_id)
VALUES ('Pizza Margherita', 'Italian', 1),
       ('Chicken Alfredo', 'Italian', 2),
       ('Beef Burger', 'American', 3),
       ('Salmon Salad', 'Healthy', 4),
       ('Chocolate Cake', 'Dessert', 5);
```

Hasil:

	name	category	order_id
1	Beef Burger	American	3
2	Chicken Alfredo	Italian	2
3	Chocolate Cake	Dessert	5
4	Pizza Margherita	Italian	1
5	Salmon Salad	Healthy	4

```
INSERT INTO Stock (number, item_type, Menu_name)
VALUES (10, 'Pizza Dough', 'Pizza Margherita'),
       (5, 'Chicken Breast', 'Chicken Alfredo'),
       (8, 'Beef Patties', 'Beef Burger'),
       (3, 'Salmon Fillets', 'Salmon Salad'),
       (5, 'Chocolate Cakes', 'Chocolate Cake');
```

Hasil:

	id	number	item_type	Menu_name
1	1	10	Pizza Dough	Pizza Margherita
2	2	5	Chicken Breast	Chicken Alfredo
3	3	8	Beef Patties	Beef Burger
4	4	3	Salmon Fillets	Salmon Salad
5	5	5	Chocolate Cakes	Chocolate Cake

```
INSERT INTO Bill (id, pay_method, time, status, order_id)
VALUES (1,'Cash', '2024-04-30 23:59', 'paid', 1);

INSERT INTO Bill (id, pay_method, time, status, order_id)
VALUES (2,'Credit Card', '2024-04-30 23:59', 'paid', 2);

INSERT INTO Bill (id, pay_method, time, status, order_id)
VALUES (3,'Debit Card', '2024-04-30 23:59', 'not paid', 3);

INSERT INTO Bill (id, pay_method, time, status, order_id)
VALUES (4,'Cash', '2024-04-30 23:59', 'not paid', 4);

INSERT INTO Bill (id, pay_method, time, status, order_id)
VALUES (5,'Gift Certificate', '2024-04-30 23:59', 'paid', 5);
```

Hasil:

	id	pay_method	time	status	order_id
1	1	Cash	2024-04-30 23:59:00.000	paid	1
2	2	Credit Card	2024-04-30 23:59:00.000	paid	2
3	3	Debit Card	2024-04-30 23:59:00.000	not paid	3
4	4	Cash	2024-04-30 23:59:00.000	not paid	4
5	5	Gift Certificate	2024-04-30 23:59:00.000	paid	5

VI. Query with conditions

```
select * from R_order
where status_of_order='pending'
```

Hasil :

	id	time_order	time_done	status_of_order	customer_id
1	1	2024-04-29 20:13:00.000	2024-04-29 22:13:00.000	pending	1
2	5	2024-04-29 20:13:00.000	2024-04-29 22:13:00.000	pending	5

VII. Query with groupings

```
select item_type, SUM(number) as jumlah
from Stock
group by item_type
```

Hasil:

	item_type	jumlah
1	Beef Patties	8
2	Chicken Breast	5
3	Chocolate Cakes	5
4	Pizza Dough	10
5	Salmon Fillets	3

VIII. Query with conditions and groupings

```
select reservation_name, sum(capacity) as capacity
from R_table
where capacity between 4 and 8
group by reservation_name;
```

Hasil:

	reservation_name	capacity
1	Group Reservation	8
2	Jane Smith Reservation	6
3	John Doe Reservation	4
4	Surprise Dinner	4

IX. Query with aggregate function

```
select sum(number) as total
from Stock
```

Hasil:

	total
1	31

X. Query with ordering

```
select * from Stock
order by number asc;
```

Hasil:

	id	number	item_type	Menu_name
1	4	3	Salmon Fillets	Salmon Salad
2	5	5	Chocolate Cakes	Chocolate Cake
3	2	5	Chicken Breast	Chicken Alfredo
4	3	8	Beef Patties	Beef Burger
5	1	10	Pizza Dough	Pizza Margherita

XI. 5 queries with subquery method involving 2 tables

- Find all customers with completed orders

```
SELECT c.name
FROM Customer c
WHERE EXISTS (
  SELECT 1
  FROM R_order o
  WHERE o.customer_id = c.id AND o.status_of_order = 'completed'
                    );
```

	name
1	Jane Smith

- List all tables with ongoing orders (not completed, cancelled, or pending)

```
SELECT rt.number, rt.capacity
FROM R_Table rt
WHERE EXISTS (
  SELECT 1
  FROM R_order o
  WHERE o.id = rt.order_id AND o.status_of_order NOT IN ('completed', 'cancelled',
'pending')
);
```

Hasil:

	number	capacity
1	3	8

- Find the customer name with bill ide more than 2

```
SELECT c.name
FROM Customer c
WHERE c.id IN (
  SELECT o.customer_id
  FROM R_order o
  INNER JOIN Bill b ON o.id = b.order_id
  GROUP BY o.customer_id
  HAVING SUM(b.id) > 2
);
```

Hasil:

	name
1	Michael Lee
2	Sarah Jones
3	David Miller

-
- List all menu items included in completed orders

```
SELECT m.name
FROM Menu m
WHERE EXISTS (
  SELECT 1
  FROM R_order o
  INNER JOIN Bill b ON o.id = b.order_id
  WHERE o.status_of_order = 'completed'
  AND b.status = 'paid'
  AND m.order_id = o.id

);
```

Hasil:

	name
1	Chicken Alfredo

- Find the customer names and total bill amounts for non-paid bills

```
SELECT rt.number
FROM R_Table rt
WHERE rt.order_id NOT IN (
  SELECT o.id
  FROM R_order o
  WHERE o.status_of_order IN ('completed', 'cancelled', 'pending')
);
```

Hasil:

	number
1	3

XII. 5 queries with join table method involving 2 tables

- List all menu items with their corresponding order details (order ID, status, customer name)

```
SELECT m.name AS menu_item, o.id AS order_id, o.status_of_order, c.name AS
customer_name
FROM Menu m
INNER JOIN R_order o ON m.order_id = o.id
INNER JOIN Customer c ON o.customer_id = c.id;
Hasil:
```

	menu_item	order_id	status_of_order	customer_name
1	Beef Burger	3	in progress	Michael Lee
2	Chicken Alfredo	2	completed	Jane Smith
3	Chocolate Cake	5	pending	David Miller
4	Pizza Margherita	1	pending	John Doe
5	Salmon Salad	4	cancelled	Sarah Jones

- Find tables with their current reservations (reservation name, customer contact)

```
SELECT rt.number AS table_number, rt.capacity, r.name AS reservation_name,
r.contacts
FROM R_Table rt
LEFT JOIN Reservation r ON rt.reservation_name = r.name;

Hasil:
```

	table_number	capacity	reservation_name	contacts
1	1	4	John Doe Reservation	555-123-4567
2	2	6	Jane Smith Reservation	555-789-0123
3	3	8	Group Reservation	555-456-7890
4	4	2	Walk-in Party	N/A
5	5	4	Surprise Dinner	555-098-7654

- List all paid bills with the corresponding order details (customer name, order time)

```
use restoran;
SELECT b.id AS bill_id, b.pay_method, b.time, c.name AS customer_name,
o.time_order AS order_time
FROM Bill b
INNER JOIN R_order o ON b.order_id = o.id
INNER JOIN Customer c ON o.customer_id = c.id
WHERE b.status = 'paid';

Hasil:
```

	bill_id	pay_method	time	customer_name	order_time
1	1	Cash	2024-04-30 23:59:00.000	John Doe	2024-04-29 20:13:00.000
2	2	Credit Card	2024-04-30 23:59:00.000	Jane Smith	2024-04-29 20:13:00.000
3	5	Gift Certificate	2024-04-30 23:59:00.000	David Miller	2024-04-29 20:13:00.000

- Find the total number of orders (completed, cancelled, pending) for each customer

```
SELECT c.name AS customer_name, COUNT(*) AS total_orders
FROM Customer c
INNER JOIN R_order o ON c.id = o.customer_id
GROUP BY c.name;
```

Hasil:

	customer_name	total_orders
1	David Miller	1
2	Jane Smith	1
3	John Doe	1
4	Michael Lee	1
5	Sarah Jones	1

- List all menu items, the time when the order is made and done, and the customer of that order

```
SELECT m.name AS menu_item, o.time_order AS time_order, o.time_done, c.name AS
customer_name
FROM Menu m
INNER JOIN R_order o ON m.order_id = o.id
INNER JOIN Customer c ON o.customer_id = c.id;
```

Hasil:

	menu_item	time_order	time_done	customer_name
1	Beef Burger	2024-04-29 20:13:00.000	2024-04-29 22:13:00.000	Michael Lee
2	Chicken Alfredo	2024-04-29 20:13:00.000	2024-04-29 22:13:00.000	Jane Smith
3	Chocolate Cake	2024-04-29 20:13:00.000	2024-04-29 22:13:00.000	David Miller
4	Pizza Margherita	2024-04-29 20:13:00.000	2024-04-29 22:13:00.000	John Doe
5	Salmon Salad	2024-04-29 20:13:00.000	2024-04-29 22:13:00.000	Sarah Jones