**Midterm Report**

**Introduction:**

An online shop database is a crucial component of any e-commerce website. It manages all the data related to the products, customers, and transactions involved in online shopping. The database consists of various tables that store the necessary information, and each table is interconnected through various relationships to ensure the smooth functioning of the system.

**Table Structure:**

The online shop database comprises 15 tables, including payment\_method, kaspi\_payment, cash\_payment, creditcard\_payment, client, client\_address, client\_payment, department, employees, suppliers, category, products, orders, delivery, and order\_details.

The **payment\_method** **table** stores the different payment methods used in the online shop. The **kaspi\_payment, cash\_payment, and creditcard\_payment tables** store the payment details of each payment method.

The **client table** stores the customer's information, and the **client\_address table** stores the address of the customers.

The **client\_payment** table stores the payment method used by each client.

The **department table** stores the department's information, and the **employees table** stores the employee's details working in the department.

The **suppliers table** stores the supplier's information, and the **category table** stores the category of the products.

The **products table** stores the product's information, and the orders table stores the details of each order.

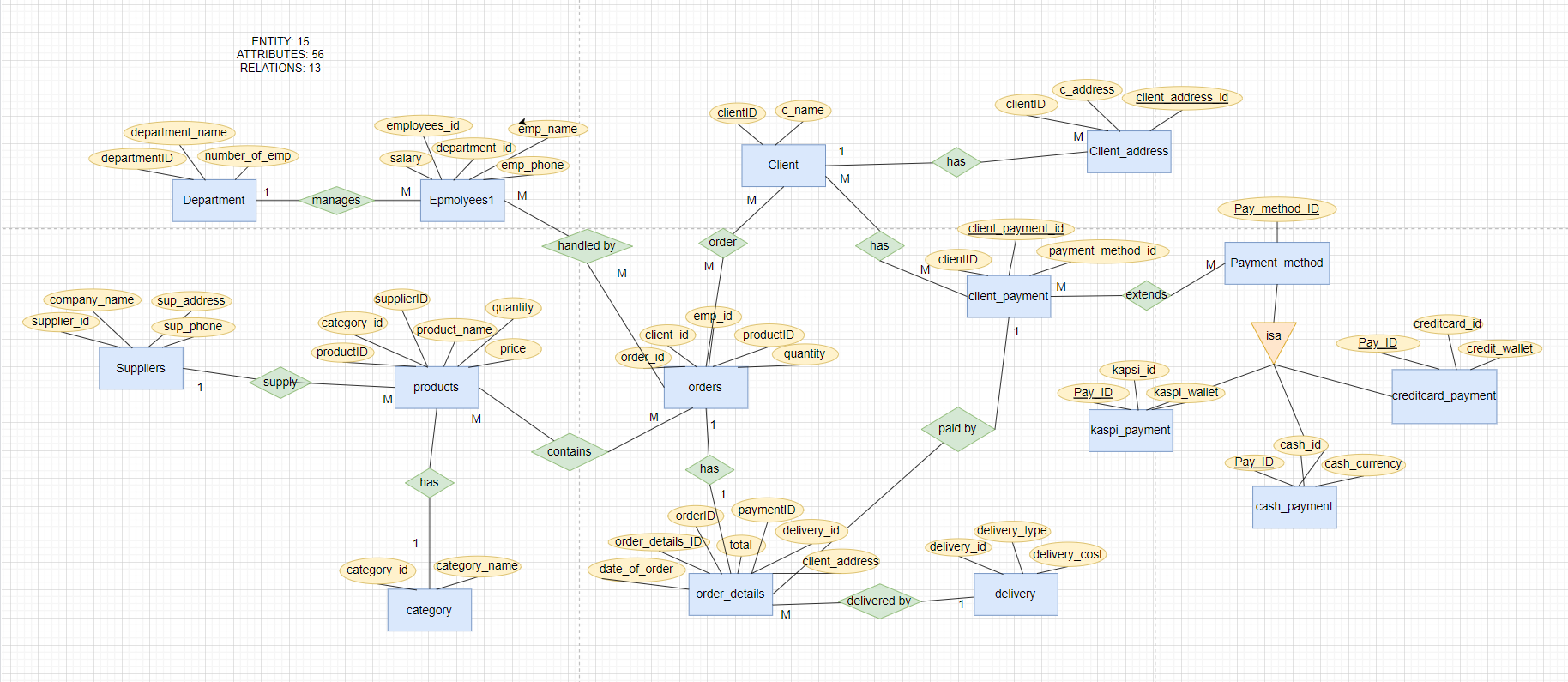
The **delivery table** stores the information related to the type and cost of the delivery.

The **order\_details table** stores the details of each order, including the payment, delivery, and client's address information.

Tables with data:

|  |
| --- |
|  |
|  | -- drop table payment\_method  -- create table payment\_method ( |
|  | -- payment\_method\_id int primary key |
|  | -- ) |
|  |  |
|  | -- insert into payment\_method (payment\_method\_id) values (1); |
|  | -- insert into payment\_method (payment\_method\_id) values (2) |
|  | -- insert into payment\_method (payment\_method\_id) values (3); |
|  | -- insert into payment\_method (payment\_method\_id) values (4); |
|  | -- insert into payment\_method (payment\_method\_id) values (5); |
|  | -- insert into payment\_method (payment\_method\_id) values (6); |
|  |  |
|  |  |
|  |  |
|  | -- create table kaspi\_payment( |
|  | -- pay\_id int primary key, |
|  | -- kaspi\_id int, |
|  | -- kaspi\_wallet int, |
|  | -- foreign key (pay\_id) references payment\_method (payment\_method\_id) |
|  | -- ) |
|  |  |
|  | -- insert into kaspi\_payment (pay\_id, kaspi\_id, kaspi\_wallet) values (1, 111, 123456789) |
|  | -- insert into kaspi\_payment (pay\_id, kaspi\_id, kaspi\_wallet) values (2, 222, 789456123) |
|  |  |
|  |  |
|  | -- drop table cash\_payment |
|  | -- create table cash\_payment( |
|  | -- pay\_id int primary key, |
|  | -- cash\_id int, |
|  | -- cash\_currency varchar(45), |
|  | -- foreign key (pay\_id) references payment\_method (payment\_method\_id) |
|  | -- ) |
|  |  |
|  | -- insert into cash\_payment (pay\_id, cash\_id, cash\_currency) values (3, 1001, 'tenge') |
|  | -- insert into cash\_payment (pay\_id, cash\_id, cash\_currency) values (4, 1002, 'dollars') |
|  |  |
|  |  |
|  |  |
|  | -- drop table creditcard\_payment |
|  | -- create table creditcard\_payment( |
|  | -- pay\_id int primary key, |
|  | -- creditcard\_id int, |
|  | -- credit\_wallet int, |
|  | -- foreign key (pay\_id) references payment\_method (payment\_method\_id) |
|  | -- ) |
|  |  |
|  | -- insert into creditcard\_payment (pay\_id, creditcard\_id, credit\_wallet) values (4, 11, 99999999) |
|  | -- insert into creditcard\_payment (pay\_id, creditcard\_id, credit\_wallet) values (5, 22, 77777777) |
|  |  |
|  |  |
|  |  |
|  | -- drop table client |
|  | -- create table client( |
|  | -- clientID int primary key, |
|  | -- c\_name varchar(45) |
|  | -- ) |
|  |  |
|  | -- insert into client (clientID, c\_name) values (1, 'Alex') |
|  | -- insert into client (clientID, c\_name) values (2, 'Olzhas') |
|  | -- insert into client (clientID, c\_name) values (3, 'Malika') |
|  | -- insert into client (clientID, c\_name) values (4, 'Polina') |
|  | -- insert into client (clientID, c\_name) values (5, 'Max') |
|  |  |
|  |  |
|  |  |
|  | -- drop table client\_address |
|  | -- create table client\_address ( |
|  | -- client\_address\_id int primary key, |
|  | -- clientID int, |
|  | -- c\_address varchar(45), |
|  | -- foreign key (clientID) references client (clientID) |
|  | -- ) |
|  |  |
|  | -- insert into client\_address (client\_address\_id, clientID, c\_address) values (1, 1, 'Akkent 45, 46') |
|  | -- insert into client\_address (client\_address\_id, clientID, c\_address) values (2, 1, 'Stepnoi 4, 24') |
|  | -- insert into client\_address (client\_address\_id, clientID, c\_address) values (3, 2, 'SDU') |
|  | -- insert into client\_address (client\_address\_id, clientID, c\_address) values (4, 3, 'MegaPark') |
|  | -- insert into client\_address (client\_address\_id, clientID, c\_address) values (5, 4, 'Aport') |
|  | -- insert into client\_address (client\_address\_id, clientID, c\_address) values (6, 5, 'Esentai mall') |
|  |  |
|  |  |
|  |  |
|  |  |
|  | -- drop table client\_payment |
|  | -- create table client\_payment ( |
|  | -- client\_payment\_id int primary key, |
|  | -- client\_id int, |
|  | -- payment\_method\_id int, |
|  | -- foreign key (payment\_method\_id) references payment\_method (payment\_method\_id), |
|  | -- foreign key (client\_id) references client (clientID) |
|  | -- ) |
|  |  |
|  | -- insert into client\_payment (client\_payment\_id, client\_id, payment\_method\_id) values (1, 1, 1) |
|  | -- insert into client\_payment (client\_payment\_id, client\_id, payment\_method\_id) values (2, 2, 2) |
|  | -- insert into client\_payment (client\_payment\_id, client\_id, payment\_method\_id) values (3, 3, 3) |
|  | -- insert into client\_payment (client\_payment\_id, client\_id, payment\_method\_id) values (4, 4, 4) |
|  | -- insert into client\_payment (client\_payment\_id, client\_id, payment\_method\_id) values (5, 5, 5) |
|  |  |
|  |  |
|  |  |
|  | -- drop table department; |
|  | -- create table department( |
|  | -- department\_id int primary key, |
|  | -- dept\_name varchar(45), |
|  | -- number\_of\_emp int |
|  | -- ) |
|  |  |
|  | -- insert into department (department\_id, dept\_name, number\_of\_emp) values (1, 'Manager', 3) |
|  | -- insert into department (department\_id, dept\_name, number\_of\_emp) values (2, 'Assistant', 1) |
|  | -- insert into department (department\_id, dept\_name, number\_of\_emp) values (3, 'IT manager', 1) |
|  | -- insert into department (department\_id, dept\_name, number\_of\_emp) values (4, 'HR', 1) |
|  |  |
|  |  |
|  |  |
|  | -- drop table employees; |
|  | -- create table employees( |
|  | -- employee\_id int primary key, |
|  | -- employee\_name varchar(45), |
|  | -- employee\_phone int, |
|  | -- salary int, |
|  | -- department\_id int, |
|  | -- foreign key (department\_id) references department(department\_id) |
|  | -- ) |
|  |  |
|  | -- insert into employees (employee\_id, employee\_name, employee\_phone, salary, department\_id) values (1, 'Sania', '87777777777', 150000, 1) |
|  | -- insert into employees (employee\_id, employee\_name, employee\_phone, salary, department\_id) values (2, 'Roma', '87477474747', 180000, 1) |
|  | -- insert into employees (employee\_id, employee\_name, employee\_phone, salary, department\_id) values (3, 'Erzhan', '87027020202', 170000, 1) |
|  | -- insert into employees (employee\_id, employee\_name, employee\_phone, salary, department\_id) values (4, 'Lola', '87017010101', 190000, 2) |
|  | -- insert into employees (employee\_id, employee\_name, employee\_phone, salary, department\_id) values (5, 'Aknur', '87057050505',210000, 3) |
|  | -- insert into employees (employee\_id, employee\_name, employee\_phone, salary, department\_id) values (6, 'Merey', '87717717171', 500000, 4) |
|  |  |
|  |  |
|  |  |
|  | -- create table suppliers( |
|  | -- supplier\_id int primary key, |
|  | -- company\_name varchar(45), |
|  | -- sup\_address varchar(45), |
|  | -- sup\_phone int |
|  | -- ) |
|  |  |
|  | -- insert into suppliers (supplier\_id, company\_name, sup\_address, sup\_phone) values (1, 'Alma-Ata company', 'Akkent 45,6', 87767767676) |
|  | -- insert into suppliers (supplier\_id, company\_name, sup\_address, sup\_phone) values (2, 'Qwerty company', 'Qwerty 221', 87012212121) |
|  | -- insert into suppliers (supplier\_id, company\_name, sup\_address, sup\_phone) values (3, 'Karaganda company', 'Gogolya 71', 87000000000) |
|  | -- insert into suppliers (supplier\_id, company\_name, sup\_address, sup\_phone) values (4, 'Shymkent company', 'Shym 789', 87789789789) |
|  | -- insert into suppliers (supplier\_id, company\_name, sup\_address, sup\_phone) values (5, 'Nur-Sultan company', 'Nazarbayev 99', 87777777171) |
|  |  |
|  |  |
|  | -- create table category( |
|  | -- category\_id int primary key, |
|  | -- category\_name varchar(45) |
|  | -- ) |
|  |  |
|  | -- insert into category (category\_id, category\_name) values (1, 'tables') |
|  | -- insert into category (category\_id, category\_name) values (2, 'chairs') |
|  | -- insert into category (category\_id, category\_name) values (3, 'sofa') |
|  | -- insert into category (category\_id, category\_name) values (4, 'closet') |
|  | -- insert into category (category\_id, category\_name) values (5, 'bed') |
|  |  |
|  | -- select \* from category |
|  |  |
|  |  |
|  | -- create table products( |
|  | -- product\_id int primary key, |
|  | -- category\_id int, |
|  | -- product\_name varchar(45), |
|  | -- quntity int, |
|  | -- price int, |
|  | -- supplier\_id int, |
|  | -- foreign key (category\_id) references category(category\_id), |
|  | -- foreign key (supplier\_id) references suppliers(supplier\_id) |
|  | -- ) |
|  |  |
|  | -- select \* from products |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (1, 1, 'Living room tables', 5, 120000, 1) |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (2, 1, 'Accent tables', 5, 120000, 1) |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (3, 1, 'Console tables', 3, 100000, 1) |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (4, 1, 'Kitchen tables', 10, 190000, 1) |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (5, 1, 'Work tables', 7, 140000, 1) |
|  |  |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (6, 2, 'Armchair', 3, 110000, 2) |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (7, 2, 'Recliner', 4, 180000, 2) |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (8, 2, 'Slipper chair', 5, 150000, 2) |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (9, 2, 'Bar stool chair', 15, 10000, 2) |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (10, 2, 'Side chair', 20, 30000, 2) |
|  |  |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (11, 3, 'Sectional sofa', 4, 200000, 3) |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (12, 3, 'Chesterfield sofa', 6, 195000, 3) |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (13, 3, 'English Roll-Arm sofa', 3, 225000, 3) |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (14, 3, 'Bridgewater sofa', 6, 230000, 3) |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (15, 3, 'Chaise Lounge sofa', 8, 210000, 3) |
|  |  |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (16, 4, 'Hinged closet door', 6, 195000, 4) |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (17, 4, 'Sliding coset door', 4, 290000, 4) |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (18, 4, 'Barn coset door', 2, 310000, 4) |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (19, 4, 'Flush coset door', 5, 350000, 4) |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (20, 4, 'Mirror coset door', 9, 400000, 4) |
|  |  |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (21, 5, 'Platform bed', 7, 200000, 5) |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (22, 5, 'Panel bed', 3, 275000, 5) |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (23, 5, 'Wall bed', 4, 280000, 5) |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (24, 5, 'Trundle bed', 6, 300000, 5) |
|  | -- insert into products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (25, 5, 'Adjustable bed', 5, 400000, 5) |
|  |  |
|  |  |
|  | -- drop table orders |
|  | -- create table orders( |
|  | -- order\_id int primary key, |
|  | -- client\_id int, |
|  | -- emp\_id int, |
|  | -- product\_id int, |
|  | -- quantity int, |
|  | -- foreign key (client\_id) references client(clientID), |
|  | -- foreign key (emp\_id) references employees(employee\_id), |
|  | -- foreign key (product\_id) references products(product\_id) |
|  | -- ) |
|  |  |
|  | -- insert into orders (order\_id, client\_id, emp\_id, product\_id, quantity) values (1, 1, 1, 1, 1) |
|  | -- insert into orders (order\_id, client\_id, emp\_id, product\_id, quantity) values (2, 2, 2, 21, 1) |
|  | -- insert into orders (order\_id, client\_id, emp\_id, product\_id, quantity) values (3, 3, 5, 15, 2) |
|  | -- insert into orders (order\_id, client\_id, emp\_id, product\_id, quantity) values (4, 4, 4, 19, 1) |
|  | -- insert into orders (order\_id, client\_id, emp\_id, product\_id, quantity) values (5, 1, 1, 8, 1) |
|  | -- insert into orders (order\_id, client\_id, emp\_id, product\_id, quantity) values (6, 1, 1, 7, 1) |
|  | -- insert into orders (order\_id, client\_id, emp\_id, product\_id, quantity) values (7, 2, 1, 4, 1) |
|  | -- insert into orders (order\_id, client\_id, emp\_id, product\_id, quantity) values (8, 4, 1, 20, 1) |
|  | -- insert into orders (order\_id, client\_id, emp\_id, product\_id, quantity) values (9, 5, 1, 18, 1) |
|  |  |
|  |  |
|  | -- drop table delivery |
|  | -- create table delivery ( |
|  | -- delivery\_id int primary key, |
|  | -- delivery\_type varchar(45), |
|  | -- delivery\_cost int |
|  | -- ) |
|  |  |
|  | -- insert into delivery (delivery\_id, delivery\_type, delivery\_cost) values (1, 'pickup', 0) |
|  | -- insert into delivery (delivery\_id, delivery\_type, delivery\_cost) values (2, 'delivery', 2000) |
|  |  |
|  |  |
|  | -- drop table order\_details |
|  | -- create table order\_details( |
|  | -- order\_details\_id int primary key, |
|  | -- order\_id int, |
|  | -- payment\_id int, |
|  | -- delivery\_id int, |
|  | -- client\_address\_id int, |
|  | -- total int, |
|  | -- date\_of\_order date, |
|  | -- foreign key (order\_id) references orders (order\_id), |
|  | -- foreign key (payment\_id) references client\_payment (client\_payment\_id), |
|  | -- foreign key (delivery\_id) references delivery (delivery\_id), |
|  | -- foreign key (client\_address\_id) references client\_address (client\_address\_id) |
|  | -- ) |
|  |  |
|  | -- insert into order\_details (order\_details\_id, order\_id, payment\_id, delivery\_id, client\_address\_id, total, date\_of\_order) values (1, 1, 1, 2, 1, 122000, SYSDATE) |
|  | -- insert into order\_details (order\_details\_id, order\_id, payment\_id, delivery\_id, client\_address\_id, total, date\_of\_order) values (2, 2, 2, 1, 2, 200000, SYSDATE) |
|  | -- insert into order\_details (order\_details\_id, order\_id, payment\_id, delivery\_id, client\_address\_id, total, date\_of\_order) values (3, 3, 3, 2, 3, 422000, SYSDATE) |
|  | -- insert into order\_details (order\_details\_id, order\_id, payment\_id, delivery\_id, client\_address\_id, total, date\_of\_order) values (4, 4, 4, 1, 4, 350000, SYSDATE) |
|  | -- insert into order\_details (order\_details\_id, order\_id, payment\_id, delivery\_id, client\_address\_id, total, date\_of\_order) values (5, 5, 1, 1, 1, 150000, SYSDATE) |
|  | -- insert into order\_details (order\_details\_id, order\_id, payment\_id, delivery\_id, client\_address\_id, total, date\_of\_order) values (6, 6, 1, 1, 1, 180000, SYSDATE) |
|  | -- insert into order\_details (order\_details\_id, order\_id, payment\_id, delivery\_id, client\_address\_id, total, date\_of\_order) values (7, 7, 2, 2, 3, 200000, SYSDATE) |
|  | -- insert into order\_details (order\_details\_id, order\_id, payment\_id, delivery\_id, client\_address\_id, total, date\_of\_order) values (8, 8, 4, 1, 5, 410000, SYSDATE) |
|  | -- insert into order\_details (order\_details\_id, order\_id, payment\_id, delivery\_id, client\_address\_id, total, date\_of\_order) values (9, 9, 5, 1, 6, 310000, SYSDATE) |
|  |  |
|  |  |
|  | -- select \* from order\_details |

**Relations:**



**One-to-one Relationships:**

* There is a one-to-one relationship between the **orders table** and the **order\_details table.** **One order can have only one order details, and one order detail can belong to only one order**.

**One-to-Many Relationship:**

* There is a one-to-many relationship between the **client table** and the **client\_address table**. **One client can have multiple address, but each address can belong to only one client.**
* There is a one-to-many relationship between the **department table** and the **employees table**. **One department can have multiple employees, but each employee can belong to only one department.**
* There is a one-to-many relationship between the **suppliers table** and the **products table**. **One supplier can supply multiple products, but each product can be supplied by only one supplier.**
* There is a one-to-many relationship between the **category table** and the **products table**. **One category can have multiple products, but each product can belong to only one supplier.**
* There is a one-to-many relationship between the **delivery table** and the **order\_details table**. **One delivery can can be used in multiple orders, but an order can only have one delivery method.**

**Many-to-Many Relationship:**

* There is a many-to-many relationship between the **client table** and the **payment\_method table** through the client\_payment table. **One client can have multiple payment methods, and one payment method can belong to multiple clients.**
* There is a many-to-many relationship between the **products table** and the **category table**. **One product can belong to multiple categories, and one category can have multiple products.**
* There is a many-to-many relationship between the **payment\_method table** and the **client\_payment table**. **A payment method can be used by many clients, and each client can use multiple payment methods.**
* There is a many-to-many relationship between the **Client, Employees, and Products tables. Since an order can have multiple clients, employees, and products, and a client, employee, or product can be associated with multiple orders.**

**Normal Forms:**

**1NF (First Normal Form):** A table is in 1NF if and only if all the columns contain atomic values (i.e., values cannot be further divided into smaller units) and there are no repeating groups or arrays of data in the table.

**2NF (Second Normal Form):** A table is in 2NF if it is in 1NF and every non-key column in the table is dependent on the whole primary key and not on just part of it.

**3NF (Third Normal Form):** A table is in 3NF if it is in 2NF and there are no transitive dependencies. A transitive dependency occurs when a non-key column depends on another non-key column.

**BCNF (Boyce-Codd Normal Form):** A table is in BCNF if it is in 3NF and for every functional dependency X → Y, X is a superkey (i.e., a unique identifier of a record) or Y is a part of a candidate key.

|  |  |
| --- | --- |
|  | |
|  | 1. Table 'payment\_method' has only one attribute, which is the primary key. Therefore, it is already in BCNF.  2. Table 'kaspi\_payment' has two attributes as its primary key (pay\_id and kaspi\_id), and no other attributes are dependent on these primary keys. Therefore, it is already in BCNF. |
|  | 3. Table 'cash\_payment' has two attributes as its primary key (pay\_id and cash\_id), and no other attributes are dependent on these primary keys. Therefore, it is already in BCNF. |
|  | 4. Table 'creditcard\_payment' has two attributes as its primary key (pay\_id and creditcard\_id), and no other attributes are dependent on these primary keys. Therefore, it is already in BCNF. |
|  | 5. Table 'client' has only one attribute as its primary key. Therefore, it is already in BCNF. |
|  | 6. Table 'client\_address' has two attributes as its primary key (client\_address\_id and clientID), and no other attributes are dependent on these primary keys. Therefore, it is already in BCNF. |
|  | 7. Table 'client\_payment' has three attributes as its primary key (client\_payment\_id, client\_id, and payment\_method\_id), and no other attributes are dependent on these primary keys. Therefore, it is already in BCNF. |
|  | 8. Table 'department' has only one attribute as its primary key. Therefore, it is already in BCNF. |
|  | 9. Table 'employees' has only one attribute as its primary key (employee\_id), and department\_id is dependent on this primary key. Since department\_id is a candidate key, this table is already in BCNF. |
|  | 10. Table 'suppliers' has only one attribute as its primary key. Therefore, it is already in BCNF. |
|  | 11. Table 'category' has only one attribute as its primary key. Therefore, it is already in BCNF. |
|  | 12. Table 'products' has two attributes as its primary key (product\_id and category\_id), and supplier\_id is dependent on product\_id. Since product\_id is a candidate key, this table is already in BCNF. |
|  | 13. Table 'orders' has only one attribute as its primary key (order\_id), and client\_id, emp\_id, and product\_id are dependent on this primary key. Since all of these attributes are candidate keys, this table is already in BCNF. |
|  | 14. Table 'delivery' has only one attribute as its primary key. Therefore, it is already in BCNF. |
|  | 15. Table 'order\_details' has only one attribute as its primary key (order\_details\_id), and order\_id, payment\_id, delivery\_id, and client\_address\_id are dependent on this primary key. Since all of these attributes are candidate keys, this table is already in BCNF. |
|  |  |

Based on the table structure and relationships we provided, it appears that all 15 tables are in BCNF (Boyce-Codd Normal Form) as they satisfy the requirements of BCNF, which includes that every determinant of a table must be a candidate key, and there must be no non-trivial functional dependencies between candidate keys.

**Therefore, all tables in the given database are already in BCNF.**

**Queries:**

1. **Procedure which does group by informations:**

|  |  |
| --- | --- |
|  | |
|  | | -- CREATE OR REPLACE PROCEDURE use\_payment AS  -- BEGIN | | | |
|  | | -- DECLARE | | | |
|  | | -- num\_kaspi\_clients NUMBER; | | | |
|  | | -- num\_cash\_clients NUMBER; | | | |
|  | | -- num\_creditcard\_clients NUMBER; | | | |
|  | | -- BEGIN | | | |
|  | | -- SELECT COUNT(\*) INTO num\_kaspi\_clients FROM client\_payment WHERE payment\_method\_id = 1; | | | |
|  | | -- SELECT COUNT(\*) INTO num\_cash\_clients FROM client\_payment WHERE payment\_method\_id = 2; | | | |
|  | | -- SELECT COUNT(\*) INTO num\_creditcard\_clients FROM client\_payment WHERE payment\_method\_id = 3; | | | |
|  | | -- DBMS\_OUTPUT.PUT\_LINE('Number of clients using Kaspi: ' || num\_kaspi\_clients); | | | |
|  | | -- DBMS\_OUTPUT.PUT\_LINE('Number of clients using cash: ' || num\_cash\_clients); | | | |
|  | | -- DBMS\_OUTPUT.PUT\_LINE('Number of clients using credit card: ' || num\_creditcard\_clients); | | | |
|  | | -- END; | | | |
|  | | -- END; | | | |
|  | |  | | | |
|  | | -- BEGIN | | | |
|  | | -- use\_payment; | | | |
|  | | -- END; | | | |
|  | |
|  | | -- CREATE OR REPLACE PROCEDURE made\_payment AS  -- BEGIN | |
|  | | -- DECLARE | |
|  | | -- num\_kaspi\_payments NUMBER; | |
|  | | -- num\_cash\_payments NUMBER; | |
|  | | -- num\_creditcard\_payments NUMBER; | |
|  | | -- BEGIN | |
|  | | -- SELECT COUNT(\*) INTO num\_kaspi\_payments FROM kaspi\_payment; | |
|  | | -- SELECT COUNT(\*) INTO num\_cash\_payments FROM cash\_payment; | |
|  | | -- SELECT COUNT(\*) INTO num\_creditcard\_payments FROM creditcard\_payment; | |
|  | | -- DBMS\_OUTPUT.PUT\_LINE('Number of payments made with Kaspi: ' || num\_kaspi\_payments); | |
|  | | -- DBMS\_OUTPUT.PUT\_LINE('Number of payments made with cash: ' || num\_cash\_payments); | |
|  | | -- DBMS\_OUTPUT.PUT\_LINE('Number of payments made with credit card: ' || num\_creditcard\_payments); | |
|  | | -- END; | |
|  | | -- END; | |
|  | |  | |
|  | | -- BEGIN | |
|  | | -- made\_payment; | |
|  | | -- END; | |

|  |
| --- |
|  |
|  | -- CREATE OR REPLACE PROCEDURE avg\_salary AS  -- BEGIN |
|  | -- DECLARE |
|  | -- manager\_avg\_salary NUMBER; |
|  | -- assistant\_avg\_salary NUMBER; |
|  | -- it\_manager\_avg\_salary NUMBER; |
|  | -- hr\_avg\_salary NUMBER; |
|  | -- BEGIN |
|  | -- SELECT AVG(salary) INTO manager\_avg\_salary FROM EMPLOYEES WHERE department\_id = 1; |
|  | -- SELECT AVG(salary) INTO assistant\_avg\_salary FROM EMPLOYEES WHERE department\_id = 2; |
|  | -- SELECT AVG(salary) INTO it\_manager\_avg\_salary FROM EMPLOYEES WHERE department\_id = 3; |
|  | -- SELECT AVG(salary) INTO hr\_avg\_salary FROM EMPLOYEES WHERE department\_id = 4; |
|  | -- DBMS\_OUTPUT.PUT\_LINE('Average salary for Managers: ' || manager\_avg\_salary); |
|  | -- DBMS\_OUTPUT.PUT\_LINE('Average salary for Assistants: ' || assistant\_avg\_salary); |
|  | -- DBMS\_OUTPUT.PUT\_LINE('Average salary for IT managers: ' || it\_manager\_avg\_salary); |
|  | -- DBMS\_OUTPUT.PUT\_LINE('Average salary for HR: ' || hr\_avg\_salary); |
|  | -- END; |
|  | -- END; |
|  |  |
|  | -- BEGIN |
|  | -- avg\_salary; |
|  | -- END; |

1. Function which counts the number of records

-- DROP FUNCTION count\_records

-- CREATE OR REPLACE FUNCTION count\_records

-- (table\_name IN VARCHAR2)

-- RETURN NUMBER

-- IS

-- total\_count NUMBER;

-- BEGIN

-- EXECUTE IMMEDIATE 'SELECT COUNT(\*) FROM ' || table\_name INTO total\_count;

-- RETURN total\_count;

-- END;

-- DECLARE

-- total\_count NUMBER;

-- BEGIN

-- total\_count := count\_records('employees');

-- DBMS\_OUTPUT.PUT\_LINE('Number of records in employees table: ' || total\_count);

-- END;

-- select \* from employees

The EXECUTE IMMEDIATE statement prepares an executable form of an SQL statement from a character string form of the statement and then executes the SQL statement.

|  |
| --- |
|  |
|  | -- DROP FUNCTION get\_count  -- CREATE OR REPLACE FUNCTION get\_count |
|  | -- RETURN NUMBER IS |
|  | -- total\_c number; |
|  | -- BEGIN |
|  | -- SELECT count(\*) INTO total\_c FROM employees; |
|  | -- RETURN total\_c; |
|  | -- END; |
|  |  |
|  | -- DECLARE |
|  | -- total\_count number; |
|  | -- BEGIN |
|  | -- total\_count := get\_count(); |
|  | -- DBMS\_OUTPUT.PUT\_LINE('Number of records in employees table: ' || total\_count); |
|  |  |
|  | -- END; |

|  |
| --- |
|  |
|  | -- DROP FUNCTION count\_orders\_by\_employee  -- CREATE OR REPLACE FUNCTION count\_orders\_by\_employee |
|  | -- RETURN SYS\_REFCURSOR AS |
|  | -- result\_set SYS\_REFCURSOR; |
|  | -- BEGIN |
|  | -- OPEN result\_set FOR |
|  | -- SELECT e.employee\_name, COUNT(\*) AS order\_count |
|  | -- FROM employees e |
|  | -- JOIN orders o ON e.employee\_id = o.emp\_id |
|  | -- GROUP BY e.employee\_name; |
|  | -- RETURN result\_set; |
|  | -- END; |
|  |  |
|  | -- DECLARE |
|  | -- result\_set SYS\_REFCURSOR; |
|  | -- employee\_name VARCHAR2(45); |
|  | -- order\_count NUMBER; |
|  | -- BEGIN |
|  | -- result\_set := count\_orders\_by\_employee; |
|  | -- LOOP |
|  | -- FETCH result\_set INTO employee\_name, order\_count; |
|  | -- EXIT WHEN result\_set%NOTFOUND; |
|  | -- DBMS\_OUTPUT.PUT\_LINE('Employee ' || employee\_name || ' has ' || order\_count || ' orders.'); |
|  | -- END LOOP; |
|  | -- END; |

1. Procedure which uses SQL%ROWCOUNT to determine the number of rows affected

|  |
| --- |
|  |
|  | -- CREATE OR REPLACE PROCEDURE update\_salary (  -- d\_id IN NUMBER, |
|  | -- salary\_increase IN NUMBER |
|  | -- ) |
|  | -- IS |
|  | -- BEGIN |
|  | -- UPDATE employees |
|  | -- SET salary = salary - salary\_increase |
|  | -- WHERE department\_id = d\_id; |
|  |  |
|  | -- IF SQL%ROWCOUNT > 0 THEN |
|  | -- DBMS\_OUTPUT.PUT\_LINE(SQL%ROWCOUNT || ' rows updated.'); |
|  | -- ELSE |
|  | -- DBMS\_OUTPUT.PUT\_LINE('No rows updated.'); |
|  | -- END IF; |
|  | -- END; |
|  |  |
|  | -- DECLARE |
|  | -- department\_id employees.department\_id%TYPE := 1; |
|  | -- salary number := 110000; |
|  | -- BEGIN |
|  | -- update\_salary(department\_id, salary); |
|  | -- END; |
|  |  |
|  | -- select \* from employees; |

1. Add user-defined exception which disallows to enter title of item (e.g. book) to be less than 5 characters

|  |
| --- |
|  |
|  | -- DROP TRIGGER trigger\_for\_client  -- CREATE OR REPLACE TRIGGER trigger\_for\_client |
|  | -- BEFORE INSERT ON client |
|  | -- FOR EACH ROW |
|  | -- DECLARE |
|  | -- count\_row number; |
|  | -- BEGIN |
|  | -- SELECT COUNT(\*) INTO count\_row FROM client; |
|  | -- DBMS\_OUTPUT.PUT\_LINE('Before inser = '||count\_row); |
|  | -- END; |
|  |  |
|  | -- insert into client values (111, 'Akerke'); |
|  | -- select \* from client |

1. Create a trigger before insert on any entity which will show the current number of rows in the table

|  |
| --- |
|  |
|  | DECLARE |
|  | category\_id products.category\_id%TYPE := 1; |
|  | product\_name products.product\_name%TYPE := :product\_name; |
|  | quantity products.quntity%TYPE := 1; |
|  | price products.QUNTITY%TYPE := :price; |
|  | supplier\_id products.supplier\_id%type := 1; |
|  | invalid\_data EXCEPTION; |
|  |  |
|  | BEGIN |
|  | IF length(product\_name) < 5 THEN |
|  | raise invalid\_data; |
|  | ELSE |
|  | INSERT INTO products (product\_id, category\_id, product\_name, quntity, price, supplier\_id) values (product\_id, category\_id, product\_name, quantity, price, supplier\_id); |
|  | END IF; |
|  |  |
|  | EXCEPTION |
|  | WHEN invalid\_data THEN |
|  | DBMS\_OUTPUT.PUT\_LINE('Invalid name entered.'); |
|  | WHEN others THEN |
|  | DBMS\_OUTPUT.PUT\_LINE('Error!'); |
|  | END; |