Custom Database Report:

Name: SM Ragib Rezwan

ID: 103172423

Content List:

  - Table Employee_position
  - Table Employee_and_Branch_details
  - Table Branch_details

- Scripts used to make the database

- Scripts used to insert data for all the tables

- Use of 5 joins in the database:

  1) Use Join to make a view with full employee details (including position and their branch names)

  2) Use Join to make a view for full product details (including subsidiary company name

  3) Use Join to make a view for order stating name of employee and product and customer buying it

  4) Use Join to find which product was ordered more than once

  5) Use Join to find what the weird customer had ordered in details:

- Reference

Overview of the database:

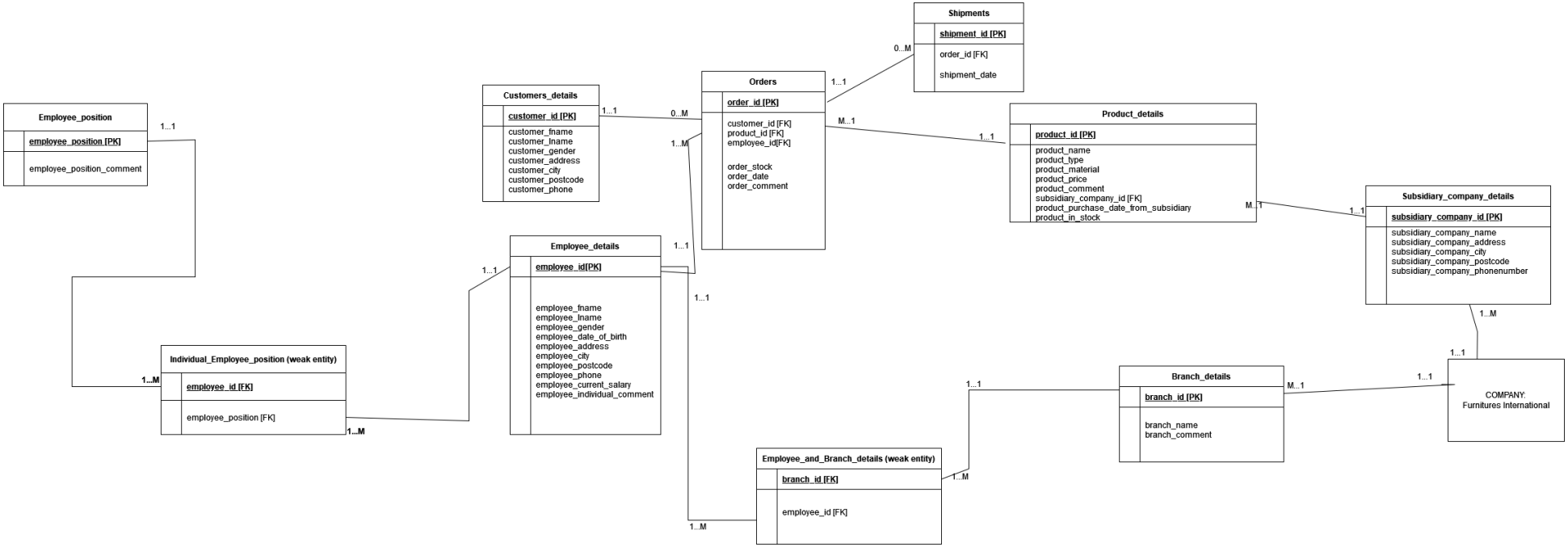
Background of company owning the database:

This is a database made for the Company called Furnitures International. Although it may seem like any other ordinary furniture selling shop, its current situation is rather serious. Previously they had been a big company, controlling a significant portion in the furniture market industry in Southeast Asia with branches in all relevant countries. But unfortunately, they had become a victim of a hostile takeover, losing almost all of their business and employees in the process. Furthermore, they were unable to cope with Covid-19 Crisis which led to further losses in personnel and assets. Now, they are currently trying to build up their business from scratch, trying to make the connections and gather employees using their last 2 branches and few subsidiary companies that they still have connection with.

Main use of the database:

This database will mainly be used to keep track of employee information, Customer information, product information, subsidiary company information, order details and shipping. Thus I used MYSQL here as it is highly structural, provides robust transactions following ACID(atomic, consistent, isolated, durable), etc. Furthermore, data can also be quickly updated and inserted into relevant tables via short queries by using MYSQL.

Simplified UML Diagram :



Tables, attributes, data types and the reasoning behind them:

a) Table Product

Attribute name	Datatype	Null or not	Justification
product_id	Varchar(10)	NOT NULL	The Id is a combination of letter and number to kept it varchar to save space and also allow alphanumeric input. Also all products must have an id so kept it not null
product_name	Varchar(20)	NOT NULL	The name is made of letter so kept it varchar to save space and also allow alphabet input. Also all products must have a name so kept it not null
product_type	Varchar(8)	NOT NULL	Fixed product type to only let chair, table, sofa, wardrobe as input. So kept it Varchar as letter input and also to save space. All products have a type so kept it as not null
product_material	Varchar(7)	NOT NULL	Fixed product type to only let wood, plastic, glass as input. So kept it Varchar as letter input and also to save space. All products are made of a material so kept it as not null
product_price	Int(6) UNISGNED	NOT NULL	Price of products sold by company is in int and wont go above 6 digits, so kept it as int(6). Also it cant be negative so unsigned. All products have a price, so kept it as not null
product_comment	Varchar(50)	NULL	This is the individual comment about the product so kept it varchar as letter and to save space. Product may or may not have a comment so kept it as null
subsidiary_company_id	Varchar(10)	NOT NULL	This is the id of the subsidiary company product was purchased from. So kept it as varchar for alphanumeric input and also to save space. All products have a subsidiary company id so kept it as not null

product_purchase_date_from_subsidiary	Date	NOT NULL	This is the date product was the last purchased from the subsidiary company. So kept it as date type. All products have a date when it had been last purchase from subsidiary company
Product_in_stock	Int (3)	NULL	Kept it int 3 as company currently don't have storage to store more than 999 product of a type. Also made it Nullable as some products may be newly introduced and thus they may not have stock for it yet. (alternatively, it can be kept as not null, but then 0 has to be kept as default for the new product introduction case)

Primary key name: product_id

Foreign key names: subsidiary_company_id

b) Table subsidiary_company_details

Attribute name	Datatype	Null or not	Justification
Subsidiary_company_id	Varchar(10)	NOT NULL	The Id is a combination of letter and number to kept it varchar to save space and also allow alphanumeric input. Also all company must have an id so kept it not null
Subsidiary_company_name	Varchar(20)	NOT NULL	The name is made of letter so kept it varchar to save space and also allow alphabet input. Also all company must have a name so kept it not null
Subsidiary_company_address	Varchar(40)	NOT NULL	Address is made of letter and numbers, so Varchar as alphanumeric input and also to save space. All companies have their own addresses and so not null

Subsidiary_company_city	Varchar(20)	NOT NULL	City is made of letter, so Varchar as letter input and also to save space. All companies have their own city and so not null
Subsidiary_company_postcode	Int(4) UNSIGNED	NOT NULL	PostCode is made of numbers, so int as number input and restricted it to 4 digits as postcode only have 4 digits. All companies have their own postcode and so not null
Subsidiary_company_phonenumber	Bigint(13) UNISGNED	NOT NULL	Phonenumber is made of numbers, so made it bigint as number input and 13 digits. Didn't use int as it can only take from 0 to 4294967295 when unsigned which is not enough! All companies have their own number and so not null

Primary key name: subsidiary_company_id

c) Table shipments

Attribute name	Datatype	Null or not	Justification
shipment_id	Varchar(10)	NOT NULL	The Id is a combination of letter and number to kept it varchar to save space and also allow alphanumeric input. Also all shipment must have an id so kept it not null
Order_id	Varchar(10)	NOT NULL	The Id is a combination of letter and number to kept it varchar to save space and also allow alphanumeric input. Also all shipment must have an order_id to correspond to it, so kept it not null
Shipment_date	Date	NOT NULL	Ship date is bsacially a date so kept it as date type. All shipments have their own shipping date to kept it not null

Primary key name: shipment_id

Foreign key names: order_id

d) Table Orders

Attribute name	Datatype	Null or not	Justification
order_id	Varchar(10)	NOT NULL	The Id is a combination of letter and number to kept it varchar to save space and also allow alphanumeric input. Also all orders must have an order id so kept it not null
customer_id	Varchar(10)	NOT NULL	The Id is a combination of letter and number to kept it varchar to save space and also allow alphanumeric input. Also all orders must have an customer id so kept it not null
product_id	Varchar(10)	NOT NULL	The Id is a combination of letter and number to kept it varchar to save space and also allow alphanumeric input. Also all orders must have a product id so kept it not null
employee_id	Varchar(10)	NOT NULL	The Id is a combination of letter and number to kept it varchar to save space and also allow alphanumeric input. Also all orders must have an employeer id so kept it not null
order_stock	Int(3)	NOT NULL	Order Stock is basically number of products purchased to kept it int to allow number input. Also it it wont be above 3 digit as originally stock stored is not more than 3 digits Also all orders must have order_stock so not null
order_date	date	NOT NULL	Order date is basically the date product has been ordered. So kept it date type. Also all orders must have an order date to kept it as not null
Order_comment	Varchar(50)	NULL	Comment is combination of number and letters to kept it as varhar to save space and also allow alphanumeric input. An order may or may not have any comment so it can be null

Primary key name: order_id

Foreign key names: customer_id, product_id, employee_id

Table type: It is a weak entity table

e) Table Customers details

Attribute name	Datatype	Null or not	Justification
customer_id	Varchar(10)	NOT NULL	The Id is a combination of letter and number to kept it varchar to save space and also allow alphanumeric input. Also all customers must have an customer id so kept it not null
customer_fname	Varchar(20)	NOT NULL	The fname is basically the firstname of the customer so kept it as varchar to save space and also allow letter input. Also all customers must have firstname so kept it not null
customer_lname	Varchar(20)	NOT NULL	The lname is basically the lastname of the customer so kept it as varchar to save space and also allow letter input. Also all customers must have lastname so kept it not null
customer_gender	Varchar(1)	NOT NULL	Gender basically has 3 possiblities: Male(M), Female (F), Unknown(U). So restricted its input to M,F,U Also all customers must have gender so kept it not null
customer_address	Varchar(20)	NOT NULL	The address is basically the address of the customer so kept it as varchar to save space and also allows letter input. Also all customers must have address so kept it not null
customer_city	Varchar(20)	NOT NULL	The city is basically the city of the customer so kept it as varchar to save space and also allows letter input. Also all customers must have city so kept it not null
customer_postcode	Int(4)	NOT NULL	The postis basically the postcode of the customer living area. So made it int and restricted input to 4 digits. Also all customers must have postcode so kept it not null
customer_phone	Bigint(13)	NOT NULL	Phonenumber is made of numbers, so made it bigint as number input and 13 digits. Didn't use int as it can only take from 0 to 4294967295 when unsigned which is not enough! All customers have their own number and so not null

Primary key name: customer_id

f) Table Employee details

Attribute name	Datatype	Null or not	Justification
employee_id	Varchar(10)	NOT NULL	The Id is a combination of letter and number to kept it varchar to save space and also allow alphanumeric input. Also all employee must have an customer id so kept it not null
employee _fname	Varchar(20)	NOT NULL	The fname is basically the firstname of the employee so kept it as varchar to save space and also allow letter input. Also all employee must have firstname so kept it not null
employee _lname	Varchar(20)	NOT NULL	The lname is basically the lastname of the employee so kept it as varchar to save space and also allow letter input. Also all employee must have lastname so kept it not null
employee _gender	Varchar(1)	NOT NULL	Gender basically has 3 possiblities: Male(M), Female (F), Unknown(U). So restricted its input to M,F,U Also all employee must have gender so kept it not null
Employee_date_of_birth	date	NOT NULL	The date of birth is basically the birthday of the employee so kept it as date type. Also all employee must have birthday so kept it not null
employee _address	Varchar(20)	NOT NULL	The address is basically the address of the employee so kept it as varchar to save space and also allows letter input. Also all employee must have address so kept it not null
employee _city	Varchar(20)	NOT NULL	The city is basically the city of the employee so kept it as varchar to save space and also allows letter input. Also all employee must have city so kept it not null
employee _postcode	Int(4)	NOT NULL	The postcode is basically the postcode of the employee's living area. So made it int and restricted input to 4 digits. Also all employee must have postcode so kept it not null

employee _phone	Bigint(13)	NOT NULL	Phonenumber is made of numbers, so made it bigint as number input and 13 digits. Didn't use int as it can only take from 0 to 4294967295 when unsigned which is not enough! All employee have their own number and so not null
employee_current_salary	Int(10)	NULL	Salary basically is the amount of money employee currently earns per year Employee may have newly joined the company and thus he may not have received his salary yet and thus kept it null (alternatively, it is also possible to keep it as not null, but then we will need to keep default as 0)
employee_individual_comment	Varchar(50)	NULL	This is the individual comment about the employee so kept it varchar as letter and to save space. employee may or may not have a comment so kept it as null

Primary key name: employee_id

g) Table Individual_Employee_position

Attribute name	Datatype	Null or not	Justification
employee_id	Varchar(10)	NOT NULL	The Id is a combination of letter and number to kept it varchar to save space and also allow alphanumeric input. Also all employee must have an customer id so kept it not null
employee _position	Varchar(10)	NOT NULL	Position basically has 3 possiblities: Manager, technician, normal. So restricted its input to these and used vachar for letter input and also to save space Also all employee must have position so kept it not null

Foreign key names: employee_id, employee_position

Table type: It is a weak entity table

h) Table Employee_position

Attribute name	Datatype	Null or not	Justification
employee _position	Varchar(10)	NOT NULL	Position basically has 3 possiblities: Manager, technician, normal. So restricted its input to these and used vachar for letter input and also to save space Also all employee must have position so kept it not null
Employee_position_comment	Varchar(50)	NOT NULL	This tell details of what the employee at this position does. So kept it as varchar as alphanumeric input and also to save space. Also made this specific comment not null as tells details about the employee's position's responsibility And thus must have relevant information in it regarding all possible employee positions.

Primary key names: employee_position

i) Table Employee_and_Branch_details

Attribute name	Datatype	Null or not	Justification
Branch_id	Varchar(10)	NOT NULL	The Id is a combination of letter and number to kept it varchar to save space and also allow alphanumeric input. Also all branches must have a branch id so kept it not null
Employee_id	Varchar(10)	NOT NULL	The Id is a combination of letter and number to kept it varchar to save space and also allow alphanumeric input. Also all employes must have a employee id so kept it not null

Foreign key names: branch_id, employee_id

Table type: It is a weak entity table

j) Table Branch_details

Attribute name	Datatype	Null or not	Justification
Branch_id	Varchar(10)	NOT NULL	The Id is a combination of letter and number to kept it varchar to save space and also allow alphanumeric input. Also all branches must have a branch id so kept it not null
Branch_name	Varchar(20)	NOT NULL	The name is basically name of branch so kept it varchar to save space and also allow letter input Also all branches must have a branch name so kept it not null
Branch_comment	Varchar(20)	NULL	This is the individual comment about the branch so kept it varchar as letter and to save space. branch may or may not have a comment so kept it as null

Primary key names: branch_id

Scripts used to make the database:

```
CREATE database CompanyNew;
Use CompanyNew;

Create table Branch_details
(
branch_id varchar(10) NOT NULL,
branch_name varchar(20) NOT NULL,
branch_comment varchar(50),
 PRIMARY KEY(branch_id)
);

Create table Employee_details
(
employee_id varchar(10) NOT NULL ,
employee_fname varchar(20) NOT NULL,
```

```
employee_lname varchar(20) NOT NULL,
employee_gender varchar(1) NOT NULL
CHECK (employee_gender IN ('M','F', 'U')),
employee_date_of_birth date NOT NULL,
employee_address varchar(20) NOT NULL,
employee_city varchar(20) NOT NULL,
employee_postcode int(4) UNSIGNED NOT NULL,
employee_phone  BIGINT(13) UNSIGNED NOT NULL,
employee_current_salary int(10) UNSIGNED,
employee_individual_comment varchar(50),
PRIMARY KEY(employee_id)
);

Create table Employee_and_Branch_details
(
branch_id varchar(10) NOT NULL,
employee_id varchar(10) NOT NULL ,
PRIMARY KEY(branch_id , employee_id),
```

```
FOREIGN KEY(branch_id) REFERENCES Branch_details (branch_id),
FOREIGN KEY(employee_id) REFERENCES Employee_details (employee_id)
);

Create table Employee_position
(
employee_position varchar(10) NOT NULL
CHECK (employee_position IN ('Manager','Technician', 'Normal')),
employee_position_comment varchar(50) NOT NULL,
PRIMARY KEY(employee_position)
);

Create table Individual_Employee_position
(
employee_id varchar(10) NOT NULL ,
employee_position varchar(10) NOT NULL
CHECK (employee_position IN ('Manager','Technician', 'Normal')),
PRIMARY KEY(employee_id, employee_position),
```

```
FOREIGN KEY(employee_id) REFERENCES Employee_details (employee_id),
FOREIGN KEY(employee_position) REFERENCES Employee_position (employee_position)
);

Create table Subsidiary_company_details
(
subsidiary_company_id varchar(10) NOT NULL,
subsidiary_company_name varchar(20) NOT NULL,
subsidiary_company_address varchar(40) NOT NULL,
subsidiary_company_city varchar(20) NOT NULL,
subsidiary_company_postcode int(4) UNSIGNED NOT NULL,
subsidiary_company_phonenumber bigint(13) UNSIGNED NOT NULL,
PRIMARY KEY(subsidiary_company_id)
);

Create table Product_details
(
product_id varchar(10) NOT NULL ,
```

```
product_name varchar(20) NOT NULL,
product_type varchar(8) NOT NULL
CHECK (product_type IN ('chair', 'table', 'sofa', 'wardrobe')),
product_material varchar(7) NOT NULL
CHECK (product_material IN ('wood', 'plastic', 'glass')),
product_price int(6) UNSIGNED NOT NULL,
product_comment varchar(50),
subsidiary_company_id varchar(10) NOT NULL,
product_purchase_date_from_subsidiary Date NOT NULL,
product_in_stock int(3) UNSIGNED NOT NULL,
PRIMARY KEY(product_id ,subsidiary_company_id),
FOREIGN KEY(subsidiary_company_id) REFERENCES Subsidiary_company_details (subsidiary_company_id)
);

Create table Customers_details
(
customer_id varchar(10) NOT NULL,
customer_fname varchar(20) NOT NULL,
```

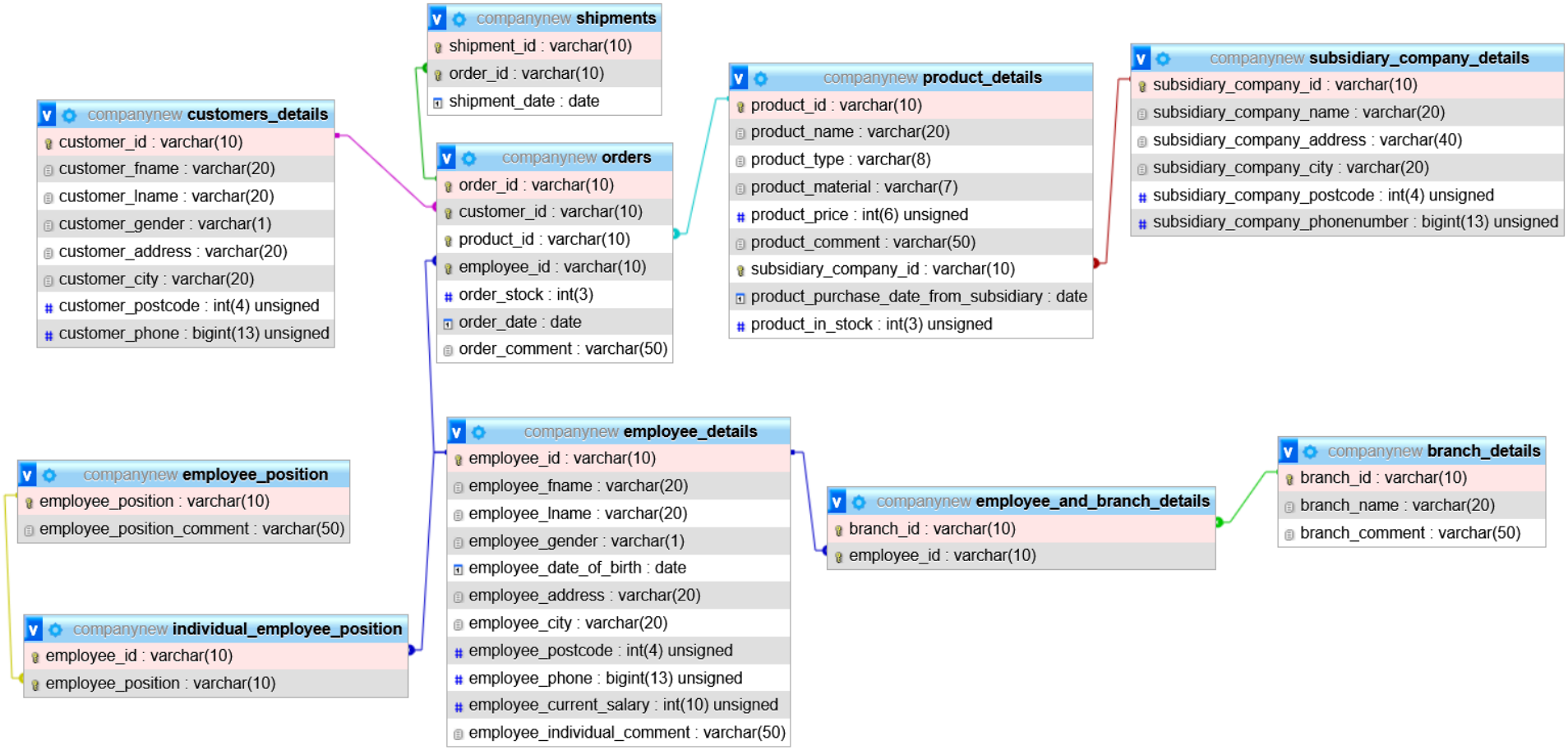
```
customer_lname varchar(20) NOT NULL,
customer_gender varchar(1) NOT NULL
CHECK (customer_gender IN ('M','F', 'U')),
customer_address varchar(20) NOT NULL,
customer_city varchar(20) NOT NULL,
customer_postcode int(4) UNSIGNED NOT NULL,
customer_phone  BIGINT(13) UNSIGNED NOT NULL,
PRIMARY KEY(customer_id)
);

Create table Orders
(
order_id varchar(10) NOT NULL,
customer_id varchar (10) NOT NULL,
product_id varchar(10) NOT NULL,
employee_id varchar(10) NOT NULL ,
order_stock int(3) NOT NULL,
order_date date NOT NULL,
```

```
order_comment varchar(50),

PRIMARY KEY(order_id , customer_id ,product_id, employee_id),
FOREIGN KEY(customer_id) REFERENCES Customers_details (customer_id),
FOREIGN KEY(product_id) REFERENCES Product_details (product_id),
FOREIGN KEY( employee_id) REFERENCES Employee_details ( employee_id)
);

Create table Shipments
(
shipment_id varchar(10) NOT NULL,
order_id varchar(10) NOT NULL,
shipment_date date NOT NULL,
PRIMARY KEY(shipment_id  ,order_id),
FOREIGN KEY(order_id) REFERENCES Orders (order_id)
);
```


companynew shipments
shipment_id : varchar(10)
order_id : varchar(10)
shipment_date : date
companynew customers_details
customer_id : varchar(10)
customer_fname : varchar(20)
customer_lname : varchar(20)
customer_gender : varchar(1)
customer_address : varchar(20)
customer_city : varchar(20)
customer_postcode : int(4) unsigned
customer_phone : bigint(13) unsigned
companynew orders
order_id : varchar(10)
customer_id : varchar(10)
product_id : varchar(10)
employee_id : varchar(10)
order_stock : int(3)
order_date : date
order_comment : varchar(50)
companynew product_details
product_id : varchar(10)
product_name : varchar(20)
product_type : varchar(8)
product_material : varchar(7)
product_price : int(6) unsigned
product_comment : varchar(50)
subsidiary_company_id : varchar(10)
product_purchase_date_from_subsidiary : date
product_in_stock : int(3) unsigned
companynew subsidiary_company_details
subsidiary_company_id : varchar(10)
subsidiary_company_name : varchar(20)
subsidiary_company_address : varchar(40)
subsidiary_company_city : varchar(20)
subsidiary_company_postcode : int(4) unsigned
subsidiary_company_phonenumber : bigint(13) unsigned
companynew employee_details
employee_id : varchar(10)
employee_fname : varchar(20)
employee_lname : varchar(20)
employee_gender : varchar(1)
employee_date_of_birth : date
employee_address : varchar(20)
employee_city : varchar(20)
employee_postcode : int(4) unsigned
employee_phone : bigint(13) unsigned
employee_current_salary : int(10) unsigned
employee_individual_comment : varchar(50)
companynew employee_position
employee_position : varchar(10)
employee_position_comment : varchar(50)
companynew individual_employee_position
employee_id : varchar(10)
employee_position : varchar(10)
companynew employee_and_branch_details
branch_id : varchar(10)
employee_id : varchar(10)
companynew branch_details
branch_id : varchar(10)
branch_name : varchar(20)
branch_comment : varchar(50)

Scripts used to insert data for all the tables:

```
INSERT INTO customers_details
 (customer_id, customer_fname, customer_lname, customer_gender,
  customer_address, customer_city, customer_postcode, customer_phone)
 VALUES ('HABCA00024', 'Sultan', 'Ahmed', 'M',
  'Road 2, Gulshan', 'Dhaka', 2141, 8801792458030);
  INSERT INTO customers_details
 (customer_id, customer_fname, customer_lname, customer_gender,
  customer_address, customer_city, customer_postcode, customer_phone)
 VALUES ('ADBCA00554', 'Kawsar', 'Hossain', 'M',
  'Road 10, Bonani', 'Dhaka', 4156, 8801796958971);
   INSERT INTO customers_details
 (customer_id, customer_fname, customer_lname, customer_gender,
  customer_address, customer_city, customer_postcode, customer_phone)
 VALUES ('FABCA09548', 'Robert', 'Brown', 'M',
  'Road 10, Modina', 'Sylhet', 3114, 8801892884308);
```

```
 INSERT INTO customers_details
(customer_id, customer_fname, customer_lname, customer_gender,
 customer_address, customer_city, customer_postcode, customer_phone)
VALUES ('ALBCA11020', 'Anne', 'Stuward', 'F',
 'Road 2, Rampura', 'Dhaka', 6915, 8801692433190);
 INSERT INTO customers_details
(customer_id, customer_fname, customer_lname, customer_gender,
 customer_address, customer_city, customer_postcode, customer_phone)
VALUES ('GABCA07324', 'Salma', 'Begum', 'F',
 'Road 21, Gulshan', 'Dhaka', 2114, 8801592457044);


INSERT INTO subsidiary_company_details
 (subsidiary_company_id, subsidiary_company_name, subsidiary_company_address, subsidiary_company_city,
 subsidiary_company_postcode, subsidiary_company_phonenumber)
VALUES ('SCOMP00001', 'Office Products', 'Road 49, Gulshan', 'Dhaka',
 2441, 8801972448031);
 INSERT INTO subsidiary_company_details
(subsidiary_company_id, subsidiary_company_name, subsidiary_company_address, subsidiary_company_city,
```

```
 subsidiary_company_postcode, subsidiary_company_phonenumber)
VALUES ('SCOMP00444', 'Stylish Products', 'Road 2, Gulshan', 'Dhaka',
 2411, 8801973448031);
 INSERT INTO subsidiary_company_details
(subsidiary_company_id, subsidiary_company_name, subsidiary_company_address, subsidiary_company_city,
 subsidiary_company_postcode, subsidiary_company_phonenumber)
VALUES ('SCOMP00051', 'RFL', 'Road 4,Bonani', 'Dhaka',
 6518, 8801745336711);
 INSERT INTO subsidiary_company_details
(subsidiary_company_id, subsidiary_company_name, subsidiary_company_address, subsidiary_company_city,
 subsidiary_company_postcode, subsidiary_company_phonenumber)
VALUES ('SCOMP05001', 'Glassify', 'Road 4, Kumarpara', 'Sylhet',
 3141, 8801745336700);


   INSERT INTO product_details
(product_id, product_name, product_type, product_material,
 product_price, product_comment, subsidiary_company_id, product_purchase_date_from_subsidiary,product_in_stock)
VALUES ('PRDTA00103', 'Office chair', 'chair', 'wood',
```

```
 4000, 'good office chair made of wood', 'SCOMP00001', '2020-12-12',70);
 INSERT INTO product_details
(product_id, product_name, product_type, product_material,
 product_price, product_comment, subsidiary_company_id, product_purchase_date_from_subsidiary, product_in_stock)
VALUES ('PHGTA00144', 'Office table', 'table', 'wood',
 6000, 'good office table made of wood', 'SCOMP00001', '2019-2-16', 40);
 INSERT INTO product_details
(product_id, product_name, product_type, product_material,
 product_price, product_comment, subsidiary_company_id, product_purchase_date_from_subsidiary, product_in_stock)
VALUES ('PRDTG00210', 'stylish chair', 'chair', 'wood',
 10000, 'good chair made of wood', 'SCOMP00444', '2020-12-15', 50);
 INSERT INTO product_details
(product_id, product_name, product_type, product_material,
 product_price, product_comment, subsidiary_company_id, product_purchase_date_from_subsidiary, product_in_stock)
VALUES ('PRDTA00140', 'RFL sofa', 'sofa', 'plastic',
 500, 'good RFL chair made of plastic', 'SCOMP00051', '2018-10-12', 90);
 INSERT INTO product_details
(product_id, product_name, product_type, product_material,
```

```
 product_price, product_comment, subsidiary_company_id, product_purchase_date_from_subsidiary, product_in_stock)
VALUES ('PROOA00103', 'clear wardrobe', 'wardrobe', 'glass',
 8000, 'good wardrobe made of glass', 'SCOMP05001', '2020-10-10', 90);


INSERT INTO employee_details
(employee_id, employee_fname, employee_lname, employee_gender, employee_date_of_birth,
 employee_address, employee_city, employee_postcode, employee_phone, employee_current_salary, employee_individual_comment)
VALUES ('EMPLY44556', 'Karim', 'Kombol', 'M',
 '1990-10-24', 'Road 45, Gulshan', 'Dhaka', 2356, 8801731246924, 80000, 'good old employee');
INSERT INTO employee_details
(employee_id, employee_fname, employee_lname, employee_gender, employee_date_of_birth,
 employee_address, employee_city, employee_postcode, employee_phone, employee_current_salary, employee_individual_comment)
VALUES ('EMPLE24598', 'Bokul', 'Ahmed', 'F',
 '1995-1-2', 'Road 40, Gulshan', 'Dhaka', 2557, 8801933246924, 60000, 'clumsy employee');
INSERT INTO employee_details
(employee_id, employee_fname, employee_lname, employee_gender, employee_date_of_birth,
 employee_address, employee_city, employee_postcode, employee_phone, employee_current_salary, employee_individual_comment)
VALUES ('ETYLE24594', 'Mokles', 'Miah', 'M',
```

```
  '2000-10-20', 'Road 45, Subidbazar', 'Sylhet', 3156, 8801731296924, 40000, 'Lazy  employee');
INSERT INTO employee_details
 (employee_id, employee_fname, employee_lname, employee_gender, employee_date_of_birth,
  employee_address, employee_city, employee_postcode, employee_phone, employee_current_salary, employee_individual_comment)
 VALUES ('QWRTY11111', 'Raima', 'Karim', 'F',
  '2001-3-4', 'Road 4, Modinamarket', 'Sylhet', 4156, 8801455672298, 80000, 'Diligent employee');
INSERT INTO employee_details
 (employee_id, employee_fname, employee_lname, employee_gender, employee_date_of_birth,
  employee_address, employee_city, employee_postcode, employee_phone, employee_current_salary, employee_individual_comment)
 VALUES ('QSART45692', 'Kashem', 'Uddin', 'M',
  '1995-5-5', 'Road 8, Rampura', 'Dhaka', 2400, 8801459272298, 90000, 'old techy employee');


INSERT INTO branch_details
 (branch_id, branch_name, branch_comment)
 VALUES ('BRNCH40002', 'Dhaka Branch', 'Exclusive Branch in Dhaka');
 INSERT INTO branch_details
 (branch_id, branch_name, branch_comment)
 VALUES ('BRNCH20003', 'Sylhet Branch', 'Exclusive Branch in Sylhet');
```

```
INSERT INTO employee_and_branch_details
 (branch_id, employee_id)
 VALUES ('BRNCH40002', 'EMPLY44556');
  INSERT INTO employee_and_branch_details
 (branch_id, employee_id)
 VALUES ('BRNCH40002', 'EMPLE24598');
  INSERT INTO employee_and_branch_details
 (branch_id, employee_id)
 VALUES ('BRNCH40002', 'ETYLE24594');
 INSERT INTO employee_and_branch_details
 (branch_id, employee_id)
 VALUES ('BRNCH20003', 'QWRTY11111');
 INSERT INTO employee_and_branch_details
 (branch_id, employee_id)
 VALUES ('BRNCH20003', 'QSART45692');
```

```
INSERT INTO employee_position
 (employee_position, employee_position_comment)
 VALUES ('Manager', 'responsible for controlling group of staff');
 INSERT INTO employee_position
 (employee_position, employee_position_comment)
 VALUES ('Technician', 'maintains technical equipment');
  INSERT INTO employee_position
 (employee_position, employee_position_comment)
 VALUES ('Normal', 'ordinary person employeed');

 INSERT INTO individual_employee_position
 (employee_id, employee_position)
 VALUES ('EMPLY44556', 'Manager');
 INSERT INTO individual_employee_position
 (employee_id, employee_position)
 VALUES ('EMPLE24598', 'Normal');
 INSERT INTO individual_employee_position
```

```
(employee_id, employee_position)
VALUES ('ETYLE24594', 'Normal');
INSERT INTO individual_employee_position
(employee_id, employee_position)
VALUES ('QWRTY11111', 'Manager');
INSERT INTO individual_employee_position
(employee_id, employee_position)
VALUES ('QSART45692', 'Technician');

 INSERT INTO orders
(order_id, customer_id, product_id, employee_id, order_stock, order_date, order_comment)
VALUES ('ORDRE45679',  'HABCA00024',  'PHGTA00144',  'EMPLE24598',1,  '2021-5-24',  'customer felt weird');
INSERT INTO orders
(order_id, customer_id, product_id, employee_id, order_stock, order_date, order_comment)
VALUES ('ORDRE45680',  'HABCA00024',  'PRDTA00103',  'EMPLE24598',1,  '2021-5-24',  'customer felt weird');
INSERT INTO orders
(order_id, customer_id, product_id, employee_id, order_stock, order_date, order_comment)
VALUES ('ORDRE44990',  'FABCA09548',  'PROOA00103',  'ETYLE24594',10,  '2021-6-26',  'good old customer back');
```

```
INSERT INTO orders
 (order_id, customer_id, product_id, employee_id, order_stock, order_date, order_comment)
 VALUES ('ORDRE49100',  'GABCA07324',  'PRDTA00103',  'EMPLE24598',10,  '2021-7-7',  'customer wants to track this');

INSERT INTO shipments
 (shipment_id, order_id, shipment_date)
 VALUES ('SHPIN30078',  'ORDRE45679',  '2021-5-26');
 INSERT INTO shipments
 (shipment_id, order_id, shipment_date)
 VALUES ('SHPIN30079',  'ORDRE45680',  '2021-5-26');
  INSERT INTO shipments
 (shipment_id, order_id, shipment_date)
 VALUES ('SHPIN40705',  'ORDRE44990',  '2021-6-28');
INSERT INTO shipments
 (shipment_id, order_id, shipment_date)
 VALUES ('SHPIN55555',  'ORDRE49100',  '2021-7-9');
```

Use of 5 joins:

Some of the joins I used in the database and their purposes: (added reasoning for the places where I made views and sides notes for normal joins)

1) Use Join to make a view with full employee details (including position and their branch names)

Reason:

To make it easier for employer to find all employee details in a single view instead of looking at several tables.
Also, since view is a stored query, it is already pre-optimized and thus would execute faster when opened next time as no need to re-optimize it (As query execution strategy has already been decided on by DBMS when the view was first created). Also, no need to use join and query through multiple tables again and again to find the data. Thus, overall it increases the performance of the database.

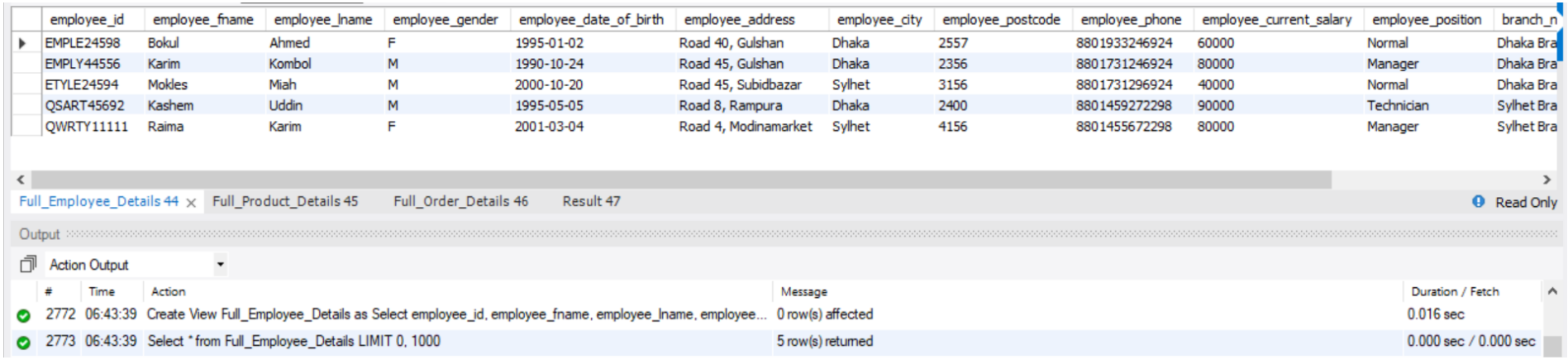
Commands:

```
Create View Full_Employee_Details as
Select employee_id, employee_fname, employee_lname, employee_gender, employee_date_of_birth, employee_address,
employee_city, employee_postcode, employee_phone, employee_current_salary, employee_position,branch_name,
employee_individual_comment

 From employee_details e Natural Join employee_and_branch_details eb Natural Join branch_details b NATURAL Join
individual_employee_position ie NATURAL Join employee_position
Order by employee_id;

Select *
from Full_Employee_Details;
```

Output:



employee_id	employee_fname	employee_lname	employee_gender	employee_date_of_birth	employee_address	employee_city	employee_postcode	employee_phone	employee_current_salary	employee_position	branch_n
EMPLE24598	Bokul	Ahmed	F	1995-01-02	Road 40, Gulshan	Dhaka	2557	8801933246924	60000	Normal	Dhaka Bra
EMPLY44556	Karim	Kombol	M	1990-10-24	Road 45, Gulshan	Dhaka	2356	8801731246924	80000	Manager	Dhaka Bra
ETYLE24594	Mokles	Miah	M	2000-10-20	Road 45, Subidbazar	Sylhet	3156	8801731296924	40000	Normal	Dhaka Bra
QSART45692	Kashem	Uddin	M	1995-05-05	Road 8, Rampura	Dhaka	2400	8801459272298	90000	Technician	Sylhet Bra
QWRTY11111	Raima	Karim	F	2001-03-04	Road 4, Modinamarket	Sylhet	4156	8801455672298	80000	Manager	Sylhet Bra

Full_Employee_Details 44 | Full_Product_Details 45 | Full_Order_Details 46 | Result 47 | Read Only

Output

Action Output

#	Time	Action	Message	Duration / Fetch
2772	06:43:39	Create View Full_Employee_Details as Select employee_id, employee_fname, employee_lname, employee...	0 row(s) affected	0.016 sec
2773	06:43:39	Select * from Full_Employee_Details LIMIT 0, 1000	5 row(s) returned	0.000 sec / 0.000 sec

2) Use Join to make a view for full product details (including subsidiary company name

Reason:

To have a single view with both poduct, subsidiary company information. This makes it easier for company to restock the products quickly, as they can easily see which company its from and the phone no of that company. Also there is company's city included and thus they can use it to estimate how long it will take for the product to be delivered to them.

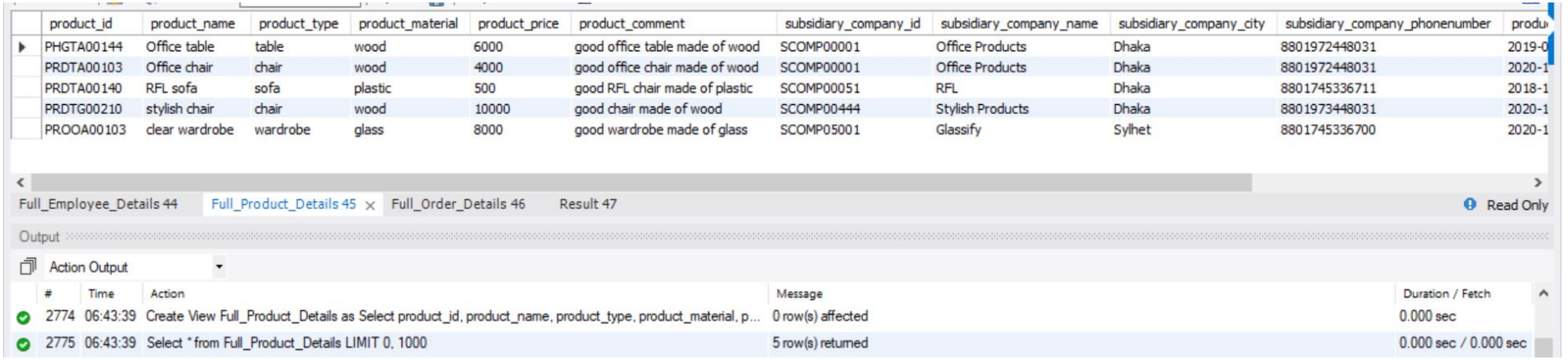
Also, since view is a stored query, it is already pre-optimized and thus would execute faster when opened next time as no need to re-optimize it (As query execution strategy has already been decided on by DBMS when the view was first created). Also, no need to use join and query through multiple tables again and again to find the data. Thus, overall it increases the performance of the database.

Command:

```
Create View Full_Product_Details as
Select product_id, product_name, product_type, product_material, product_price, product_comment, subsidiary_company_id,
subsidiary_company_name, subsidiary_company_city, subsidiary_company_phonenumber, product_purchase_date_from_subsidiary,
product_in_stock
 From product_details pd Natural Join subsidiary_company_details scd
Order by product_id;

Select *
from Full_Product_Details;
```

Output:



product_id	product_name	product_type	product_material	product_price	product_comment	subsidiary_company_id	subsidiary_company_name	subsidiary_company_city	subsidiary_company_phonenumber	produ
PHGTA00144	Office table	table	wood	6000	good office table made of wood	SCOMP00001	Office Products	Dhaka	8801972448031	2019-0
PRDTA00103	Office chair	chair	wood	4000	good office chair made of wood	SCOMP00001	Office Products	Dhaka	8801972448031	2020-1
PRDTA00140	RFL sofa	sofa	plastic	500	good RFL chair made of plastic	SCOMP00051	RFL	Dhaka	8801745336711	2018-1
PRDTG00210	stylish chair	chair	wood	10000	good chair made of wood	SCOMP00444	Stylish Products	Dhaka	8801973448031	2020-1
PROOA00103	clear wardrobe	wardrobe	glass	8000	good wardrobe made of glass	SCOMP05001	Glassify	Sylhet	8801745336700	2020-1

3) Use Join to make a view for order stating name of employee and product and customer buying it

Reason:

To have a single view to see all order details including customer, employee who sold it, total cost of order, etc.

Also, since view is a stored query, it is already pre-optimized and thus would execute faster when opened next time as no need to re-optimize it (As query execution strategy has already been decided on by DBMS when the view was first created). Also, no need to use join and query through multiple tables again and again to find the data. Thus, overall it increases the performance of the database.

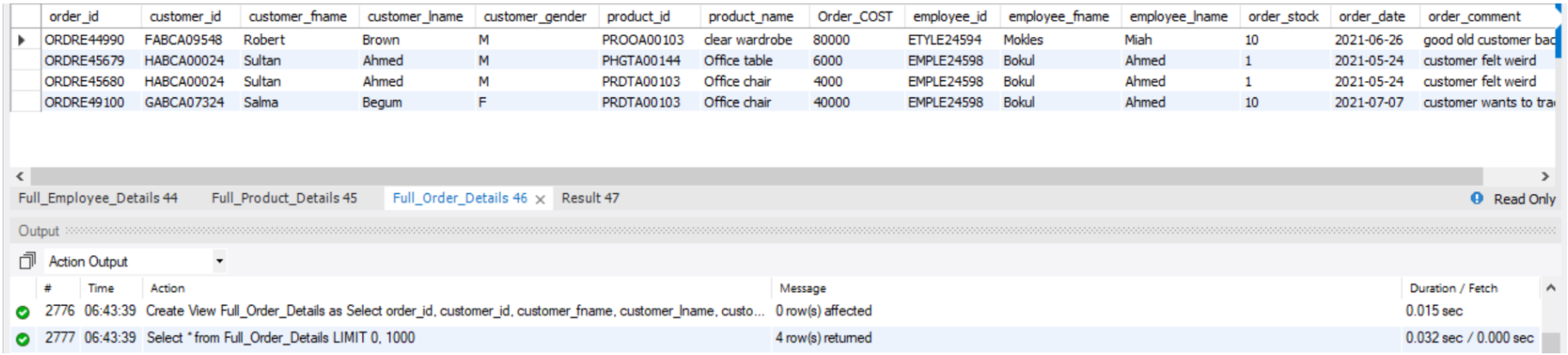
Command:

```
Create View Full_Order_Details as

Select order_id, customer_id, customer_fname, customer_lname, customer_gender, product_id, product_name,
(SUM(product_price*order_stock)) as Order_COST,employee_id,
employee_fname, employee_lname, order_stock, order_date, order_comment
 From orders o Natural Join product_details p  Natural Join customers_details c Natural Join employee_details e
Group by order_id
Order by order_id;

Select *
from Full_Order_Details;
```

Output



order_id	customer_id	customer_fname	customer_lname	customer_gender	product_id	product_name	Order_COST	employee_id	employee_fname	employee_lname	order_stock	order_date	order_comment
ORDRE44990	FABCA09548	Robert	Brown	M	PROOA00103	clear wardrobe	80000	ETYLE24594	Mokles	Miah	10	2021-06-26	good old customer bac
ORDRE45679	HABCA00024	Sultan	Ahmed	M	PHGTA00144	Office table	6000	EMPLE24598	Bokul	Ahmed	1	2021-05-24	customer felt weird
ORDRE45680	HABCA00024	Sultan	Ahmed	M	PRDTA00103	Office chair	4000	EMPLE24598	Bokul	Ahmed	1	2021-05-24	customer felt weird
ORDRE49100	GABCA07324	Salma	Begum	F	PRDTA00103	Office chair	40000	EMPLE24598	Bokul	Ahmed	10	2021-07-07	customer wants to tra

Full_Employee_Details 44 Full_Product_Details 45 Full_Order_Details 46 Result 47 Read Only

Output

Action Output

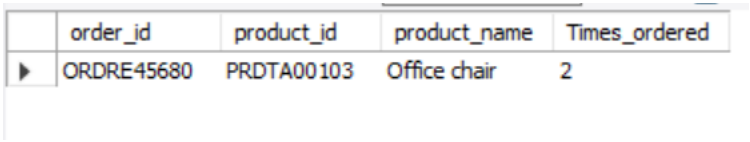
#	Time	Action	Message	Duration / Fetch
2776	06:43:39	Create View Full_Order_Details as Select order_id, customer_id, customer_fname, customer_lname, custo...	0 row(s) affected	0.015 sec
2777	06:43:39	Select * from Full_Order_Details LIMIT 0, 1000	4 row(s) returned	0.032 sec / 0.000 sec

4) Use join to find which product was ordered more than once:

Commands:

```
Select order_id, product_id, product_name, (Count(product_id)) as Times_ordered
 From orders o Natural Join product_details p
group by product_id
Having Times_ordered>1;
```

Output:



order_id	product_id	product_name	Times_ordered
ORDRE45680	PRDTA00103	Office chair	2

Sidenote:

(We can also directly query from the “view” instead of using the joins, if that has been made beforehand. That will be more efficient and better as joins are expensive as it needs to look though many tables and compare datas before querying)

5) Use Join to find what the weird customer had ordered in details:

Command:

```
Select order_id, customer_id, customer_fname, customer_lname, product_id, product_name,  (SUM(product_price*order_stock)) as
Order_COST,
order_stock, order_date, order_comment
 From orders o Natural Join product_details p  Natural Join customers_details c
Group by order_id
Having customer_id = 'HABCA00024'
Order by order_id;
```

Output:

order_id	customer_id	customer_fname	customer_lname	product_id	product_name	Order_COST	order_stock	order_date	order_comment
ORDRE45679	HABCA00024	Sultan	Ahmed	PHGTA00144	Office table	6000	1	2021-05-24	customer felt weird
ORDRE45680	HABCA00024	Sultan	Ahmed	PRDTA00103	Office chair	4000	1	2021-05-24	customer felt weird

Sidenote:

(We can also directly query from the “view” instead of using the joins, if that has been made beforehand. That will be more efficient and better as joins are expensive as it needs to look though many tables and compare datas before querying)

Reference:

- Information from COS20015 (Fundamentals of database management) slide notes regarding MYSQL