due date: 10.01.2023

Beykoz University

Department of “Computer Engineering”

“Database Systems - 60612MEEOZ-CME0075”

- Restaurant Management System -

- Project Phase II -

Lecturer: Dr. Selçuk KIRAN

Project Owner:
Leyla Abdullayeva -
1904010038

sign:

The names and current photos of the students should be on the first page and the following text should be inserted and signed.

Şerefim ve inandığım tüm değerler adına yemin ederim ki bu sınavda hiçbir yerden kopya çekmedim ve etik kurallarına karşı gelmedim.

I swear on behalf of my honor and all the values I believe in that I did not cheat in this exam from anywhere and I did not violate the ethic rules.

TABLE OF CONTENTS

PHASE I

1.Brief Description of the System

A Restaurant Management Database System is a computerized system that is designed to help manage the various aspects of a restaurant, including menu management, employee scheduling, and inventory tracking. The system is typically implemented using a database management system such as Microsoft SQL Server.

This study includes a new database application that will help the restaurant manager to manage the restaurant more effectively and efficiently by computerizing meal ordering, billing and inventory control. This is the main purpose of this Project.

The database infrastructure of the program is MS SQL.

The goals of our system are:

1. Improved efficiency in managing and updating the restaurant menu.
2. Streamlined employee scheduling and task assignments.
3. Enhanced tracking and management of inventory.
4. Improved financial reporting and analysis.
5. Enhanced customer relationship management.
6. Increased ability to measure and analyze key performance indicators.
7. Improved communication and collaboration among staff.
8. Enhanced security and data protection.
9. Increased overall profitability of the restaurant.

1.1 Scope Definition

The scope of the Restaurant Management Database System project is to develop a comprehensive computerized system that automates various aspects of restaurant operations, including billing, employee management, and record-keeping. The system will be implemented using a database management system such as Microsoft SQL Server, and will be designed to be used by employees at all levels of the organization. The main goals of the project are to improve efficiency, streamline processes, and increase profitability by providing real-time data analysis and decision-making tools. The system will be designed to be user-friendly, with different levels of access granted to employees based on their roles and responsibilities. This will ensure that the privacy and security of sensitive records are protected. In addition to benefiting employees, the system is also expected to provide a better experience for customers by enabling faster, more accurate service. The database will be the central repository for all

restaurant-related data, and will be used to track and manage inventory, employee schedules, and other key aspects of restaurant management. The database will be maintained throughout the project to ensure the accuracy and integrity of the data it contains.

1.2. Determining System Requirements

1. Users (both restaurant managers, waiters, chefs) can login to the system and will be able to do different operations.
2. Restaurant manager (owner) can modify (change) its own data.
3. The system can verify the data before a transaction.
4. Restaurant manager should be able to update information about his/her restaurant.
5. Restaurant manager should be able to view weekly sales for his/her (own) outlet.
6. All the users including (restaurant manager, chef, waiter, customer) can check menu / food data by clicking on a certain menu's main page in a web application (or just page).
7. The system should have a login feature for different users, including restaurant managers, waiters, and chefs, to access and perform various actions.
8. Restaurant managers should have the ability to edit their own information.
9. The system should have the ability to verify data before transactions.
10. Restaurant managers should be able to update information about their restaurant.
11. Restaurant managers should be able to view weekly sales for their specific outlet.
12. Users, including restaurant managers, chefs, waiters, and customers, should be able to view menu and food information through a web application or page.
13. The system should be able to handle large amounts of data efficiently.
14. The system should have secure access controls to protect sensitive information.
15. The system should have backup and recovery capabilities in case of data loss.
16. The system should be scalable to allow for potential expansion in the future.

1.3 Potential Users of the System

After the database design, user screen designs with different authorizations to use the system were made. Database tables were created to meet the needs of three different types of users.

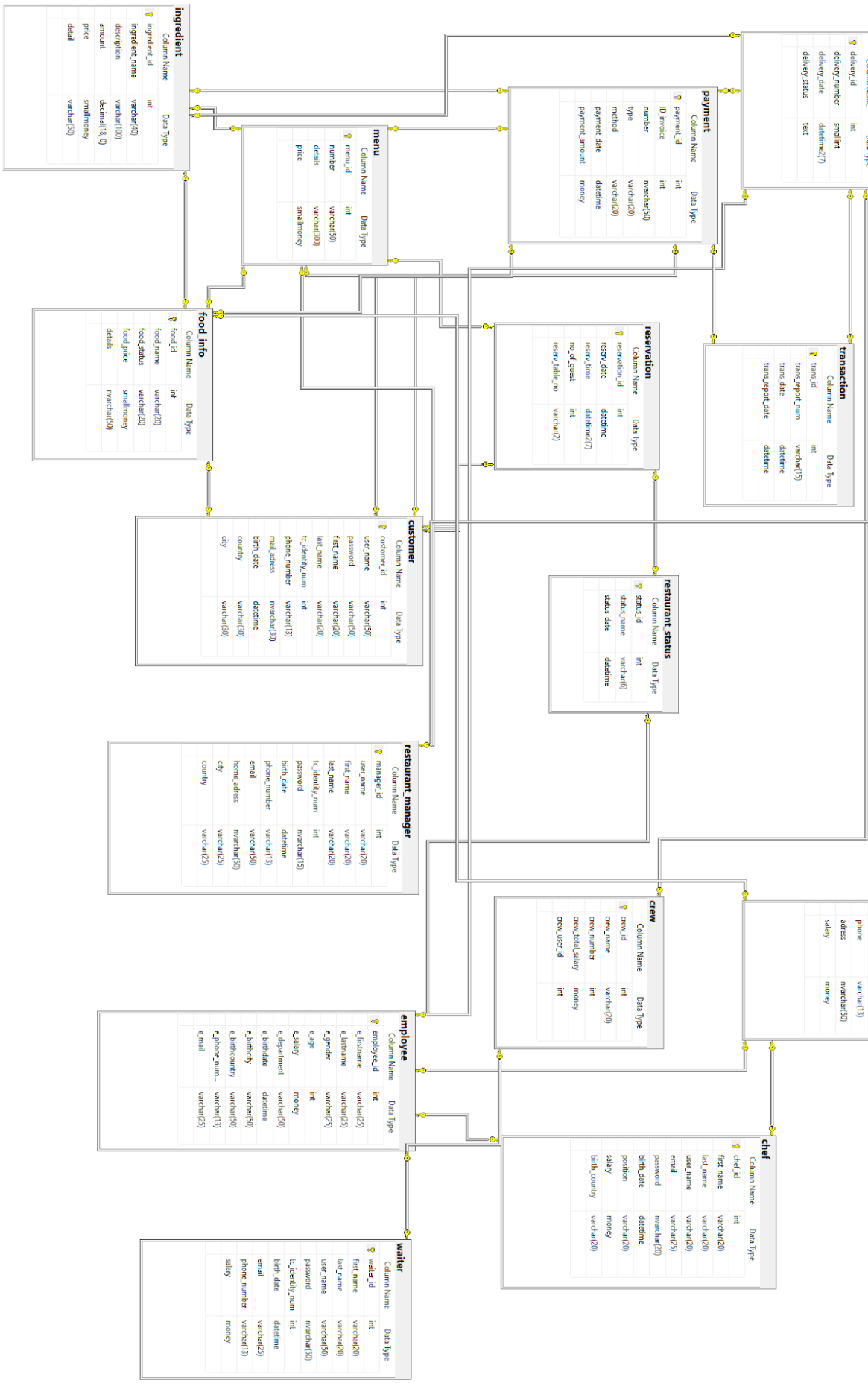
- ***Restaurant Manager*** – Restaurant manager will receive statistical reports and will be able to follow business operations. These users will have the highest level of access to the system and will be responsible for managing and updating information about their restaurant, such as menu items and pricing. They will also be able to view sales data and make decisions based on this information.
- ***Customer*** – While customers may not have direct access to the system, they will still be considered users as they will be able to view menu and pricing information through a web application or page. This will allow them to make informed decisions about their orders. Customers can order the food by using this software from computers & also mobile smartphones.
- ***Employees (Waiter,Chef, Staff)*** - These users may include any other staff members at the restaurant, such as dishwashers or bussers, who may not have the same level of access as the other user groups. Their access and responsibilities will depend on their specific roles within the restaurant.
- ***Waiters -*** These users will be responsible for inputting orders into the system and updating the status of orders as they are prepared and served to customers. They will also be able to access menu and pricing information as needed.
- ***Chefs -*** These users will be able to view orders as they are inputted into the system and mark them as completed when they are finished preparing the food. They will also have access to menu and ingredient information to assist with food preparation.

2.Des

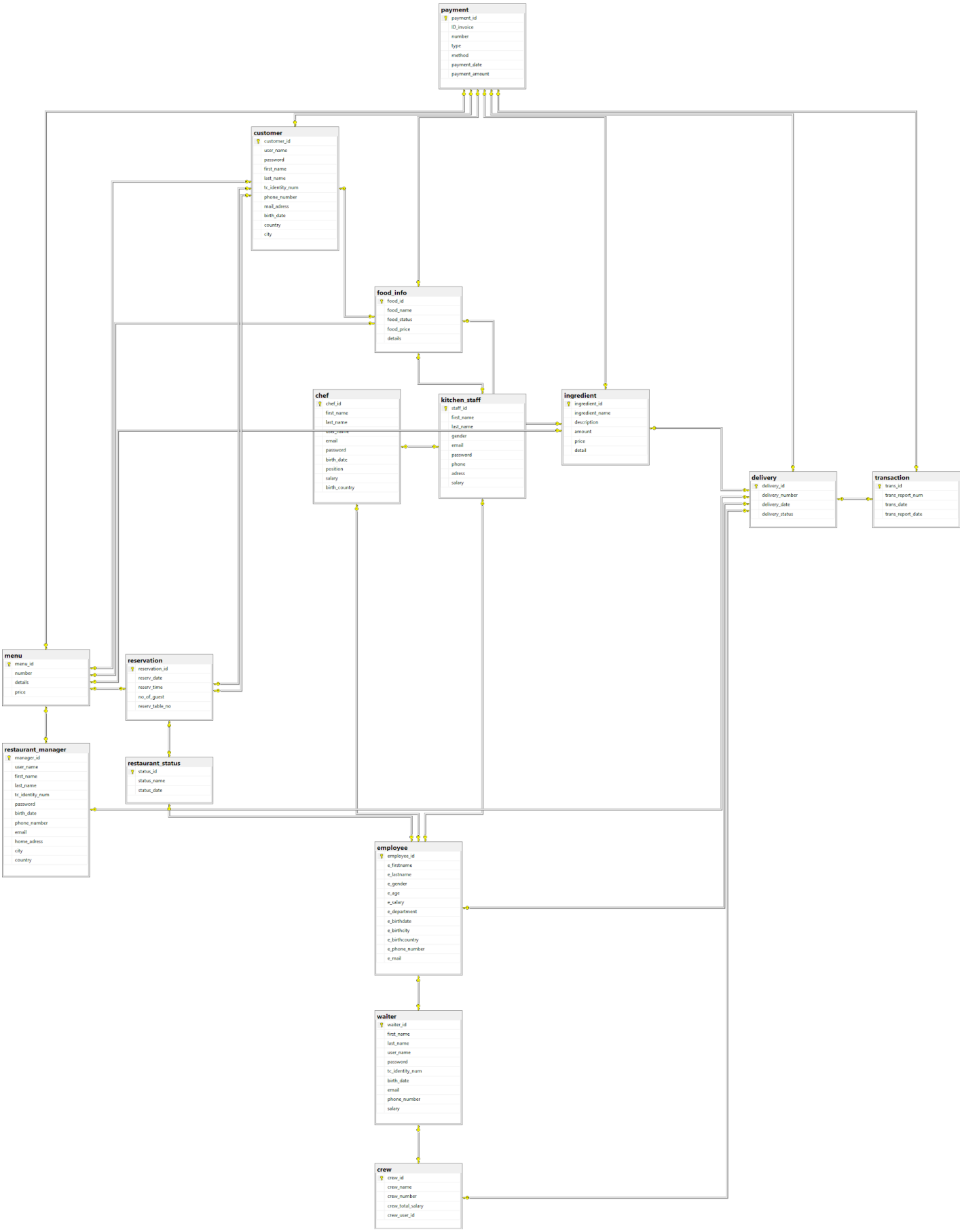
2.1 ERD


dbo.menu
menu_id int PK
number varchar(50) NULL
ingredient_id int FK
payment_id int FK
food_ID int FK
customer_id int FK
reservation_id int FK
details varchar(300) NULL
price smallmoney
dbo.reservation
reservation_id int PK
reserv_date datetime NULL
customer_id int FK
menu_id int FK
reserv_time datetime2(7)
no_of_guest int
reserv_table_no varchar(2) NULL
dbo.food_info
food_ID int PK
food_name varchar(20) NULL
menu_id int FK
food_status varchar(20) NULL
food_price smallmoney NULL
details nvarchar(50)
dbo.customer
customer_id int PK
user_name varchar(50) NULL
menu_id int FK
password varchar(50)
first_name varchar(20)
last_name varchar(20)
tc_identity_num_1 int NULL
phone_number varchar(13) NULL
mail_adress nvarchar(30)
birth_date datetime NULL
country varchar(30) NULL
city_1 varchar(30) NULL
dbo.delivery
delivery_id int PK
ingredient_id_2 int PK FK
delivery number smallint
ingredient_id_3 int FK
ingredient_id_1 int FK
ingredient_id int FK
delivery_date datetime2(7) NULL
delivery_status text
dbo.payment
payment_id int PK
ID_invoice int NULL
trans_ID int FK
manager_id int FK
customer_id int FK
ingredient_id int FK
menu_id int FK
number nvarchar(50) NULL
type varchar(20) NULL
method varchar(20) NULL
payment_date datetime
payment_amount money
dbo.restaurant_manager
manager_id int PK
user_name varchar(20)
menu_id int FK
first_name varchar(20)
last_name varchar(20)
tc_identity_num_1 int NULL
password nvarchar(15)
birth_date datetime NULL
phone_number varchar(13) NULL
email varchar(50)
home_adress nvarchar(50) NULL
city varchar(25) NULL
country varchar(20) NULL
dbo.crew
crew_id int PK
crew_name varchar(20) NULL
chef_id int FK
waiter_id int FK
employee_id int FK
crew_number int
crew_total_salary money NULL
crew_user_id int
dbo.employee
e_firstname varchar(25)
waiter_id int FK
staff_id int FK
e_lastname varchar(25)
e_gender varchar(25) NULL
e_age int NULL
e_salary money
e_department varchar(50)
e_birthdate datetime NULL
e_birthcity varchar(50) NULL
e_birthcountry varchar(50) NULL
e_phone_number varchar(13) NULL
e_mail varchar(25) NULL
dbo.waiter
waiter_id int PK
first_name varchar(20) NULL
last_name varchar(20) NULL
user_name varchar(50)
password nvarchar(50)
tc_identity_num int NULL
birth_date datetime NULL
email varchar(25)
phone_number varchar(13) NULL
salary money NULL
dbo.employee_id int
last_name varchar(20)
user_name varchar(20) NULL
email varchar(25)
password nvarchar(50)
birth_date datetime
position varchar(20) NULL
salary money NULL
birth_country varchar(20) NULL
dbo.kithcen_staff
staff_id int PK
first_name varchar(20)
waiter_id int FK
chef_id int FK
last_name varchar(20)
gender varchar(10) NULL
email varchar(25) NULL
password nvarchar(25)
phone varchar(13) NULL
adress nvarchar(50) NULL
salary money
dbo.ingredient
ingredient_id int PK
ingredient_name varchar(40) NULL
trans_ID int FK
menu_id int FK
description varchar(100) NULL
amount decimal(18,0)
price smallmoney
detail varchar(50) NULL
dbo.transaction
trans_ID int PK
trans_report_num varchar(15) NULL
payment_id int FK
ingredient_id int FK
trans_date datetime
trans_report_date datetime NULL

2.2 Databas


ingredient
Column Name
Data Type
ingredient_id
int
ingredient_name
varchar(40)
description
varchar(100)
amount
decimal(18, 0)
price
smallmoney
detail
varchar(50)
payment
payment_id
int
ID_invoice
int
number
nvarchar(30)
type
varchar(20)
method
varchar(20)
payment_date
datetime
payment_amount
money
menu
menu_id
int
number
varchar(50)
details
varchar(300)
price
smallmoney
food_info
food_id
int
food_name
varchar(20)
food_status
varchar(20)
food_price
smallmoney
details
nvarchar(30)
reservation
reservation_id
int
reserv_date
datetime
reserv_time
datetime(7)
no_of_guest
int
reserv_table_no
varchar(2)
transaction
trans_id
int
trans_report_num
varchar(15)
trans_date
datetime
trans_report_date
datetime
customer
customer_id
int
user_name
varchar(50)
password
varchar(50)
first_name
varchar(20)
last_name
varchar(20)
tc_identity_num
int
phone_number
varchar(13)
mail_adress
nvarchar(30)
birth_date
datetime
country
varchar(30)
city
varchar(30)
restaurant status
status_id
int
status_name
varchar(6)
status_date
datetime
restaurant manager
manager_id
int
user_name
varchar(20)
password
varchar(20)
first_name
varchar(20)
last_name
varchar(20)
tc_identity_num
int
birth_date
nvarchar(15)
phone_number
varchar(13)
email
varchar(50)
home_adress
nvarchar(50)
city
varchar(25)
country
varchar(25)
crew
crew_id
int
crew_name
varchar(20)
crew_number
int
crew_total_salary
money
crew_user_id
int
employee
employee_id
int
e_firstname
varchar(25)
e_lastname
varchar(25)
e_gender
varchar(25)
e_age
int
e_salary
money
e_department
varchar(50)
e_birthdate
datetime
e_birthcity
varchar(50)
e_birthcountry
varchar(50)
e_phone_num...
varchar(13)
e_mail
varchar(25)
chef
chef_id
int
first_name
varchar(20)
last_name
varchar(20)
user_name
varchar(25)
email
nvarchar(20)
password
nvarchar(20)
birth_date
datetime
position
varchar(20)
salary
money
birth_country
varchar(20)
phone
varchar(13)
adress
nvarchar(50)
salary
money
waiter
waiter_id
int
first_name
varchar(20)
last_name
varchar(20)
user_name
varchar(50)
password
nvarchar(50)
tc_identity_num
int
birth_date
datetime
email
varchar(25)
phone_number
varchar(13)
salary
money

2.2 Database ER Diagram from MS SQL - Version 2



2.3 Datal

customer	tc_identity_num	TC identity Number of customer	int	9999			0-11	
	phone_number	Customer Phone Number	varchar(13)	999-999-99-99				
	mail_adress	Customer Mail Adress	nvarchar(30)	Xxxxxxxxx	Y			
	birth_date	Customer Birthday	datetime	YYYY-MM-DD hh:mm:ss[.nnn]				
	country	Resident Country of customer	varchar(30)	xxxxxxx				
	city	Resident City of customer	varchar(30)	xxxxxxx				
restaurant_manager	manager_id	Restaurant Manager ID number	int	9999	Y	PK		
	user_name	Restaurant Manager username of app	varchar(20)	xxxxxxx	Y	FK	menu	
	first_name	Restaurant Manager First Name	varchar(20)	Xxxxxxxxx		FK	employee	
	last_name	Restaurant Manager Last Name	varchar(20)	Xxxxxxxxx	Y			
	tc_identity_num	TC Identity Number of customer	int	9999		0-11		
	password	Restrautant Manager Account Password	nvarchar(15)	Xxxxxxxxx	Y			
	birth_date	Resturant Manager Birthday	datetime	YYYY-MM-DD hh:mm:ss[.nnn]				
	phone_number	Resturant Manager Phone Number	varchar(13)	999-999-99-99				
	email	Resturant Manager Email adress	varchar(50)	xxxxxxx				
	home_adress	Resturant Manager Residency adress	nvarchar(50)	Xxxxxxxxx				
	city	Resturant Manager Resident city	varchar(25)	xxxxxxx				
	country	Resturant Manager Resident country	varchar(20)	xxxxxxx				
employee	employe_id	Employee ID number	int	9999	Y	PK		
	e_lastname	Employee Lastname	varchar(25)	Xxxxxxxxx	Y	FK	kitchen_staff	
	e_firstname	Employee Firstname	varchar(25)	Xxxxxxxxx	Y	FK	restaurant_manager	
	e_gender	Employee Gender	varchar(25)	xxxxxxx				
	e_age	Employee Age	int	99				
	e_salary	Employee monthly salary amount	money	9999.99 ($)	Y			
	e_department	Employee working department name	varchar(50)	Xxxxxxxxx	Y			
	e_birthdate	Employee birthday	datetime	YYYY-MM-DD hh:mm:ss[.nnn]				
	e_birthcity	Employee nation city	varchar(50)	xxxxxxx				
	e_birthcountry	Employee birth country (residency)	varchar(50)	xxxxxxx				
	e_phone_number	Employee Phone Number	varchar(13)	999-999-99-99				
	e_mail	Employee legal mail address	varchar(25)	xxxxxxxxxxxxx				
crew	crew_id	ID number of crew (department each)	int	9999	Y	PK		
	crew_name	Crew name (e.g- waiters)	varchar(20)	Xxxxxxxxx		FK	employee	
	crew_number	Number of crew (each group number)	int	99	Y			
	crew_total_salary	Salary of each crew in total	money	99999.99 ($)				
	crew_user_id	User ID of ceach crew (group)	int	9999	Y			
kitchen_staff	staff_id	ID number of staff	int	99999	Y	PK		
	first_name	Staff Firstname	varchar(20)	Xxxxxxxxx	Y	FK	employee	
	last_name	Staff Lastname	varchar(20)	Xxxxxxxxx	Y			
	gender	Staff gender (each)	varchar(10)	xxxxxxx				
	email	Staff email (each)	varchar(25)	xxxxxxx				
	password	Staff user account password	nvarchar(25)	Xxxxxxxxx	Y			
	phone	Staff phone number	varchar(13)	999-999-99-99				
	address	Staff home adress (residency)	nvarchar(50)	xxxxxxxxxxxxx				
	salary	Staff salary (each person)	money	9999.99 ($)	Y			

Part I

	password	Chef password for app	nvarchar(50)	xxxxxxxxxxxxxx	Y				
	birth_date	Chef birthday	datetime	YYYY-MM-DD hh:mm:ss[.nnn]					
	position	Chef position in resturant (each dept)	varchar(20)	xxxxxx					
	salary	Chef salary (monthly)	money	9999.99 ($)	Y				
	birth_country	Chef birth country (residency)	varchar(20)	xxxxxx					
waiter	waiter_id	Waiter ID number	int	9999	Y		PK		
	first_name	Waiter First Name	varchar(20)	Xxxxxxxxx			FK	employee	
	last_name	Waiter Lastname	varchar(20)	Xxxxxxxxx			FK	kithchen_staff	
	user_name	Waiter username for app	varchar(50)	xxxxxx	Y				
	password	Waiter restaurant app password	nvarchar(50)	xxxxxxxxxxxxx	Y				
	tc_identity_num	TC identity Number of waiter	int	9999		0-11			
	birth_date	Waiter birthdate	datetime	YYYY-MM-DD hh:mm:ss[.nnn]					
	email	Waiter email adress (restaurant app)	varchar(25)	xxxxxxxxxxxxx	Y				
	phone_number	Waiter phone number	varchar(13)	999-999-99-99					
	salary	Waiter salary (monthly)	money	9999.99 ($)					
menu	menu_id	Menu ID	int	99	Y	0-50	PK		
	number	Menu Number	int	99					
	details	Deails of each menu (ingredients and etc)	varchar(300)	xxxxxxxxxxxxx					
	price	Price of Menu (each is seperate)	smallmoney	99.99 ($)	Y		FK	customer	
food_info	food_id	Meal ID	int	9999	Y		PK		
	food_name	Food spesific name	varchar(20)	Xxxxxxxxx					
	food_status	Status of meal (ready or not)	varchar(20)	xxxxxx					
	food_price	Food price (written in menu)	smallmoney	99.99 ($)	Y		FK	menu	
	details	Dealits about food and meal	nvarchar(50)	xxxxxxxxxxxxx	Y				
ingredient	ingredient_id	ID of ingredient related food	int	9999	Y		PK		
	ingredient_name	Ingredient name	varchar(40)	Xxxxxxxxx			FK	food_info	
	description	Ingredient description (extra)	varchar(100)	xxxxxxxxxxxxx					
	amount	Ingredient amount	decimal(18,0)	99,99	Y				
	price	Ingredient price (each)	smallmoney	99.99 ($)	Y				
	detail	Ingredient detail (if necessary)	varchar(50)	xxxxxxxxxxxxx					
reservation	reservation_id	Reservation ID	int	9999	Y		PK		
	reserv_date	Reservation Date	datetime	YYYY-MM-DD hh:mm:ss[.nnn]					
	reserv_time	Reservation hour (time)	datetime2(7)	hh:mm:ss[.nnn]	Y				
	no_of_guest	Number of guests for reserv table	int	999	Y		FK	customer	
	reserv_table_no	Reserved table number	varchar(2)	99		0-10	FK	payment	
payment	payment_id	Payment ID	int	9999	Y		PK		
	invoice_id	Payment ID of invoice	int	99999			FK	customer	
	number	Payment number	nvarchar(50)	xxxxxxx					
	type	Payment type (credit card or cash and etc)	varchar(20)	xxxxxxx					
	method	Method (cash in advance,hire purchase)	varchar(20)	xxxxxx					
	payment_date	Payment date	datetime	YYYY-MM-DD hh:mm:ss[.nnn]	Y		FK	reservation	
	payment_amount	Payment amount (e.g price)	money	99.99 ($)	Y				
delivery	delivery_id	Delivery ID	int	9999	Y		PK		
	delivery_number	Delivery no	smallint	99	Y				
	delivery_date	Delivery date	datetime2(7)	YYYY-MM-DD hh:mm:ss[.nnn]			FK	transaction	
	delivery_status	Delivery status	text	xxxxxxx	Y				
transaction	trans_ID	Transaction id number	int	9999	Y		PK		
	trans_report_no	Transaction Report number	varchar(15)	xxxxxxx			FK	delivery	
	trans_date	Transaction date	datetime	YYYY-MM-DD hh:mm:ss[.nnn]					
	trans_report_date	Transaction report entrance date	datetime	YYYY-MM-DD hh:mm:ss[.nnn]	Y				

General view - Data Dictionary

Table Name	Attribute Name	Contents	Type	Format	Required	Domain	PK/FK	FK Reference
customer	customer_id	Customer ID	int	9999	Y		PK	
	user_name	Customer Username	varchar(50)	xxxxxxx			FK	payment
	password	Customer Account Password	varchar(20)	xxxxxxx	Y			
	first_name	Customer Firstname	nvarchar(30)	xxxxxxxxxxxxxx	Y			
	tc_identity_num	TC Identity Number of customer	int	9999		0-11		
	phone_number	Customer Phone Number	varchar(13)	999-999-99-99				
	mail_adress	Customer Mail Adress	nvarchar(30)	Xxxxxxxxx	Y			
	birth_date	Customer Birthday	datetime	YYYY-MM-DD hh:mm:ss[.nnn]				
	country	Resident Country of customer	varchar(30)	xxxxxxx				
	city	Resident City of customer	varchar(30)	xxxxxxx				
restaurant_manager	manager_id	Restaurant Manager ID number	int	9999	Y		PK	
	user_name	Restaurant Manager username of app	varchar(20)	xxxxxxx	Y		FK	menu
	first_name	Restaurant Manager First Name	varchar(20)	Xxxxxxxxx			FK	employee
	last_name	Restaurant Manager Last Name	varchar(20)	Xxxxxxxxx	Y			
	tc_identity_num	TC Identity Number of customer	int	9999		0-11		
	password	Restrautant Manager Account Password	nvarchar(15)	Xxxxxxxxx	Y			
	birth_date	Resturant Manager Birthday	datetime	YYYY-MM-DD hh:mm:ss[.nnn]				
	phone_number	Resturant Manager Phone Number	varchar(13)	999-999-99-99				
	email	Resturant Manager Email adress	varchar(50)	xxxxxxx				
	home_adress	Resturant Manager Residency adress	nvarchar(50)	Xxxxxxxxx				
	city	Resturant Manager Resident city	varchar(25)	xxxxxxx				
	country	Resturant Manager Resident country	varchar(20)	xxxxxxx				
employee	employe_id	Employee ID number	int	9999	Y		PK	
	e_lastname	Employee Lastname	varchar(25)	Xxxxxxxxx	Y		FK	kitchen_staff
	e_firstname	Employee Firstname	varchar(25)	Xxxxxxxxx	Y		FK	restaurant_manager
	e_gender	Employee Gender	varchar(25)	xxxxxxx				
	e_age	Employee Age	int	99				
	e_salary	Employee monthly salary amount	money	9999.99 ($)	Y			
	e_department	Employee working department name	varchar(50)	Xxxxxxxxx	Y			
	e_birthdate	Employee birthday	datetime	YYYY-MM-DD hh:mm:ss[.nnn]				
	e_birthcity	Employee nation city	varchar(50)	xxxxxxx				
	e_birthcountry	Employee birth country (residency)	varchar(50)	xxxxxxx				
	e_phone_number	Employee Phone Number	varchar(13)	999-999-99-99				
	e_mail	Employee legal mail address	varchar(25)	xxxxxxxxxxxxxx				
crew	crew_id	ID number of crew (department each)	int	9999	Y		PK	
	crew_name	Crew name (e.g- waiters)	varchar(20)	Xxxxxxxxx			FK	employee
	crew_number	Number of crew (each group number)	int	99	Y			
	crew_total_salary	Salary of each crew in total	money	99999.99 ($)				
	crew_user_id	User ID of ceach crew (group)	int	9999	Y			
kithchen_staff	staff_id	ID number of staff	int	99999	Y		PK	
	first_name	Staff Firstname	varchar(20)	Xxxxxxxxx	Y		FK	employee
	last_name	Staff Lastname	varchar(20)	Xxxxxxxxx	Y			
	gender	Staff gender (each)	varchar(10)	xxxxxxx				
	email	Staff email (each)	varchar(25)	xxxxxxx				
	password	Staff user account password	nvarchar(25)	Xxxxxxxxx	Y			
	phone	Staff phone number	varchar(13)	999-999-99-99				
	address	Staff home adress (residency)	nvarchar(50)	xxxxxxxxxxxxxx				
	salary	Staff salary (each person)	money	9999.99 ($)	Y			
chef	chef_id	ID number of chef	int	9999	Y		PK	
	first_name	Chef Firstname	varchar(20)	Xxxxxxxxx	Y		FK	employee
	last_name	Chef Lastname	varchar(20)	Xxxxxxxxx	Y		FK	kithchen_staff
	user_name	Chef username for app	varchar(20)	xxxxxxx				
	email	Chef email adress for restaurant (company)	varchar(20)	xxxxxxxxxxxxx	Y			
	password	Chef password for app	nvarchar(50)	xxxxxxxxxxxxx	Y			
	birth_date	Chef birthday	datetime	YYYY-MM-DD hh:mm:ss[.nnn]				
	position	Chef position in resturant (each dept)	varchar(20)	xxxxxxx				
	salary	Chef salary (monthly)	money	9999.99 ($)	Y			
	birth_country	Chef birth country (residency)	varchar(20)	xxxxxxx				
waiter	waiter_id	Waiter ID number	int	9999	Y		PK	
	first_name	Waiter First Name	varchar(20)	Xxxxxxxxx			FK	employee
	last_name	Waiter Lastname	varchar(20)	Xxxxxxxxx			FK	kithchen_staff
	user_name	Waiter username for app	varchar(50)	xxxxxxx	Y			
	password	Waiter restaurant app password	nvarchar(50)	xxxxxxxxxxxxx	Y			
	tc_identity_num	TC Identity Number of waiter	int	9999		0-11		
	birth_date	Waiter birthdate	datetime	YYYY-MM-DD hh:mm:ss[.nnn]				
	email	Waiter email adress (restaurant app)	varchar(25)	xxxxxxxxxxxxx	Y			
	phone_number	Waiter phone number	varchar(13)	999-999-99-99				
	salary	Waiter salary (monthly)	money	9999.99 ($)				
menu	menu_id	Menu ID	int	99	Y	0-50	PK	
	number	Menu Number	int	99				
	details	Deails of each menu (ingredients and etc)	varchar(300)	xxxxxxxxxxxxx				
	price	Price of Menu (each is seperate)	smallmoney	99.99 ($)	Y		FK	customer
food_info	food_id	Meal ID	int	9999	Y		PK	
	food_name	Food spesific name	varchar(20)	Xxxxxxxxx				
	food_status	Status of meal (ready or not)	varchar(20)	xxxxxxx				
	food_price	Food price (written in menu)	smallmoney	99.99 ($)	Y		FK	menu
	details	Deailts about food and meal	nvarchar(50)	xxxxxxxxxxxxx	Y			
ingredient	ingredient_id	ID of ingredient related food	int	9999	Y		PK	
	ingredient_name	Ingredient name	varchar(40)	Xxxxxxxxx			FK	food_info
	description	Ingredient description (extra)	varchar(100)	xxxxxxxxxxxxx				
	amount	Ingredient amount	decimal(18,0)	99,99	Y			
	price	Ingredient price (each)	smallmoney	99.99 ($)	Y			
	detail	Ingredient detail (if necessary)	varchar(50)	xxxxxxxxxxxxx				
reservation	reservation_id	Reservation ID	int	9999	Y		PK	
	reserv_date	Reservation Date	datetime	YYYY-MM-DD hh:mm:ss[.nnn]				
	reserv_time	Reservation hour (time)	datetime2(7)	hh:mm:ss[.nnn]	Y			
	no_of_guest	Number of guests for reserv table	int	999	Y		FK	customer
	reserv_table_no	Reserved table number	varchar(2)	99		0-10	FK	payment
payment	payment_id	Payment ID	int	9999	Y		PK	
	invoice_id	Payment ID of invoice	int	99999			FK	customer
	number	Payment number	nvarchar(50)	xxxxxxx				
	type	Payment type (credit card or cash and etc)	varchar(20)	xxxxxxx				
	method	Method (cash in advance,hire purchase)	varchar(20)	xxxxxxx				
	payment_date	Payment date	datetime	YYYY-MM-DD hh:mm:ss[.nnn]	Y		FK	reservation
	payment_amount	Payment amount (e.g price)	money	99.99 ($)	Y			
delivery	delivery_id	Delivery ID	int	9999	Y		PK	
	delivery_number	Delivery no	smallint	99	Y			
	delivery_date	Delivery date	datetime2(7)	YYYY-MM-DD hh:mm:ss[.nnn]			FK	transaction
	delivery_status	Delivery status	text	xxxxxxx	Y			
transaction	trans_ID	Transaction id number	int	9999	Y		PK	
	trans_report_no	Transaction Report number	varchar(15)	xxxxxxx			FK	delivery
	trans_date	Transaction date	datetime	YYYY-MM-DD hh:mm:ss[.nnn]				
	trans_report_date	Transaction report entrance date	datetime	YYYY-MM-DD hh:mm:ss[.nnn]	Y			

PHASE II

MS SQL - DDL (CREATE, DROP, ALTER TABLE, INDEX)

1. CREATE TABLE Clauses for Each Table (Create Tables.sql file)

```
/****** Object:  Table [dbo].[chef] Script Date: 7.01.2023 23:39:37 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[chef](
	[chef_id] [int] NOT NULL,
	[first_name] [varchar](20) NOT NULL,
	[last_name] [varchar](20) NOT NULL,
	[user_name] [varchar](20) NULL,

	[email] [varchar](25) NOT NULL,
	[password] [nvarchar](20) NULL,
	[birth_date] [datetime] NOT NULL,
	[position] [varchar](20) NULL,
	[salary] [money] NULL,
	[birth_country] [varchar](20) NULL,
 CONSTRAINT [PK_chef] PRIMARY KEY CLUSTERED
(
	[chef_id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON
[PRIMARY]
) ON [PRIMARY]
GO
-----
/****** Object:  Table [dbo].[crew] Script Date: 7.01.2023 23:39:38 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[crew](
	[crew_id] [int] NOT NULL,
	[crew_name] [varchar](20) NULL,
	[crew_number] [int] NOT NULL,
	[crew_total_salary] [money] NULL,
	[crew_user_id] [int] NOT NULL,
CONSTRAINT [PK_crew] PRIMARY KEY CLUSTERED
(
	[crew_id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON
[PRIMARY]
) ON [PRIMARY]
GO
```

```
/****** Object:  Table [dbo].[delivery]    Script Date: 7.01.2023 23:39:38 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[delivery](
	[delivery_id] [int] NOT NULL,
	[delivery_number] [smallint] NULL,
	[delivery_date] [datetime2](7) NOT NULL,
	[delivery_status] [text] NULL,
CONSTRAINT [PK_delivery] PRIMARY KEY CLUSTERED
(
	[delivery_id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON
[PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
----
/****** Object:  Table [dbo].[employee]    Script Date: 7.01.2023 23:39:38 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[employee](
	[employee_id] [int] NOT NULL,
	[e_firstname] [varchar](25) NOT NULL,
	[e_lastname] [varchar](25) NOT NULL,
	[e_gender] [varchar](25) NULL,
	[e_age] [int] NULL,
	[e_salary] [money] NOT NULL,
	[e_department] [varchar](50) NOT NULL,
	[e_birthdate] [datetime] NULL,
	[e_birthcity] [varchar](50) NULL,
	[e_birthcountry] [varchar](50) NULL,
	[e_phone_number] [varchar](13) NULL,
	[e_mail] [varchar](25) NULL,
CONSTRAINT [PK_employee] PRIMARY KEY CLUSTERED
(
	[employee_id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON
[PRIMARY]
) ON [PRIMARY]
GO
```

```
/****** Object:  Table [dbo].[food_info]  Script Date: 7.01.2023 23:39:38 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[food_info](
	[food_id] [int] NOT NULL,
	[food_name] [varchar](20) NULL,
	[food_status] [varchar](20) NULL,
	[food_price] [smallmoney] NULL,
	[details] [nvarchar](50) NOT NULL,
CONSTRAINT [PK_food_info] PRIMARY KEY CLUSTERED
(
	[food_id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON
[PRIMARY]
) ON [PRIMARY]
GO
```

```
/****** Object:  Table [dbo].[ingredient] Script Date: 7.01.2023 23:39:38 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[ingredient](
	[ingredient_id] [int] NOT NULL,
	[ingredient_name] [varchar](40) NULL,
	[description] [varchar](100) NULL,
	[amount] [decimal](18, 0) NOT NULL,
	[price] [smallmoney] NOT NULL,
	[detail] [varchar](50) NULL,
CONSTRAINT [PK_ingredient] PRIMARY KEY CLUSTERED
(
	[ingredient_id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON
[PRIMARY]
) ON [PRIMARY]
GO
```

```
/****** Object:  Table [dbo].[kitchen_staff]    Script Date: 7.01.2023 23:39:38 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[kitchen_staff](
	[staff_id] [int] NOT NULL,
	[first_name] [varchar](20) NOT NULL,
	[last_name] [varchar](20) NOT NULL,
	[gender] [varchar](10) NULL,
	[email] [varchar](25) NULL,
	[password] [nvarchar](25) NULL,
	[phone] [varchar](13) NULL,
	[adress] [nvarchar](50) NULL,
	[salary] [money] NOT NULL,
CONSTRAINT [PK_kitchen_staff] PRIMARY KEY CLUSTERED
(
	[staff_id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON
[PRIMARY]
) ON [PRIMARY]
GO
```

```
/****** Object:  Table [dbo].[menu] Script Date: 7.01.2023 23:39:38 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[menu](
	[menu_id] [int] NOT NULL,
	[number] [varchar](50) NULL,
	[details] [varchar](300) NULL,
	[price] [smallmoney] NOT NULL,
CONSTRAINT [PK_menu] PRIMARY KEY CLUSTERED
(
	[menu_id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON
[PRIMARY]
) ON [PRIMARY]
GO
```

```
/****** Object:  Table [dbo].[payment]    Script Date: 7.01.2023 23:39:38 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[payment](
	[payment_id] [int] NOT NULL,
	[ID_invoice] [int] NULL,
	[number] [nvarchar](50) NULL,
	[type] [varchar](20) NULL,
	[method] [varchar](20) NULL,
	[payment_date] [datetime] NOT NULL,
	[payment_amount] [money] NOT NULL,
CONSTRAINT [PK_payment] PRIMARY KEY CLUSTERED
(
	[payment_id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON
[PRIMARY]
) ON [PRIMARY]
GO
```

```
/****** Object:  Table [dbo].[reservation]    Script Date: 7.01.2023 23:39:38 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[reservation](
	[reservation_id] [int] NOT NULL,
	[reserv_date] [datetime] NULL,
	[reserv_time] [datetime2](7) NOT NULL,
	[no_of_guest] [int] NOT NULL,
	[reserv_table_no] [varchar](2) NULL,
 CONSTRAINT [PK_reservation] PRIMARY KEY CLUSTERED
(
	[reservation_id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON
[PRIMARY]
) ON [PRIMARY]
GO
```

```
/****** Object:  Table [dbo].[restaurant_manager]    Script Date: 7.01.2023 23:39:38
```

```
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[restaurant_manager](
	[manager_id] [int] NOT NULL,
	[user_name] [varchar](20) NOT NULL,
	[first_name] [varchar](20) NOT NULL,
	[last_name] [varchar](20) NOT NULL,
	[tc_identity_num] [int] NULL,
	[password] [nvarchar](15) NOT NULL,
	[birth_date] [datetime] NULL,
	[phone_number] [varchar](13) NULL,
	[email] [varchar](50) NOT NULL,
	[home_adress] [nvarchar](50) NULL,
	[city] [varchar](25) NULL,
	[country] [varchar](25) NULL,
 CONSTRAINT [PK_restaurant_manager] PRIMARY KEY CLUSTERED
(
	[manager_id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON
[PRIMARY]
) ON [PRIMARY]
GO
```

```
/****** Object:  Table [dbo].[restaurant_status]    Script Date: 7.01.2023 23:39:38
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[restaurant_status](
	[status_id] [int] NOT NULL,
	[status_name] [varchar](6) NOT NULL,
	[status_date] [datetime] NULL,
 CONSTRAINT [PK_restaurant_status] PRIMARY KEY CLUSTERED
(
	[status_id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON
[PRIMARY]
) ON [PRIMARY]
GO
```

```
/****** Object:  Table [dbo].[transaction]    Script Date: 7.01.2023 23:39:38 ******/
```

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[transaction](
	[trans_id] [int] NOT NULL,
	[trans_report_num] [varchar](15) NULL,
	[trans_date] [datetime] NULL,
	[trans_report_date] [datetime] NOT NULL,
 CONSTRAINT [PK_transaction] PRIMARY KEY CLUSTERED
(
	[trans_id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON
[PRIMARY]
) ON [PRIMARY]
GO
```

```
/****** Object:  Table [dbo].[waiter]    Script Date: 7.01.2023 23:39:38 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[waiter](
	[waiter_id] [int] NOT NULL,
	[first_name] [varchar](20) NULL,
	[last_name] [varchar](20) NULL,
	[user_name] [varchar](50) NOT NULL,
	[password] [nvarchar](50) NOT NULL,
	[tc_identity_num] [int] NULL,
	[birth_date] [datetime] NULL,
	[email] [varchar](25) NOT NULL,
	[phone_number] [varchar](13) NULL,
	[salary] [money] NULL,
 CONSTRAINT [PK_waiter] PRIMARY KEY CLUSTERED
(
	[waiter_id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON
[PRIMARY]
) ON [PRIMARY]
GO
```

```
/****** Object:  Table [dbo].[customer]    Script Date: 7.01.2023 23:39:38 ******/
```

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[customer](
	[customer_id] [int] NOT NULL,
	[user_name] [varchar](50) NULL,
	[password] [varchar](50) NOT NULL,
	[first_name] [varchar](20) NOT NULL,
	[last_name] [varchar](20) NOT NULL,
	[tc_identity_num] [int] NULL,
	[phone_number] [varchar](13) NULL,
	[mail_adress] [nvarchar](30) NOT NULL,
	[birth_date] [datetime] NULL,
	[country] [varchar](30) NULL,
	[city] [varchar](30) NULL,
 CONSTRAINT [PK_customer] PRIMARY KEY CLUSTERED
(
	[customer_id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON
[PRIMARY]
) ON [PRIMARY]
GO
```

2. ALTER TABLE Clauses (Alter_Table_Statements.sql)

- 1. Add a new column called gender to customer table:

```
-- Before adding a new column the chef table,
```

	chef_id	first_name	last_name	user_name	email	password	birth_date	position	salary	birth_country
1	22336677	Emin	Mammadov	eminchik085	eminmmadov0202@gmail.com	eminchik_085	2002-02-02 00:00:00.000	lead chef	10000,00	Azerbaijan
2	112233445	John	Doe	jdoe1	jdoe1@gmail.com	password1	1994-06-20 00:00:00.000	chef	4000,00	USA
3	334455667	Bob	Smith	bsmith1	bsmith1@gmail.com	password3	1990-03-22 00:00:00.000	chef de partie	3000,00	Canada
4	445566778	Alice	Smith	asmith1	asmith1@gmail.com	password4	1995-04-30 00:00:00.000	commis	2500,00	Canada
5	556677889	James	Bond	jbond1	jbond1@gmail.com	password5	2000-05-19 00:00:00.000	chef	4000,00	UK

```
ALTER TABLE dbo.chef
ADD gender bit
-- After adding a new column the chef table,
```

	chef_id	first_name	last_name	user_name	email	password	birth_date	position	salary	birth_country	gender
1	22336677	Emin	Mammadov	eminchik085	eminmmadov0202@gmail.com	eminchik_085	2002-02-02 00:00:00.000	lead chef	10000,00	Azerbaijan	NULL
2	112233445	John	Doe	jdoe1	jdoe1@gmail.com	password1	1994-06-20 00:00:00.000	chef	4000,00	USA	NULL
3	334455667	Bob	Smith	bsmith1	bsmith1@gmail.com	password3	1990-03-22 00:00:00.000	chef de partie	3000,00	Canada	NULL
4	445566778	Alice	Smith	asmith1	asmith1@gmail.com	password4	1995-04-30 00:00:00.000	commis	2500,00	Canada	NULL
5	556677889	James	Bond	jbond1	jbond1@gmail.com	password5	2000-05-19 00:00:00.000	chef	4000,00	UK	NULL

- 2. Drop “city” column from customer table:

```
-- Before dropping the column,
```

	customer_id	user_name	password	first_name	last_name	tc_identity_num	phone_number	mail_adress	birth_date	country	city
1	1	cust1	pass1	Nazrin	Huseynli	997896543	123-456-7890	hnazrin26@gmail.com	2000-02-26 00:00:00.000	Azerbaijan	Baku
2	2	cust2	pass2	Alex	Parker	123456789	234-567-8901	aparker1@gmail.com	1995-03-17 00:00:00.000	USA	New York
3	3	cust3	pass3	Samantha	James	234567890	345-678-9012	sjames1@gmail.com	1990-04-08 00:00:00.000	Canada	Toronto

	customer_id	user_name	password	first_name	last_name	tc_identity_num	phone_number	mail_adress	birth_date	country
1	1	cust1	pass1	Nazrin	Huseynli	997896543	123-456-7890	hnazrin26@gmail.com	2000-02-26 00:00:00.000	Azerbaijan
2	2	cust2	pass2	Alex	Parker	123456789	234-567-8901	aparker1@gmail.com	1995-03-17 00:00:00.000	USA
3	3	cust3	pass3	Samantha	James	234567890	345-678-9012	sjames1@gmail.com	1990-04-08 00:00:00.000	Canada
4	4	cust4	pass4	Michael	Brown	345678901	456-789-0123	mbrown1@gmail.com	1985-05-30 00:00:00.000	UK
5	5	cust5	pass5	Emily	Smith	456789012	567-890-1234	esmith1@gmail.com	1980-06-20 00:00:00.000	Australia
6	6	cust6	pass6	William	Johnson	567890123	678-901-2345	wjohnson1@gmail.com	1975-07-11 00:00:00.000	USA
7	7	cust7	pass7	Ashley	Williams	678901234	789-012-3456	awilliams1@gmail.com	1970-08-01 00:00:00.000	Canada

SQLQuery20.sql - D...MFDL\leyviya (61)) SQLQuery19.sql - D...MF

Column Name	Data Type	Allow Nulls
food_id	int	☐
food_name	varchar(20)	☑
food_status	varchar(20)	☑
food_price	smallmoney	☑
details	nvarchar(300)	☐
		☐

```
ALTER TABLE dbo.food_info
ALTER COLUMN details nvarchar(250)
-- food_info table after,
```

Column Name	Data Type	Allow Nulls
food_id	int	☐
food_name	varchar(20)	☑
food_status	varchar(20)	☑
food_price	smallmoney	☑
details	nvarchar(250)	☑

- 5. Add a check constraint to the “food_price” to not to be below 0.1.

```
ALTER TABLE dbo.food_info
ADD CHECK (food_price>0.1)
```

3. 2 DROP TABLE Clauses (Drop_Statements.sql)

- 1. Delete the birth_date column from the “customer” table.

```
ALTER TABLE customer DROP COLUMN birth_date
```

- 2. Delete the “gender” column from the “chef” table.

```
ALTER TABLE dbo.chef DROP COLUMN gender
```

4. 5 INDICES (Index_Statements.sql)

- 1. Create an index named 'ix_cname' on the 'first_name' and 'last_name' columns in the customer table.

```
CREATE INDEX ix_cname
ON customer (first_name, last_name)
```

- 2. Create an index named 'ix_cname' on the 'user_name' column in the "chef" table.

```
CREATE UNIQUE INDEX ix_cname
ON chef (user_name)
```

- 3. Create an index named 'ix_fname' on the 'food_name' column on the "food_info" table.

```
CREATE UNIQUE INDEX ix_fname
ON food_info (food_name)
```

- 4. Create an index named 'ix_crname' on the "crew_number" column on the "crew" table.

```
CREATE UNIQUE INDEX ix_crname
ON crew (crew_number)
```

- 5. Create an index named 'ix_iname' on the "ingredient_name" column on the "ingredient" table.

```
CREATE UNIQUE INDEX ix_iname
ON ingredient (ingredient_name)
```

DML (INSERT, UPDATE, DELETE COMMANDS)

1. 5 INSERT commands for tables (Insert_Initial_Values.sql file)

```
INSERT INTO dbo.chef
(chef_id, first_name, last_name, user_name,
email, password, birth_date, position,
salary, birth_country)
VALUES
  ('445566778', 'Alice', 'Smith',
'asmith1', 'asmith1@gmail.com',
'password4', '04-30-1995', 'commis',
'2500', 'Canada')

INSERT INTO dbo.chef
(chef_id, first_name, last_name, user_name,
```

```
INSERT INTO
RESTAURANT_MANAGEMENT_SYSTEM.dbo.employee
(employee_id, e_firstname, e_lastname,
e_gender, e_age, e_salary, e_department,
e_birthdate, e_birthcity, e_birthcountry,
e_phone_number, e_mail)
VALUES
  (1, 'Nazrin', 'Huseynli', 'M', '23', '3500',
'Front of House', '02-26-2000', 'Baku',
'Azerbaijan', '123-456-7890',
'hnazrin26@gmail.com')
```

```
email, password, birth_date, position,
salary, birth_country)
VALUES
 ('556677889', 'James', 'Bond', 'jbond1',
'jbond1@gmail.com', 'password5', '05-19-
2000', 'chef', '4000', 'UK')

 INSERT INTO dbo.chef
(chef_id, first_name, last_name, user_name,
email, password, birth_date, position,
salary, birth_country)
VALUES
('667788901', 'Emma', 'Watson', 'ewatson1',
'ewatson1@gmail.com', 'password6', '06-26-
2005', 'sous chef', '3500', 'UK')

INSERT INTO dbo.chef
(chef_id, first_name, last_name, user_name,
email, password, birth_date, position,
salary, birth_country)
VALUES
('334455667', 'Bob', 'Smith', 'bsmith1',
'bsmith1@gmail.com', 'password3', '03-22-
1990', 'chef de partie', '3000', 'Canada')

INSERT INTO dbo.chef
(chef_id, first_name, last_name, user_name,
email, password, birth_date, position,
salary, birth_country)
VALUES
('22336677', 'Emin', 'Mammadov',
'eminchik085', 'eminmmadov0202@gmail.com',
'eminchik085', '02-02-2002', 'lead chef',
'10000', 'Azerbaijan')

INSERT INTO dbo.chef
(chef_id, first_name, last_name, user_name,
email, password, birth_date, position,
salary, birth_country)
-----------------------------------------
INSERT INTO dbo.customer (customer_id,
user_name, password, first_name, last_name,
tc_identity_num, phone_number, mail_adress,
birth_date, country, city)
VALUES
  (1, 'cust1', 'pass1', 'Nazrin',
'Huseynli', '997896543', '123-456-7890',
'hnazrin26@gmail.com', '02-26-2000',
'Azerbaijan', 'Baku')
```

```
INSERT INTO
RESTAURANT_MANAGEMENT_SYSTEM.dbo.employee
(employee_id, e_firstname, e_lastname,
e_gender, e_age, e_salary, e_department,
e_birthdate, e_birthcity, e_birthcountry,
e_phone_number, e_mail)
VALUES
  (2, 'Alex', 'Parker', 'M', '25', '3000',
'Back of House', '03-17-1995', 'New York',
'USA', '234-567-8901', 'aparker1@gmail.com')

INSERT INTO
RESTAURANT_MANAGEMENT_SYSTEM.dbo.employee
(employee_id, e_firstname, e_lastname,
e_gender, e_age, e_salary, e_department,
e_birthdate, e_birthcity, e_birthcountry,
e_phone_number, e_mail)
VALUES
  (3, 'Samantha', 'James', 'F', '30', '2500',
'Bakery', '04-08-1990', 'Toronto', 'Canada',
'345-678-9012', 'sjames1@gmail.com')
INSERT INTO
RESTAURANT_MANAGEMENT_SYSTEM.dbo.employee
(employee_id, e_firstname, e_lastname,
e_gender, e_age, e_salary, e_department,
e_birthdate, e_birthcity, e_birthcountry,
e_phone_number, e_mail)
VALUES
  (4, 'Michael', 'Brown', 'M', '35', '2000',
'Bar', '05-30-1985', 'London', 'UK', '456-789-
0123', 'mbrown1@gmail.com')
INSERT INTO
RESTAURANT_MANAGEMENT_SYSTEM.dbo.employee
(employee_id, e_firstname, e_lastname,
e_gender, e_age, e_salary, e_department,
e_birthdate, e_birthcity, e_birthcountry,
e_phone_number, e_mail)
VALUES
  (5, 'Emily', 'Smith', 'F', '40', '3500',
'Management', '06-20-1980', 'Sydney',
'Australia', '567-890-1234',
'esmith1@gmail.com')

INSERT INTO
RESTAURANT_MANAGEMENT_SYSTEM.dbo.employee
(employee_id, e_firstname, e_lastname,
e_gender, e_age, e_salary, e_department,
e_birthdate, e_birthcity, e_birthcountry,
e_phone_number, e_mail)
```

```
INSERT INTO dbo.customer (customer_id,
user_name, password, first_name, last_name,
tc_identity_num, phone_number, mail_adress,
birth_date, country, city)
VALUES
  (2, 'cust2', 'pass2', 'Alex', 'Parker',
'123456789', '234-567-8901',
'aparker1@gmail.com', '03-17-1995', 'USA',
'New York')

INSERT INTO dbo.customer (customer_id,
user_name, password, first_name, last_name,
tc_identity_num, phone_number, mail_adress,
birth_date, country, city)
VALUES
  (3, 'cust3', 'pass3', 'Samantha',
'James', '234567890', '345-678-9012',
'sjames1@gmail.com', '04-08-1990',
'Canada', 'Toronto')
INSERT INTO dbo.customer (customer_id,
user_name, password, first_name, last_name,
tc_identity_num, phone_number, mail_adress,
birth_date, country, city)
VALUES
  (4, 'cust4', 'pass4', 'Michael', 'Brown',
'345678901', '456-789-0123',
'mbrown1@gmail.com', '05-30-1985', 'UK',
'London')
INSERT INTO dbo.customer (customer_id,
user_name, password, first_name, last_name,
tc_identity_num, phone_number, mail_adress,
birth_date, country, city)
VALUES
  (5, 'cust5', 'pass5', 'Emily', 'Smith',
'456789012', '567-890-1234',
'esmith1@gmail.com', '06-20-1980',
'Australia', 'Sydney')

INSERT INTO dbo.customer (customer_id,
user_name, password, first_name, last_name,
tc_identity_num, phone_number, mail_adress,
birth_date, country, city).
```

```
----------------------------------------
INSERT INTO dbo.food_info (food_id, food_name,
food_status, food_price, details)
VALUES
  (1, 'Pizza Margherita', 'available', 10, 'A
classic pizza with tomato sauce, mozzarella
cheese, and fresh basil')
INSERT INTO dbo.food_info (food_id, food_name,
food_status, food_price, details)
VALUES
  (2, 'Spaghetti Bolognese', 'available', 12,
'Spaghetti noodles with a rich meat sauce made
with ground beef, onions, and tomatoes')
INSERT INTO dbo.food_info (food_id, food_name,
food_status, food_price, details)
VALUES
  (3, 'Lasagna', 'available', 15, 'Layers of
pasta, ground beef, and cheese, baked in a
rich tomato sauce')
INSERT INTO dbo.food_info (food_id, food_name,
food_status, food_price, details)
VALUES
  (4, 'Fish & Chips', 'available', 14,
'Battered and fried cod served with fries')
INSERT INTO dbo.food_info (food_id, food_name,
food_status, food_price, details)
VALUES
  (5, 'Roast Beef', 'available', 18, 'Tender
roast beef served with roast potatoes and
vegetables')
INSERT INTO dbo.food_info (food_id, food_name,
food_status, food_price, details).
```

2. 5 UPDATE commands for tables (Update Statements.sql)

1. Restaurant Should edit an order's status as 'On the way', 'preparing', 'ready'.

```
-- A restaurant wants to change an order whose order id is 1

-- The order type of the order(status_name)  before change,

SELECT status_name, status_date FROM restaurant_status
INNER JOIN food_info
ON status_id = food_id
WHERE status_id = 1
```

	status_name	status_date
1	available	2022-01-01 00:00:00.000

```
-- Update order's status to "preparing".

UPDATE restaurant_status
SET status_name = ('preparing')
WHERE status_name = ('available')

-- Printed again
```

Results Messages

	status_id	status_name	status_date
1	1	preparing	2022-01-01 00:00:00.000
2	2	unavailable	2022-01-02 00:00:00.000
3	3	preparing	2022-01-03 00:00:00.000
4	4	unavailable	2022-01-04 00:00:00.000
5	5	preparing	2022-01-05 00:00:00.000
6	6	unavailable	2022-01-06 00:00:00.000

2. Restaurants can change the prices of food items in "menu".

```
- - First, let's print the current price of the number 1 menu.
```

	menu_id	number	details	price
1	1	1	Spaghetti Bolognese	15,99

```
SELECT menu.menu_id, menu.number, menu.price
FROM menu
INNER JOIN order_menu
ON menu.menu_id = order_menu.menu_id
WHERE menu.menu_id = 2;
```

	menu_id	number	price
1	2	2	9,99
2	2	2	9,99
3	2	2	9,99
4	2	2	9,99
5	2	2	9,99
6	2	2	9,99

```
--Update price of menu 1 to 20.00.

UPDATE menu
SET price = 20.00
WHERE menu_id = 2

-- Print again
```

	menu_id	number	details	price
1	1	1	Spaghetti Bolognese	15,99
2	2	2	Grilled Cheese Sandwich	20,00
3	3	3	BLT Sandwich	12,99
4	4	4	Chicken Caesar Salad	14,99
5	5	5	Beef Stroganoff	17,99
6	6	6	Parmesan-Crusted Chicken	16,99
7	7	7	Fish and Chips	13,99
8	8	8	Hamburger	11,99
9	9	9	Chicken Fajitas	15,99
10	10	10	Pizza Margherita	13,99

3. Restaurant suddenly wants to increase all the foods' prices on the menu by 20%.

```
-- Food's prices before update
```

	menu_id	number	details	price
1	1	1	Spaghetti Bolognese	15,99
2	2	2	Grilled Cheese Sandwich	20,00
3	3	3	BLT Sandwich	12,99
4	4	4	Chicken Caesar Salad	14,99
5	5	5	Beef Stroganoff	17,99
6	6	6	Parmesan-Crusted Chicken	16,99
7	7	7	Fish and Chips	13,99
8	8	8	Hamburger	11,99
9	9	9	Chicken Fajitas	15,99
10	10	10	Pizza Margherita	13,99

```
-- Increase all the menu's prices by 20%

UPDATE food
SET price = price * 1.1
WHERE restaurant_id = 3
UPDATE menu
SET price = price * 2.2
WHERE menu_id = 1
UPDATE menu
SET price = price * 2.2
WHERE menu_id = 2
UPDATE menu
SET price = price * 2.2
WHERE menu_id = 3
UPDATE menu
SET price = price * 2.2
WHERE menu_id = 4
UPDATE menu
SET price = price * 2.2
WHERE menu_id = 5
UPDATE menu
SET price = price * 2.2
WHERE menu_id = 6
UPDATE menu
SET price = price * 2.2
WHERE menu_id = 7
UPDATE menu
SET price = price * 2.2
WHERE menu_id = 8
UPDATE menu
SET price = price * 2.2
WHERE menu_id = 9
UPDATE menu
```

```
SET price = price * 2.2
WHERE menu_id = 10

-- Food's prices after the update
```

	menu_id	number	details	price
1	1	1	Spaghetti Bolognese	35,178
2	2	2	Grilled Cheese Sandwich	44,00
3	3	3	BLT Sandwich	28,578
4	4	4	Chicken Caesar Salad	32,978
5	5	5	Beef Stroganoff	39,578
6	6	6	Parmesan-Crusted Chicken	37,378
7	7	7	Fish and Chips	30,778
8	8	8	Hamburger	26,378
9	9	9	Chicken Fajitas	35,178
10	10	10	Pizza Margherita	67,7116

4. If a customer's phone number doesn't start with '+90', add '+90' to customer's phone number.

```
-- Find whose phone number doesn't start with +90

SELECT id, first_name, last_name, phone_number FROM customer
WHERE phone_number NOT LIKE '+90%'
```

	customer_id	first_name	last_name	phone_number
1	5	Emily	Smith	5678901234
2	6	William	Johnson	6789012345
3	7	Ashley	Williams	7890123456

```
-- Update phone numbers which don't start with +90
UPDATE customer
SET phone_number = '+90' + phone_number
WHERE phone_number NOT LIKE '+90%'
```

	customer_id	user_name	password	first_name	last_name	tc_identity_num	phone_number	mail_adress	country

```
-- After the update,
```

	customer_id	user_name	password	first_name	last_name	tc_identity_num	phone_number	mail_adress	country
1	1	cust1	pass1	Nazrin	Huseynli	997896543	+901234567890	hnazrin26@gmail.com	Azerbaijan
2	2	cust2	pass2	Alex	Parker	123456789	+905525678901	aparker1@gmail.com	USA
3	3	cust3	pass3	Samantha	James	234567890	+905525365185	sjames1@gmail.com	Canada
4	4	cust4	pass4	Michael	Brown	345678901	+905567890123	mbrown1@gmail.com	UK
5	5	cust5	pass5	Emily	Smith	456789012	+905678901234	esmith1@gmail.com	Australia
6	6	cust6	pass6	William	Johnson	567890123	+906789012345	wjohnson1@gmail.com	USA
7	7	cust7	pass7	Ashley	Williams	678901234	+907890123456	awilliams1@gmail.com	Canada

5. If an employee wanted to update the “birth_country” for their web application account. (data info)

```
-Before the update of the employee “e_birthcity” column in the table “employee”.
```

	employee_id	e_firstname	e_lastname	e_gender	e_age	e_salary	e_department	e_birthdate	e_birthcity	e_birthcountry	e_phone_number	e_mail
1	1	Nazrin	Huseynli	M	23	3500,00	Front of House	2000-02-26 00:00:00.000	Baku	Azerbaijan	123-456-7890	hnazrin26@gmail.com
2	2	Alex	Parker	M	25	3000,00	Back of House	1995-03-17 00:00:00.000	New York	USA	234-567-8901	aparker1@gmail.com
3	3	Samantha	James	F	30	2500,00	Bakery	1990-04-08 00:00:00.000	Toronto	Canada	345-678-9012	sjames1@gmail.com
4	4	Michael	Brown	M	35	2000,00	Bar	1985-05-30 00:00:00.000	London	UK	456-789-0123	mbrown1@gmail.com
5	5	Emily	Smith	F	40	3500,00	Management	1980-06-20 00:00:00.000	Sydney	Australia	567-890-1234	esmith1@gmail.com
6	6	William	Johnson	M	45	3000,00	Accounting	1975-07-11 00:00:00.000	Chicago	USA	678-901-2345	wjohnson1@gmail.com
7	7	Ashley	Williams	F	50	2500,00	Front of House	1970-08-01 00:00:00.000	Toronto	Canada	789-012-3456	awilliams1@gmail.com
8	8	David	Jones	M	55	2000,00	Back of House	1965-09-21 00:00:00.000	London	UK	890-123-4567	djones1@gmail.com
9	9	Sara	Davis	F	60	3500,00	Bakery	1960-10-11 00:00:00.000	Sydney	Australia	901-234-5678	sdavis1@gmail.com

```
- - After the update of the employee “e_birthcity” column in the table “employee”.

UPDATE dbo.employee
SET e_birthcity = 'Gandja'
WHERE employee_id = 1;
```

	employee_id	e_firstname	e_lastname	e_gender	e_age	e_salary	e_department	e_birthdate	e_birthcity	e_birthcountry	e_phone_number	e_mail
1	1	Nazrin	Huseynli	M	23	3500,00	Front of House	2000-02-26 00:00:00.000	Gandja	Azerbaijan	123-456-7890	hnazrin26@gmail.com
2	2	Alex	Parker	M	25	3000,00	Back of House	1995-03-17 00:00:00.000	New York	USA	234-567-8901	aparker1@gmail.com
3	3	Samantha	James	F	30	2500,00	Bakery	1990-04-08 00:00:00.000	Toronto	Canada	345-678-9012	sjames1@gmail.com
4	4	Michael	Brown	M	35	2000,00	Bar	1985-05-30 00:00:00.000	London	UK	456-789-0123	mbrown1@gmail.com
5	5	Emily	Smith	F	40	3500,00	Management	1980-06-20 00:00:00.000	Sydney	Australia	567-890-1234	esmith1@gmail.com
6	6	William	Johnson	M	45	3000,00	Accounting	1975-07-11 00:00:00.000	Chicago	USA	678-901-2345	wjohnson1@gmail.com
7	7	Ashley	Williams	F	50	2500,00	Front of House	1970-08-01 00:00:00.000	Toronto	Canada	789-012-3456	awilliams1@gmail.com
8	8	David	Jones	M	55	2000,00	Back of House	1965-09-21 00:00:00.000	London	UK	890-123-4567	djones1@gmail.com
9	9	Sara	Davis	F	60	3500,00	Bakery	1960-10-11 00:00:00.000	Sydney	Australia	901-234-5678	sdavis1@gmail.com

3. 2 DELETE commands for tables ((Delete_Statements.sql)

1. Delete an ingredient from “ingredients” table.

```
- - Before deleting any data
```

	ingredient_id	ingredient_name	description	amount	price	detail
1	1	Flour	All-purpose flour	5	3,99	A staple ingredient for baking and cooking
2	2	Sugar	Granulated white sugar	10	7,99	A sweetener commonly used in baking and cooking
3	3	Butter	Unsalted butter	2	4,99	A common ingredient used in baking and cooking fo...
4	4	Eggs	Large eggs	12	3,49	A versatile ingredient commonly used in baking and ...
5	5	Salt	Fine sea salt	1	2,99	A seasoning commonly used in cooking to enhance ...
6	6	Pepper	Black peppercorns	1	4,99	A common spice used to add flavor and heat to dishes

```
DELETE FROM dbo.ingredient
WHERE ingredient_id = 3
-- After deleting ingredient 3 which is "Butter" and ingredient id=3
```

	ingredient_id	ingredient_name	description	amount	price	detail
1	1	Flour	All-purpose flour	5	3,99	A staple ingredient for baking and cooking
2	2	Sugar	Granulated white sugar	10	7,99	A sweetener commonly used in baking and cooking
3	4	Eggs	Large eggs	12	3,49	A versatile ingredient commonly used in baking and ...
4	5	Salt	Fine sea salt	1	2,99	A seasoning commonly used in cooking to enhance ...
5	6	Pepper	Black peppercorns	1	4,99	A common spice used to add flavor and heat to dishes

2. Delete a crew data from the "crew" table.

```
- - Before deleting any data
```

	crew_id	crew_name	crew_number	crew_total_salary	crew_user_id
1	1	Brunch	1	10000,00	1
2	2	Night shift	2	20000,00	2
3	3	Bakery	3	15000,00	3
4	4	Bar	4	12000,00	4
5	5	Management	5	25000,00	5
6	6	Accounting	6	20000,00	6

```
- After deleting crew_id=1
DELETE FROM dbo.crew
WHERE crew_id = 1
```

	crew_id	crew_name	crew_number	crew_total_salary	crew_user_id
1	2	Night shift	2	20000,00	2
2	3	Bakery	3	15000,00	3
3	4	Bar	4	12000,00	4
4	5	Management	5	25000,00	5
5	6	Accounting	6	20000,00	6

DQL (25 SQL SELECT COMMANDS & OUTPUTS)

All clauses are in the Queries.sql file.

```
1. --1- ordering according to names
```

```
SELECT customer_id, user_name, last_name FROM dbo.customer ORDER BY
user_name ASC
```

	customer_id	user_name	last_name
1	1	cust1	Huseynli
2	2	cust2	Parker
3	3	cust3	James
4	4	cust4	Brown
5	5	cust5	Smith
6	6	cust6	Johnson
7	7	cust7	Williams

2.
```
--2- left join from employee crew table (employee_id values)

SELECT employee.employee_id, employee.e_department,
employee.e_lastname FROM employee LEFT JOIN crew ON crew.crew_id =
employee.employee_id
```

	employee_id	e_department	e_lastname
1	1	Front of House	Huseynli
2	2	Back of House	Parker
3	3	Bakery	James
4	4	Bar	Brown
5	5	Management	Smith
6	6	Accounting	Johnson
7	7	Front of House	Williams
8	8	Back of House	Jones
9	9	Bakery	Davis

3.
```
--3- full join to see food_info's status.
SELECT * FROM food_info FULL OUTER JOIN restaurant_status ON
restaurant_status.status_name = restaurant_status.status_name
```

	food_id	food_name	food_status	food_price	details	status_id	status_name	status_date
1	1	2	available	10,00	A classic pizza with tomato sauce, mozzarella che...	1	preparing	2022-01-01 00:00:00.000
2	1	2	available	10,00	A classic pizza with tomato sauce, mozzarella che...	2	unavailable	2022-01-02 00:00:00.000
3	1	2	available	10,00	A classic pizza with tomato sauce, mozzarella che...	3	preparing	2022-01-03 00:00:00.000
4	1	2	available	10,00	A classic pizza with tomato sauce, mozzarella che...	4	unavailable	2022-01-04 00:00:00.000
5	1	2	available	10,00	A classic pizza with tomato sauce, mozzarella che...	5	preparing	2022-01-05 00:00:00.000
6	1	2	available	10,00	A classic pizza with tomato sauce, mozzarella che...	6	unavailable	2022-01-06 00:00:00.000
7	2	Spaghetti Bolognese	available	12,00	Spaghetti noodles with a rich meat sauce made wi...	1	preparing	2022-01-01 00:00:00.000
8	2	Spaghetti Bolognese	available	12,00	Spaghetti noodles with a rich meat sauce made wi...	2	unavailable	2022-01-02 00:00:00.000
9	2	Spaghetti Bolognese	available	12,00	Spaghetti noodles with a rich meat sauce made wi...	3	preparing	2022-01-03 00:00:00.000
10	2	Spaghetti Bolognese	available	12,00	Spaghetti noodles with a rich meat sauce made wi...	4	unavailable	2022-01-04 00:00:00.000

```
SELECT ingredient.ingredient_id, ingredient.ingredient_name,
ingredient.description, ingredient.amount, ingredient.price,
ingredient.detail, customer.first_name, customer.last_name
FROM ingredient INNER JOIN dbo.customer ON ingredient.ingredient_id =
customer.customer_id;
```

	ingredient_id	ingredient_name	description	amount	price	detail	first_name	last_name
1	1	Flour	All-purpose flour	5	3,99	A staple ingredient for baking and cooking	Nazrin	Huseynli
2	2	Sugar	Granulated white sugar	10	7,99	A sweetener commonly used in baking and cooking	Alex	Parker
3	4	Eggs	Large eggs	12	3,49	A versatile ingredient commonly used in baking and ...	Michael	Brown
4	5	Salt	Fine sea salt	1	2,99	A seasoning commonly used in cooking to enhance ...	Emily	Smith
5	6	Pepper	Black peppercorns	1	4,99	A common spice used to add flavor and heat to dishes	William	Johnson

5.
```
SELECT ingredient.ingredient_id, ingredient.ingredient_name,
ingredient.description, ingredient.amount, ingredient.price,
ingredient.detail, food_info.food_name, food_info.food_name
FROM ingredient FULL OUTER JOIN food_info ON ingredient.ingredient_id
= food_info.food_id;
```

	ingredient_id	ingredient_name	description	amount	price	detail	food_name	food_name
1	1	Flour	All-purpose flour	5	3,99	A staple ingredient for baking and cooking	2	2
2	2	Sugar	Granulated white sugar	10	7,99	A sweetener commonly used in baking and cooking	Spaghetti Bolognese	Spaghetti Bolognese
3	NULL	NULL	NULL	NULL	NULL	NULL	Lasagna	Lasagna
4	4	Eggs	Large eggs	12	3,49	A versatile ingredient commonly used in baking and ...	Fish & Chips	Fish & Chips
5	5	Salt	Fine sea salt	1	2,99	A seasoning commonly used in cooking to enhance ...	Roast Beef	Roast Beef
6	6	Pepper	Black peppercorns	1	4,99	A common spice used to add flavor and heat to dishes	Chicken Parmesan	Chicken Parmesan

6.
```
--7- ordering employees according to their birthdays
SELECT employee_id, e_firstname, e_lastname, e_birthdate FROM employee
ORDER BY e_birthdate desc
```

	employee_id	e_firstname	e_lastname	e_birthdate
1	1	Nazrin	Huseynli	2000-02-26 00:00:00.000
2	2	Alex	Parker	1995-03-17 00:00:00.000
3	3	Samantha	James	1990-04-08 00:00:00.000
4	4	Michael	Brown	1985-05-30 00:00:00.000
5	5	Emily	Smith	1980-06-20 00:00:00.000
6	6	William	Johnson	1975-07-11 00:00:00.000
7	7	Ashley	Williams	1970-08-01 00:00:00.000
8	8	David	Jones	1965-09-21 00:00:00.000
9	9	Sara	Davis	1960-10-11 00:00:00.000

7.
```
--8- grouping currently customers according to their country (how many
customers is from USA?)
SELECT customer.country, COUNT(*) AS 'Number of Currently USA
residents' FROM customer WHERE country = 'USA' GROUP BY country
```

Results Messages

	country	Number of Currently USA residents
1	USA	2

8.
```
--8- full join to see which types of food is available
SELECT * FROM food_info FULL OUTER JOIN restaurant_status ON
food info.food id = food info.food id;
```

	food_id	food_name	food_status	food_price	details	status_id	status_name	status_date
1	1	2	available	10,00	A classic pizza with tomato sauce, mozzarella che...	1	preparing	2022-01-01 00:00:00.000
2	1	2	available	10,00	A classic pizza with tomato sauce, mozzarella che...	2	unavailable	2022-01-02 00:00:00.000
3	1	2	available	10,00	A classic pizza with tomato sauce, mozzarella che...	3	preparing	2022-01-03 00:00:00.000
4	1	2	available	10,00	A classic pizza with tomato sauce, mozzarella che...	4	unavailable	2022-01-04 00:00:00.000
5	1	2	available	10,00	A classic pizza with tomato sauce, mozzarella che...	5	preparing	2022-01-05 00:00:00.000
6	1	2	available	10,00	A classic pizza with tomato sauce, mozzarella che...	6	unavailable	2022-01-06 00:00:00.000
7	2	Spaghetti Bolognese	available	12,00	Spaghetti noodles with a rich meat sauce made wi...	1	preparing	2022-01-01 00:00:00.000
8	2	Spaghetti Bolognese	available	12,00	Spaghetti noodles with a rich meat sauce made wi...	2	unavailable	2022-01-02 00:00:00.000
9	2	Spaghetti Bolognese	available	12,00	Spaghetti noodles with a rich meat sauce made wi...	3	preparing	2022-01-03 00:00:00.000
10	2	Spaghetti Bolognese	available	12,00	Spaghetti noodles with a rich meat sauce made wi...	4	unavailable	2022-01-04 00:00:00.000
11	2	Spaghetti Bolognese	available	12,00	Spaghetti noodles with a rich meat sauce made wi...	5	preparing	2022-01-05 00:00:00.000

9.
```
--9-- right join to see the list of employee departments with the crew names together
SELECT * FROM employee RIGHT JOIN crew ON crew.crew_name = employee.e_department
```

	e_age	e_salary	e_department	e_birthdate	e_birthcity	e_birthcountry	e_phone_number	e_mail	crew_id	crew_name
1	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	2	Night shift
2	30	2500,00	Bakery	1990-04-08 00:00:00.000	Toronto	Canada	345-678-9012	sjames1@gmail.com	3	Bakery
3	60	3500,00	Bakery	1960-10-11 00:00:00.000	Sydney	Australia	901-234-5678	sdavis1@gmail.com	3	Bakery
4	35	2000,00	Bar	1985-05-30 00:00:00.000	London	UK	456-789-0123	mbrown1@gmail.com	4	Bar
5	40	3500,00	Management	1980-06-20 00:00:00.000	Sydney	Australia	567-890-1234	esmith1@gmail.com	5	Managem...
6	45	3000,00	Accounting	1975-07-11 00:00:00.000	Chicago	USA	678-901-2345	wjohnson1@gmail....	6	Accounting

10.
```
--10-- join to see customer's email  (id and score come from different tables)
SELECT customer.mail_adress, first_name FROM customer JOIN restaurant_status ON customer.mail_adress = customer.first_name
```

	mail_adress	first_name

. 5 VİEW commands for tables (Delete_Statements.sql)