Table of Contents

[Conceptual Design in Entity Relationship Diagram 1](#_Toc193913436)

[Entities 2](#_Toc193913437)

[Relationships 2](#_Toc193913438)

[Relational Schema 3](#_Toc193913439)

[Relationship and Constraint 3](#_Toc193913440)

[Entity Relationship Diagram 4](#_Toc193913441)

[Implementation using Microsoft Access 6](#_Toc193913442)

[What is Microsoft Access 6](#_Toc193913443)

[Why Microsoft Access 6](#_Toc193913444)

[Installation 6](#_Toc193913445)

[Tables Creation for Entities 9](#_Toc193913446)

[Database Normal Forms (Normalization) 10](#_Toc193913447)

[Populating Tables with Data 14](#_Toc193913448)

[Structured Query Language 15](#_Toc193913449)

[1. DDL (Data Definition Language) 15](#_Toc193913450)

[2. DML (Data Manipulation Language) 15](#_Toc193913451)

[3. DCL (Data Control Language) 16](#_Toc193913452)

[4. TCL (Transaction Control Language) 16](#_Toc193913453)

[ACID Properties in SQL 16](#_Toc193913454)

[Transaction Locking 16](#_Toc193913455)

[Queries 17](#_Toc193913456)

[Database Application 21](#_Toc193913457)

[Types of Objects for Creating Application 21](#_Toc193913458)

[1. Forms 21](#_Toc193913459)

[2. Datasheet 21](#_Toc193913460)

[3. Reports 21](#_Toc193913461)

[Steps to Create 21](#_Toc193913462)

[Appointment Scheduling Database Application 28](#_Toc193913463)

[Consideration on Privacy and Security 34](#_Toc193913464)

# Conceptual Design in Entity Relationship Diagram

Conceptual design is the first stage of database design, where we identify key entities, their attributes, and relationships. It helps in understanding how data is structured before actual implementation in a database.

For **Appointment Scheduling for a Service-Based Small & Medium Enterprise (SME),** the conceptual design is based on functional analysis requirements. This means we first study how the company operates (functional requirements) and then translate that into a database structure.

* In the database, there are total Seven(7) tables which combines four(4) Entities and three(3) Relationship

**First, we will describe the entities and their attributes**.

**Client Entity in SME System**

The Client entity represents customers or businesses that interact with the SME system. This entity stores important details about each client, including their ClientID, ClientName, contact details, and address.  
**Key Functions of the Client Entity:**

1. Identifies each client uniquely.
2. Stores essential contact details for communication.
3. Helps in managing client orders, invoices, and payments.
4. Ensures no duplicate clients exist in the system.

**Attributes of the Client Entity**

| **Attribute** | **Description** | **Data Type** | **Constraints** |
| --- | --- | --- | --- |
| **ClientID** | Unique identifier for each client | INT | **Primary Key (PK), Auto-increment** |
| **ClientName** | Name of the client | VARCHAR(255) | **NOT NULL** |
| **ClientAddress** | Address of the client | TEXT | **NULL allowed** |
| **ClientEmail** | Email address of the client | VARCHAR(255) | **Unique, NOT NULL** |
| **ClientMobile** | Mobile number of the client | VARCHAR(15) | **Unique, NOT NULL** |

**Why is the Client Entity Important?**

1. It ensures that each client is uniquely identifiable in the system.
2. Helps in managing client interactions, orders, and services efficiently.
3. Ensures proper communication through email and phone details

**Employee Entity in SME System**

The **Employee** entity is a crucial part of the database design for the **Rainbow International SME UK** system. It represents the employees working in the organization, storing essential details about them. Employees play a key role in business operations, such as handling customer orders, providing services, managing payroll, and ensuring smooth business operations.

The Employee entity represents individuals working for the SME system. It stores key details about each employee, such as their unique identifier, payroll number, name, and mobile contact. **Attributes of the Employee Entity:**

| **Attribute** | **Description** | **Data Type** | **Constraints** |
| --- | --- | --- | --- |
| **EmployeeID** | Unique identifier for each employee | INT | **Primary Key (PK), Auto-increment** |
| **EmployeePayrollNumber** | Unique payroll number for salary processing | VARCHAR(50) | **Unique, NOT NULL** |
| **EmployeeName** | Full name of the employee | VARCHAR(255) | **NOT NULL** |
| **EmployeeMobile** | Employee's mobile phone number | VARCHAR(15) | **Unique, NOT NULL** |

**Why is the Employee Entity Important?**

1. Helps in managing employees efficiently.
2. Ensures accurate payroll and tax processing.
3. Prevents duplicate records through unique constraints.
4. Supports communication by storing contact details.

**Service Catalogue Entity in SME System**

The Service Catalogue entity represents the different services that the SME offers to its clients. This entity helps in organizing and managing the services provided, including their descriptions and pricing.

**Purpose of the Service Catalogue Entity:**

1. Stores and categorizes all services offered by the business.
2. Provides a structured way to manage service details.
3. Helps in defining service pricing and descriptions.
4. Links with orders or invoices for tracking service usage.

**Attributes of the Service Catalogue Entity:**

| **Attribute** | **Description** | **Data Type** | **Constraints** |
| --- | --- | --- | --- |
| **ServiceID** | Unique identifier for each service | INT | **Primary Key (PK), Auto-increment** |
| **ServiceName** | Name of the service | VARCHAR(255) | **NOT NULL** |
| **ServiceDescription** | Detailed description of the service | TEXT | **NULL allowed** |
| **ServiceHourlyRate** | Hourly cost for the service | DECIMAL(10,2) | **NOT NULL, Must be positive** |

**Why is the Service Catalogue Entity Important?**

1. Helps businesses manage services efficiently.
2. Ensures clear pricing and descriptions for customers.
3. Allows integration with invoices and orders for proper tracking.

**Invoice Entity in SME System**

The Invoice entity represents the billing details for services provided to clients. It tracks financial transactions, ensuring that payments are properly recorded for accounting and business operations.

**Purpose of the Invoice Entity:**

1. Keeps records of all financial transactions between the business and its clients.
2. Helps in tracking payments, discounts, and total amounts due.
3. Ensures that every service or product has a corresponding bill.
4. Links to orders and payments for proper reconciliation.  
     
   **Attributes of the Invoice Entity:**

| **Attribute** | **Description** | **Data Type** | **Constraints** |
| --- | --- | --- | --- |
| **InvoiceID** | Unique identifier for each invoice | INT | **Primary Key (PK), Auto-increment** |
| **InvoiceNumber** | Unique reference number for the invoice | VARCHAR(50) | **Unique, NOT NULL** |
| **InvoiceCost** | Total cost before discounts are applied | INT | **NOT NULL, Must be positive** |
| **InvoiceDiscount** | Discount applied on the invoice | INT | **Default 0, Must be positive** |
| **InvoiceTotal** | Final amount payable after applying discounts | INT | **NOT NULL, Must be positive** |
| **InvoiceDate** | Date when the invoice was generated | DATE | **NOT NULL** |

**Why is the Invoice Entity Important?**

1. Ensures proper billing and financial tracking.
2. Helps in calculating revenue and applying discounts.
3. Provides a structured way to track payments.

## Relational Schema

1. **Clients** { ClientID, ClientName, ClientAddress, ClientEmail, ClientMobile }
2. **Employees** { EmployeeID, EmployeePayrollNumber, EmployeeName, EmployeeMobile }
3. **ServiceCatalogue** { ServiceID, ServiceName, ServiceDescription, ServiceHourlyRate }
4. **Invoices** { InvoiceID, InvoiceNumber, InvoiceCost, InvoiceDiscount, InvoiceTotal, InvoiceDate }
5. **ClientAppointments** { ClientAppointmentID, AppointmentDate, *ServiceID*, *ClientID*, *EmployeeID* }
6. **ServicesProvided** { *ServiceID*, *EmployeeID*, *ClientAppointmentID*, Expenses }
7. **InvoiceRows** { InvoiceRowID, *InvoiceID*, *ClientAppointmentID* }

Represents Primary key

Represents Foreign key

## Relationship and Constraint

1. **One to One (1:1) Relationship:** As there is no one-to-one (1:1) relation between the entities, all have either 1:M or M:N relation in the diagram.
2. **One to many (1:M) Relationship:** This happens when one record in one entity is related to multiple records in another.
   1. **Clients → ClientAppoinments:** One client can have many appointments(One ClientAppointments)
   2. **Employees** **→** **ClientAppoinments:** A single Employee can have multiple appointments on (ClientAppointments).
   3. **ServiceCatalogue** **→** **ClientAppoinments:** One or more appointments can be booked for a Service from the Service Catalogue.
   4. **Invoices** **→** **InvoiceRows:** This means that an invoice can have zero or many invoice rows (InvoiceRows).
   5. **ClientAppoinments** **→** **InvoiceRows:** One invoice row can be attached to multiple client appointments
3. **Many to One (M:1) Relationship:** In a way these are kind of the opposite of one-to-many (1:M) relationships.
   1. Multiple appointments are associated with one client
   2. Single employee has many appointments.
   3. One service can have many appointments.
   4. One invoice has many invoice rows.
   5. Multiple invoice rows point to one client appointments.
4. **Many to Many (M:N) Relationship:** These relationships are constructed using junction tables.
   1. **Employees ⇄ ServicesProvided ⇄ Service Catalogue:** A single employee is able to provide multiple services, and a single service is being provided by multiple employees. (ServicesProvided table)
   2. **Employees ⇄ ServicesProvided ⇄ ClientAppointments:** One employee can conduct many clients appointments, and one appointment can be for many employees. (ServicesProvided table)
   3. **Service Catalogue ⇄ ServicesProvided ⇄ ClientAppointments:** And a service maybe for multiple appointments and an appointment maybe for multiple services. (ServicesProvided table)
   4. **Invoices ⇄ InvoiceRows ⇄ ClientAppointments:** One or more invoices can be associated with one or more appointments; and one or more appointments can be billed in multiple invoices. (InvoiceRows table)

**Relationship Summary**

|  |  |
| --- | --- |
| **Type of Relationships** | **Example** |
| **1:1 (One-to-One)** | No, 1:1 Relationships |
| **1:M (One-to-Many)** | Clients → ClientAppoinments, Employees → ClientAppoinments, ServiceCatalogue → ClientAppoinments, Invoices → InnvoiceRow, ClientAppoinments → InvoiceRows |
| **M:1 (Many-to-One)** | Opposite of One-to-Many |
| **M:N (Many-to-Many)** | Employees ⇄ ServicesProvided ⇄ ServiceCatalogue, Employees ⇄ ServicesProvided ⇄ ClientAppointments, ServiceCatalogue ⇄ ServicesProvided ⇄ ClientAppointments, Invoices ⇄ InvoiceRows ⇄ ClientAppointments |

**Integrity Constraints**

An entity integrity constraint is a rule that ensures each record in a database table is unique, usually enforced by primary keys and UNIQUE constraints.

1. Foreign Key Constraints: Enforce referential integrity between related tables.

## Entity Relationship Diagram

An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system. They are used to design relational database management systems.

**Partial Entity Relationship Diagram**

A group of white rectangles with black text

AI-generated content may be incorrect.

**Full Diagram with all Attributes**

A black background with white text

AI-generated content may be incorrect.

# Implementation using Microsoft Access

In this Appointment Scheduling for a Service-Based Small & Medium Enterprise (SME) Application, we are using Microsoft Access to build the database, tables and a Graphical User Interface.

## What is Microsoft Access

Microsoft Access is a database management system (DBMS) from Microsoft that combines the relational Access Database Engine with a graphical user interface and software-development tools, allowing users to create and manage databases particularly for small to medium-sized businesses and individual users, offering a user-friendly interface and integration with other Microsoft Office applications.

### Why Microsoft Access

There are several advantages using it.

1. **Ease of Use:** Access is known for its easy-to-understand interface, making it relatively easy for non-technical users to create and manage databases, including tables, forms, queries, and reports.
2. **Cost-Effectiveness:** Compared to enterprise-level database systems like SQL Server or Oracle, Access is a more affordable option, especially for small businesses and individual users.
3. **Templates and Wizards:** Access offers pre-built templates and wizards that can be customized to fit specific needs, allowing users to quickly create databases and applications.

### Installation

Microsoft Access comes with Microsoft 365 Suite. So to install Access we need to install MS 365 suite. Goto [Microsoft 365 official website](https://www.microsoft.com/en-us/microsoft-365/download-office#download) and download the installation file for Windows/MacOS.

A black rectangular object with a line

AI-generated content may be incorrect.

Once it is downloaded, we need to open the file to start the installation process. The installation process needs administrator privileges, so make sure you have the admin permissions.

A screenshot of a computer

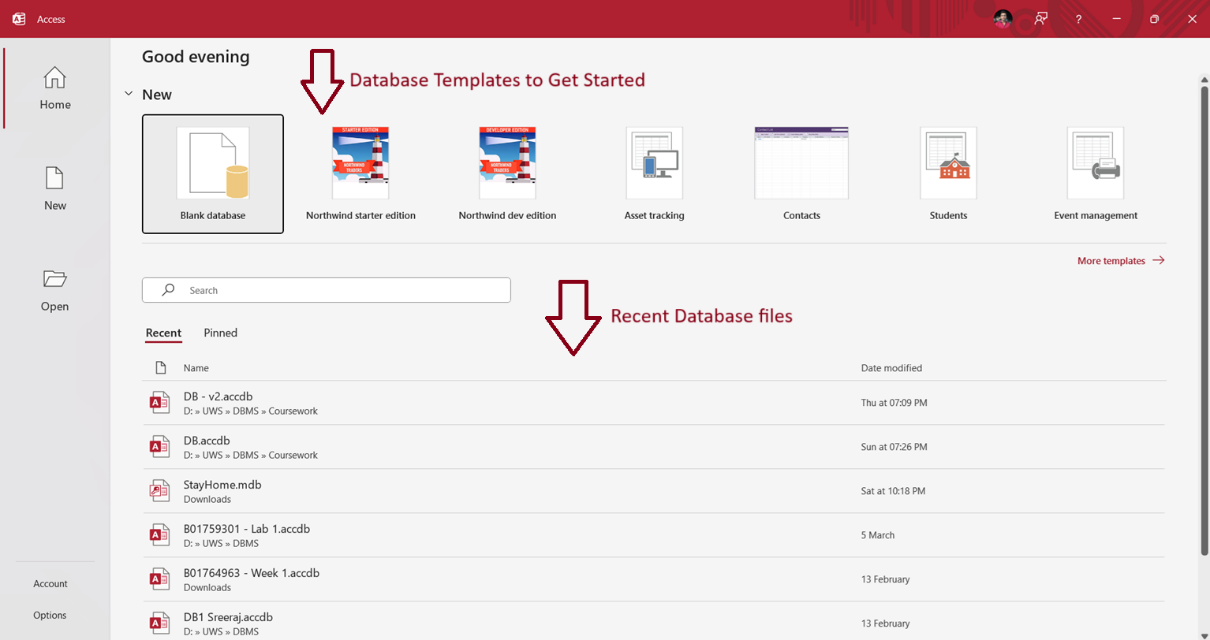
AI-generated content may be incorrect.

Once the installation file opened, please wait for couple of minutes until the installation finishes. After completion of installation, we are ready to use Access.

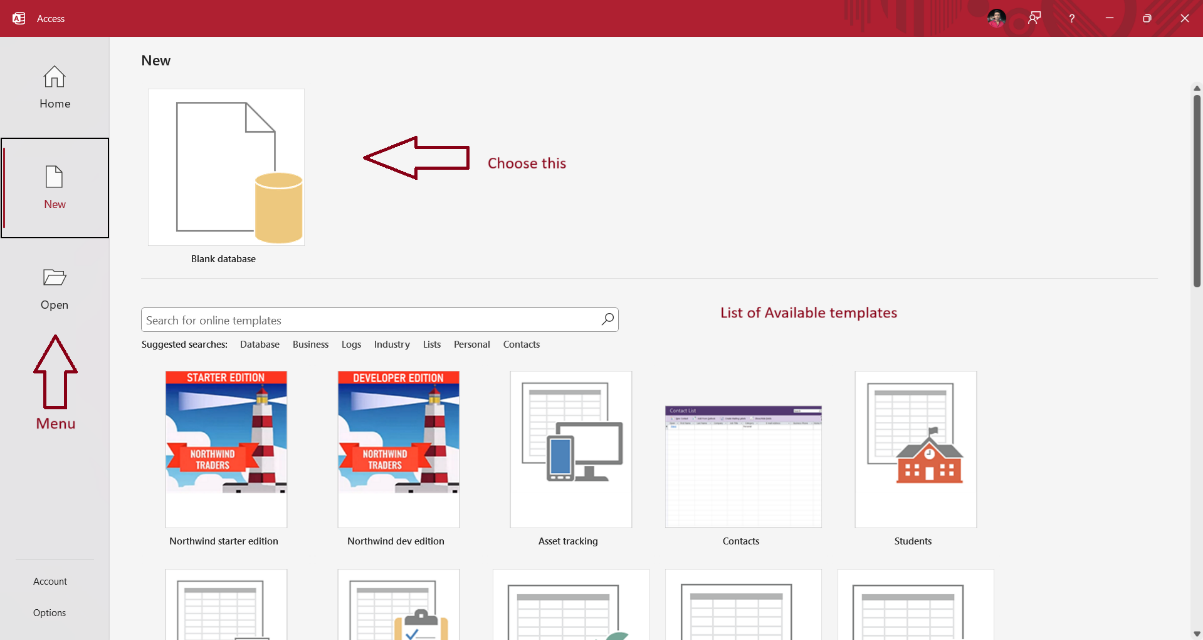
A screenshot of a computer

AI-generated content may be incorrect.

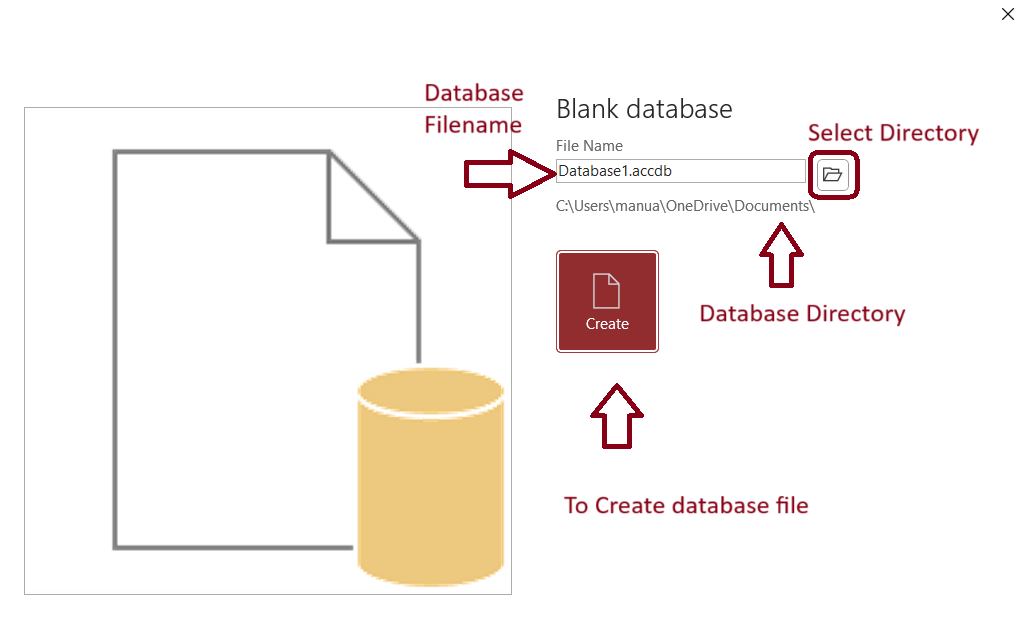
This is the icon for Microsoft Access, after opening you are presented with this user interface.



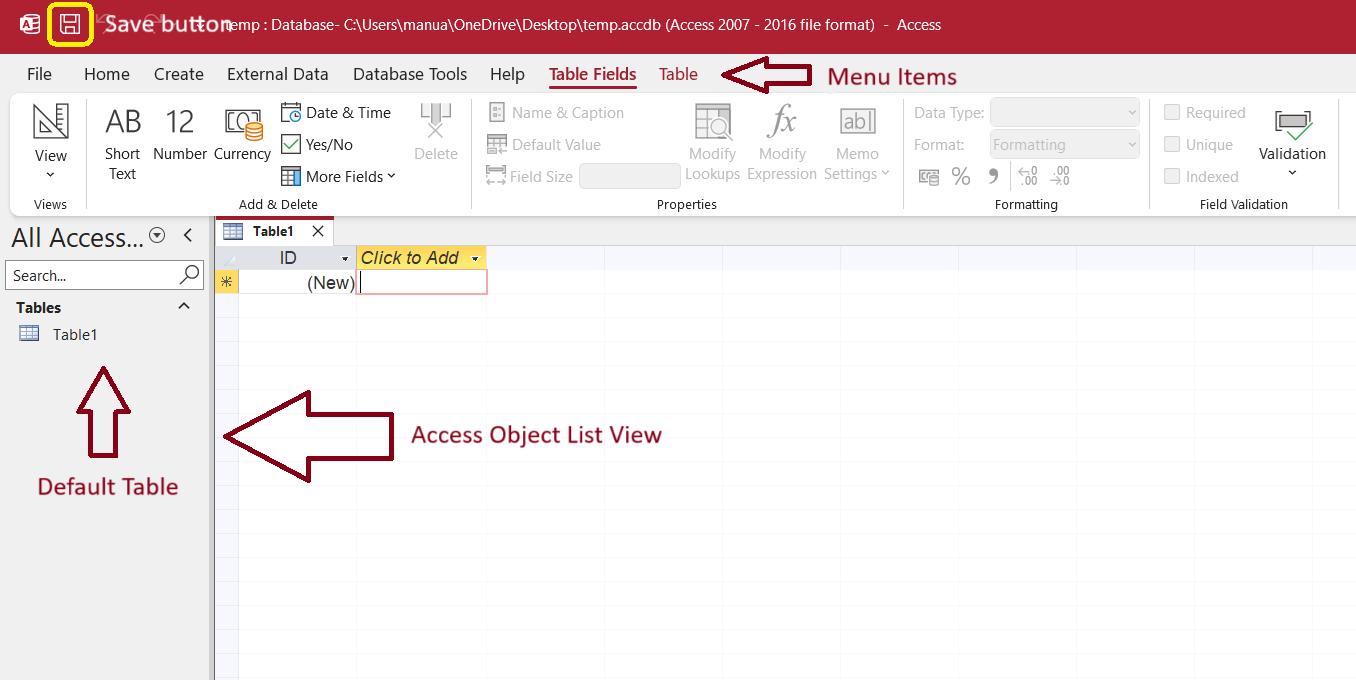
In this window, we can see there are option to select Blank Database, or we can use the pre-made templates to quickly get started with. The 2nd half of the window shows recent files that are previously opened. For fresh installations, you won’t see anything, but in my pc I already opened databases previously so I get a list of databases to open.



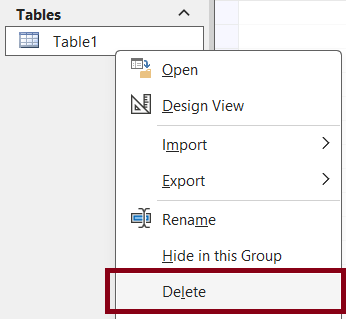
In the menu bar on left, click the new menu item to open a new window that shows a list of templates to choose from or we start our own from scratch using Blank database option. So, we are using Blank database.



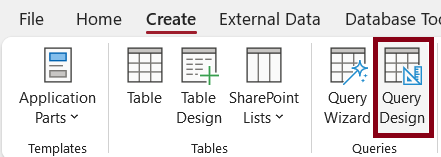
A popup window opens that asks few information, Name of the database file, where to store the file in disk. Once everything is done, we can click Create button to create blank database.



After creating the database, the file will be opened automatically. And a new window will be presented with a lot of menus and options. On the top of the window, we have Save button to save any changes made to the database. Below that, we have menu tab items. And on left side of the window, we have Access Object List Viewer which lists all the objects related to Access such as Tables, Forms, Reports, Queries etc. The main window that is focused on is Data sheet viewer of the table. Once a new database is created, automatically a new Table called ***Table1*** is created. We need to delete by right clicking on the Tables view on left side.



We need to create a table, Goto Create menu and click Query Design to open query editor window.

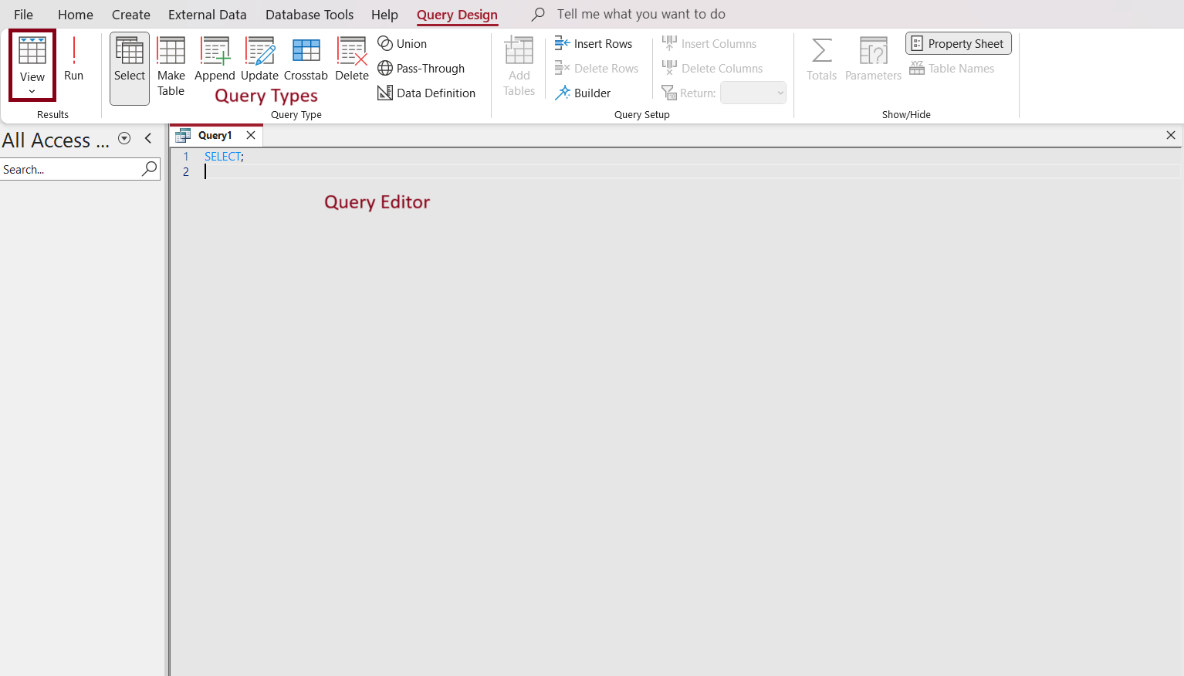


After opening query window, we need to select the SQL view using the dropdown menu on top left corner.

A screenshot of a computer

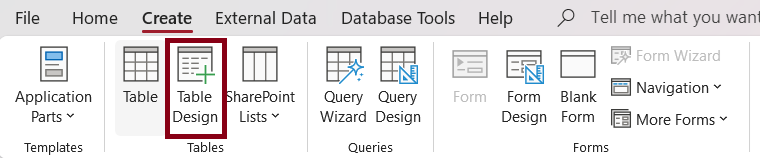
AI-generated content may be incorrect.

After selecting, the window should look something like this. Here we can write actual SQL queries in the Query Editor and run them against the database.

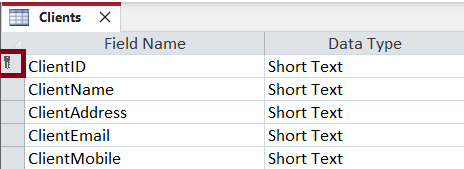


## Tables Creation for Entities

For creating tables for our entities, we use Table Design feature as it is easy and convenient. All the tables in the database are **3NF compliant**.



Using Table Design feature, we need to list out all the attributes for that entity and its data type.



After entering all the details, to create the table we need to save it, after that it will display a popup message to enter the table name. In ClientID, there is a key icon which means primary key.

A screenshot of a computer

AI-generated content may be incorrect.

### Database Normal Forms (Normalization)

Database normalization is a database design principle for organizing data in an organized and consistent way. It helps you avoid redundancy and maintain the integrity of the database. It also helps you eliminate undesirable characteristics associated with insertion, deletion, and updating.

**What is the purpose of Normalization?**

The main purpose of database normalization is to avoid complexities, eliminate duplicates, and organize data in a consistent way. In normalization, the data is divided into several tables linked together with relationships.

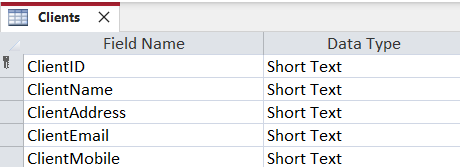
In practice, there are **three** Normal Forms currently in use, but total there are **six** Normal Forms.

1. **1st Normal Form:** For a table to be in the first normal form, it must meet the following conditions.
   1. A single cell must not hold more than one value.
   2. There must be a primary key for identification.
   3. No duplicated rows or columns.
2. **2nd Normal Form:** A table is said to be in 2NF if it meets the following conditions.
   1. It should be 1st Normal Form compliant.
   2. It has no partial dependency. That is, all non-key attributes are fully dependent on a primary key.
3. **3rd Normal Form:** So, for a table to be in 3NF, it must:
   1. Be in 2nd Normal Form.
   2. It should not have transitive partial dependency.

**Clients Table**

The Clients table stores information about the company’s clients.

Clients table has 5 attributes including ClientID, ClientName, ClientAddress, ClientEmail, ClientMobile and ClientID is the primary key. The below screenshot shows the table structure with data type.



Every attribute has a validation rule and must be provided some valid data. Input mask has been set for ClientID as **"CL-"099999** and ClientMobile as **"0"000\ 000\ 0000** is set, so user can easily enter data without confusions.

**Employees Table**

The Employees table stores all the data related to employee who works for the company.

Employees table has 4 attributes including EmployeeID, EmployeePayrollNumber, EmployeeName, EmployeeMobile and EmployeeID is the primary key. The below screenshot shows the table structure with data type.

A screenshot of a computer

AI-generated content may be incorrect.

Every attribute has a validation rule and must be provided some valid data. Input mask has been set for EmployeeID as **"EM-"099999** and EmployeeMobile as **"0"000\ 000\ 0000** is set, so user can easily enter data without confusions.

**Service Catalogue Table**

The Service Catalogue table stores all the services provided by the company.

Service Catalogue table has 4 attributes including ServiceID, ServiceName, ServiceDescription, ServiceHourlyRate and ServiceID is the primary key. The below screenshot shows the table structure with data type.

A screenshot of a computer

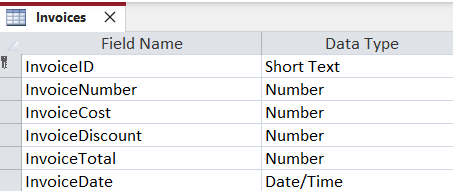
AI-generated content may be incorrect.

Every attribute has a validation rule and must be provided some valid data. Input mask has been set for ServiceID as **"SC-"099999**.

**Invoices Table**

The Invoices table stores all the billing related information for every client appointment.

Invoices table has 6 attributes including InvoiceID, InvoiceNumber, InvoiceCost, InvoiceDiscount, InvoiceTotal, InvoiceDate and InvoiceID is the primary key. The below screenshot shows the table structure with data type.



Every attribute has a validation rule and must be provided some valid data. Input mask has been set for InvoiceID as **"IN-"099999**.

**ClientAppointments Table**

The Client Appointments table stores all the information about appointment with client and employee for a particular service.

Client Appointments table has 5 attributes including ClientAppointmentID, ServiceID, ClientID, EmployeeID, AppointmentDate and ClientAppointmentID is the primary key. And Foreign keys are, ServiceID which is linked to ServiceCatalogue table, EmployeeID is linked to Employees table, and finally ClientID linked to Clients table. The below screenshot shows the table structure with data type.

A screenshot of a computer

AI-generated content may be incorrect.

Every attribute has a validation rule and must be provided some valid data. Input mask has been set for ClientAppointmentID as **"CA-"099999**.

**ServicesProvided Table**

The Services Provided table stores information about which employee provided what service to which client.

Services Provided table has 4 attributes including ServiceID, EmployeeID, ClientAppointmentID, Expenses. The below screenshot shows the table structure with data type.

A screenshot of a computer

AI-generated content may be incorrect.

**InvoiceRows Table**

The Invoice Rows links the Invoice table to the ClientAppointments table.

Invoice Rows table has 3 attributes including InvoiceRowID, InvoiceID, ClientAppointmentID, and InvoiceID is the primary key. The below screenshot shows the table structure with data type. The Foreign keys are InvoiceID which is linked to Invoices table and ClientAppointmentID is linked to ClientAppointments table.

A screenshot of a computer

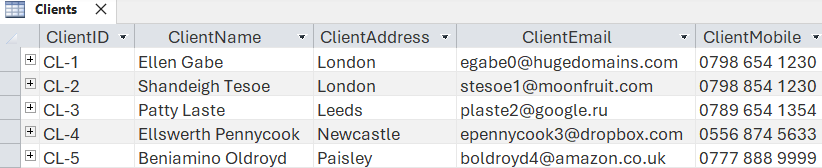
AI-generated content may be incorrect.

Every attribute has a validation rule and must be provided some valid data. Input mask has been set for InvoiceRowID as **"IR-"099999**.

# Populating Tables with Data

After creating tables through Microsoft access, we populate the tables with data either manually entering data or we can import from excel files, CSV files or other databases. The process fills the databases with records that can be store, extract or change as we need. To enter any datasets in databases, if the datasets are smaller, we can enter data manually which can be simple and quick for small datasets. If the datasets are large, it can be efficient to import from CSV or excel files, while using SQL queries we enter multiple records quickly and for non-technical users data can be entered by using forms. We have used simpler datasets so we enter data in each table manually.

1. **Clients**: - To store information about the company’s clients we create a table naming Clients with 5 attributes where ClientID is the primary key.



**Above Screenshot contain: -**

* **ClientID:** This is the unique ID for each client (CL-1, CL-2, etc.).
* **ClientName:** The name of the client (Ellen Gabe, Shandeigh Tesoe etc.).
* **ClientAddress:** The city where the client is (London, Leeds, Newcastle, etc.)
* **ClientEmail:**The client's email address (ex. egabe0@hugedomains.com)
* **ClientCellphone:** The cellphone number of the client (e.g., 0798 654 1230).

1. **Employees:** - Employees table store the data of employees who works for the company which has 4 attributes where EmployeeID is primary key.

A screenshot of a computer

AI-generated content may be incorrect.

**Above Screenshot contains: -**

* **EmployeeID:** A unique number assigned to each employee (say, EM-1, EM-2, etc.)
* **EmployeePayrollNumber:** Numeric payroll value for each employee
* **EmployeeName:** Employee name (e.g., Margaret Keyes, Mischa Sandiland)
* **EmployeeMobile:** This attributes store the value of mobile number of employees.

1. **ServiceCatalogue**: - The table ServiceCatalogue store the services which provided by company, there is 4 attributes. ServiceID is primary key in this table.

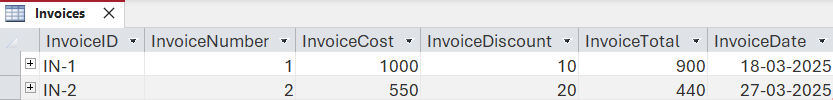
A screenshot of a computer

AI-generated content may be incorrect.

**Above Screenshot Contain**

* **ServiceID:** An identifier for each service (e.g., SC-1, SC-2, etc)
* **ServiceName**: This must be the name that you provide for the services that you provide (e.g. ExteriorPainting, Cleaning, Joinery).
* **ServiceDescription:** Short description of service (Exterior wall Painting, Moving objects from one place to another).
* **ServiceHourlyRate:** The hourly service rate (e.g., 20,25, 40).

1. **Invoices:** -The Invoices table stores all the billing related information for every client appointment. Invoices table has 6 attributes and InvoiceID is the primary key.



**Above Screenshot contain**

* **InvoiceID:** A unique ID for each invoice (eg: IN-1, IN-2).
* **InvoiceNumber:** A number that increments with every invoice.
* **InvoiceCost:** This is the raw cost without a discount.
* **InvoiceDiscount:** The amount of discount that is applied to the invoice.
* **InvoiceTotal :** The total amount at the end (InvoiceCost - InvoiceDiscount).
* **InvoiceDate:** Date when the invoice was created (Example: 18-03-2025, 27-03-2025)

1. **ClientAppointments:** - The Client Appointments table stores all the information about appointment with client and employee for a particular service. Client Appointments table has 5 attributes and ClientAppointmentID is the primary key. And Foreign keys are, ServiceID which is linked to ServiceCatalogue table, EmployeeID is linked to Employees table, and finally ClientID linked to Clients table.

A screenshot of a computer

AI-generated content may be incorrect.

**Above Screenshot contains**

* **ClientAppointmentID:** This is mainly the ID for each of the client appointments (CA-1, CA-2 ….)
* **ServiceID:** Identifier of the service being performed (e.g., SC-1, SC-3, etc.).
* **ClientID:** Which client are we providing our service (CL-5, CL-3, etc.).
* **EmployeeID:** The ID of the employee who is assigned to the appointment (like EM-3, EM-4, etc.
* **AppointmentDate:** The date on which appointment is scheduled (e.g. 06-03-2025, 12-03-2025)

1. **ServiceProvided**: - The Services Provided table stores information about which employee provided what service to which client. Services Provided table has 4 attributes including ServiceID, EmployeeID, ClientAppointmentID, Expenses.

A screenshot of a computer

AI-generated content may be incorrect.

**Above Screenshot Contain**

* **ServiceID:** Identifier of the service being performed (e.g., SC-1, SC-3, etc.).
* **EmployeeID:** The ID of the employee who is assigned to the appointment (like EM-3, EM-4, etc.
* **ClientAppointmentID:** This is mainly the ID for each of the client appointments (CA-1, CA-2 etc.)
* **Expenses:** Cost of using the service (in money-terminal e.g 350,350,125)

1. **InvoiceRows**: - The Invoice Rows links the Invoice table to the ClientAppointments table. Invoice Rows table has 3 attributes and InvoiceID is the primary key. The Foreign keys are InvoiceID which is linked to Invoices table and ClientAppointmentID is linked to ClientAppointments table.

A screenshot of a computer

AI-generated content may be incorrect.

**Above Screenshot Contain**

* **InvoiceID:** A unique ID for each invoice (eg: IN-1, IN-2).
* **InvoiceRowID:** A unique ID for each invoice row (IR-1. IR2 etc.)
* **ClientAppointmentID:** This is mainly the ID for each of the client appointments (CA-1, CA-2 etc.)

# Structured Query Language

Structured Query Language (SQL) is a common language for managing and manipulating relational database management systems. It is particularly useful in handling structured data, i.e., data incorporating relations among entities and variables.

In SQL, there are a total of 4 different types of SQL Commands.

1. DDL (Data Definition Language): It consists of SQL commands that can be used to defining, altering, and deleting database structures such as tables and indexes.

Commands used:

|  |  |
| --- | --- |
| **Command** | **Description** |
| CREATE | Used for creating database, table, indexes and views. |
| DROP | Used for deleting database objects i.e. tables. |
| ALTER | Used for modifying the structure of the table. |
| TRUNCATE | Deletes all the records from a specified table. |

1. DML (Data Manipulation Language): The SQL commands that deals with manipulation of the data in a table.

Commands used:

|  |  |
| --- | --- |
| **Command** | **Description** |
| INSERT | Insert data into database. |
| UPDATE | Updates an existing row(s) in database, optionally we can pass condition to update row otherwise it will update everything. |
| DELETE | Delete record from a table with a condition. |

1. DCL (Data Control Language)**:** DCL (Data Control Language) includes commands such as **GRANT** and **REVOKE** which mainly deal with the rights, permissions, and other controls of the database system.

|  |  |
| --- | --- |
| **Command** | **Description** |
| GRANT | Assigns new privileges to a user account, allowing access to specific database objects, actions, or functions. |
| REVOKE | Removes previously granted privileges from a user account, taking away their access to certain database objects or actions. |

1. TCL (Transaction Control Language)**:** Transactions group a set of queries into single execution unit. Each transaction begins with a specific task and ends when all the tasks in the group are successfully completed. If any of the tasks fail, the transaction fails. Therefore, transaction have only 2 states: **success** or **failure**. All transactions in SQL are ACID compliant.

|  |  |
| --- | --- |
| **Command** | **Description** |
| BEGIN TRANSACTION | Starts a new transaction. |
| COMMIT | Saves all changes made during the transaction. |
| ROLLBACK | Reverts all changes made during the transaction. |

Example Query for Transaction:

BEGIN TRANSACTION;

UPDATE Clients

SET ClientName = 'Rogers'

WHERE ClientID = 2;

COMMIT;

### ACID Properties in SQL

ACID means Atomicity, Consistency, Isolation, and Durability.

1. **Atomicity:** Each SQL statement in a transaction is treated as single unit. Either entire statement is executed or none.
2. **Consistency:** Ensures that transactions only make changes to table in predefined, predictable way. It ensures corruption of data and preserves integrity.
3. **Isolation:** Even though multiple transactions are running concurrently, Isolation makes sure that they do not interfere with each other.
4. **Durability:** Ensures all the changes made in database will be saved even on system failures.

### Transaction Locking

Transaction locking in SQL ensures data consistency and prevents conflicts by restricting access to rows or tables during transactions.

**Types of Transaction Locking**

There are 2 types of Locking mechanisms.

1. **Optimistic Locking:** Allows multiple transactions to read data and checks for conflicts before committing changes. It uses a version number or timestamp to detect conflicts before committing changes.
2. **Pessimistic Locking:** Prevents conflicts by locking data until a transaction is complete, blocking other operations.

## Queries

1. **Appointment List:** Lists out all the appointments with proper names instead of IDs. This uses inner join to link other tables such as Clients, Client Appointments and Service Catalogue.

SELECT clientappointments.clientappointmentid,

       servicecatalogue.servicename,

       clients.clientname,

       employees.employeename

FROM employees

    INNER JOIN(clients

    INNER JOIN(servicecatalogue

    INNER JOIN clientappointments

        ON servicecatalogue.serviceid = clientappointments.[serviceid])

        ON clients.clientid = clientappointments.clientid)

        ON employees.employeeid = clientappointments.employeeid;

Result:

A screenshot of a computer

AI-generated content may be incorrect.

1. **Clients List:** Lists out all the client data from the Clients table.

SELECT Clients.ClientID,

       Clients.ClientName,

       Clients.ClientAddress,

       Clients.ClientEmail,

       Clients.ClientMobile

FROM Clients;

Result:

A screenshot of a computer

AI-generated content may be incorrect.

1. **Employee List:** Lists out all employee data.

SELECT Employees.EmployeeID,

       Employees.EmployeePayrollNumber,

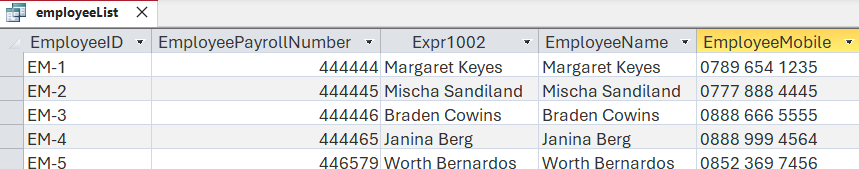
       Employees.EmployeeName,

       Employees.EmployeeName,

       Employees.EmployeeMobile

FROM Employees;

Result:



1. **Get Client By ID:** Gets a particular client by their ID. It uses parameters to get Client ID from user. A modal popup will be shown to user to enter the ID.

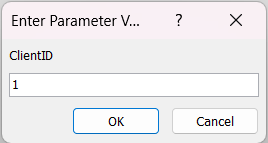
PARAMETERS ClientID Text ( 255 );

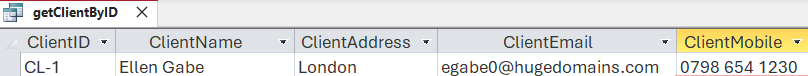
SELECT clientList.\*

FROM clientList

WHERE (((clientList.ClientID) = [ClientID]));

Result:





1. **Insert a new Client:** Creates a new record in the Clients table with the given data using insert query.

INSERT INTO Clients

(ClientID, ClientName, ClientAddress, ClientEmail, ClientMobile)

VALUES

("6", "Prajwal Aradhya", "London", "prajwal@email.com", "7757184256");

Result:



1. **Insert a new Employee:** Creates a new Employee record in Employees table with given data using insert query.

INSERT INTO Employees

(EmployeeID, EmployeeName, EmployeePayrollNumber, EmployeeMobile)

VALUES

("6", "Sayyar", 789456, "7757185698");

Result:



1. **Updating Service Catalogue:** Updates a Service Catalogue’s ServiceHourlyRate field to **55** where ServiceID equals **1**.

UPDATE ServiceCatalogue

SET ServiceHourlyRate = 55

WHERE ServiceID = "1";

Result before running query:



Result after running query:



1. **Deleting Service Catalogue:** Deletes a row from Service Catalogue table from it’s ID.

DELETE FROM ServiceCatalogue

WHERE ServiceID = "6";

Result before running query:

A yellow line on a white background

AI-generated content may be incorrect.

Result after running query:

A screenshot of a computer

AI-generated content may be incorrect.

1. **Update Client Appointment:** The query updates the appointment date for a specific client appointment in the `ClientAppointments` table based on its ID.

UPDATE ClientAppointments

SET AppointmentDate = "20-3-2025"

WHERE ClientAppointmentID = "2";

Result before running query:



Result after running query:



1. **Update Services Provided:** The query updates the `Expenses` value for a specific service entry in the `ServicesProvided` table based on its ID.

UPDATE ServicesProvided

SET Expenses = 700

WHERE ServiceProvidedID = 5;

Result before running query:



Result after running query:



# Database Application

Creating database applications in MS Access involves designing tables to store structured data, defining relationships, and using queries to retrieve and manipulate data. Forms provide user-friendly interfaces for data entry, while reports generate structured outputs.

## Types of Objects for Creating Application

1. Forms**:** They provide a user-friendly interface for entering, editing, and viewing data in a structured way.

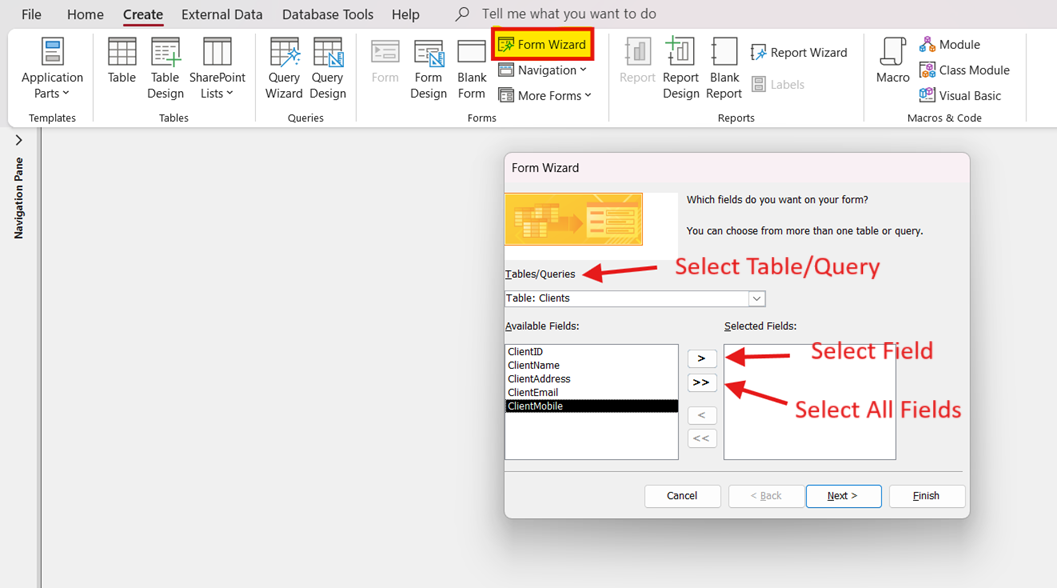
**Types of Commonly Used Controls**

* 1. **Label:** Displays static text or descriptions to guide users in forms.
  2. **Button:** Triggers actions or events, such as saving data or opening other forms.
  3. **Combo Box:** Allows users to select a value from a predefined list or enter custom data.
  4. **Text Box:** Enables users to input or display single-line or multi-line text.

1. Datasheet**:** Displays table or query results in a grid format, like an Excel spreadsheet, for easy data viewing and editing.
2. Reports**:** Generate formatted, printable summaries of data, often used for analysis or presentation.

## Steps to Create

1. Goto Create tab in menu, click Form Wizard and select the table to create a form for and move the fields which we want to see in form.

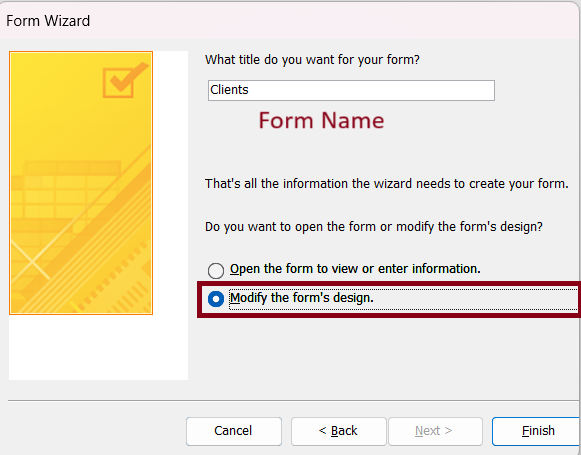


1. Click Next and Choose Columnar layout for the form and hit Next.

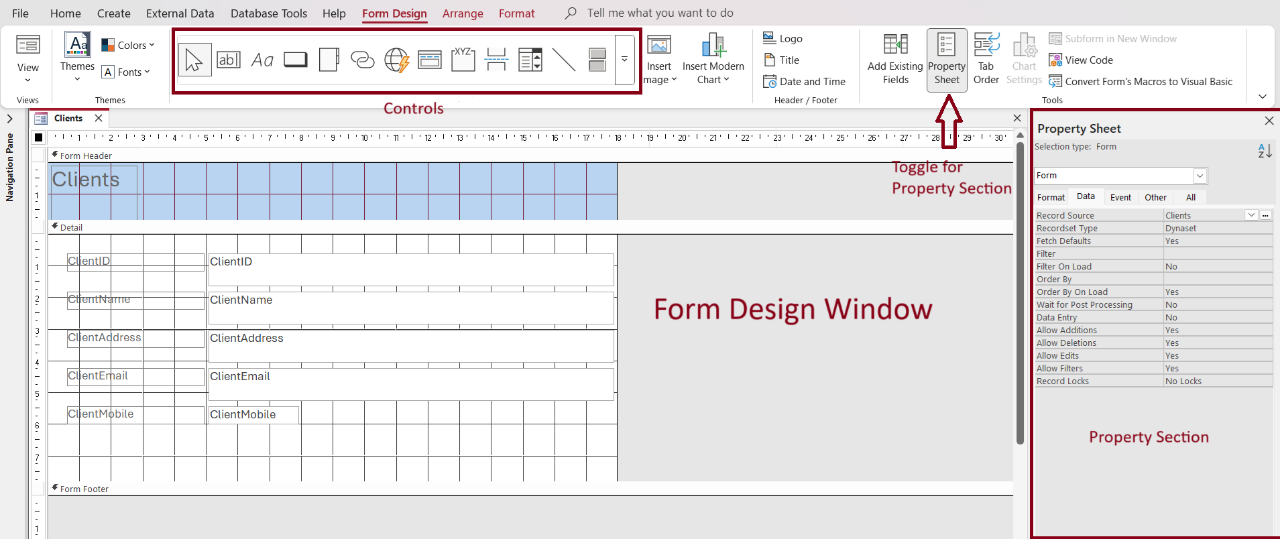
A screenshot of a computer

AI-generated content may be incorrect.

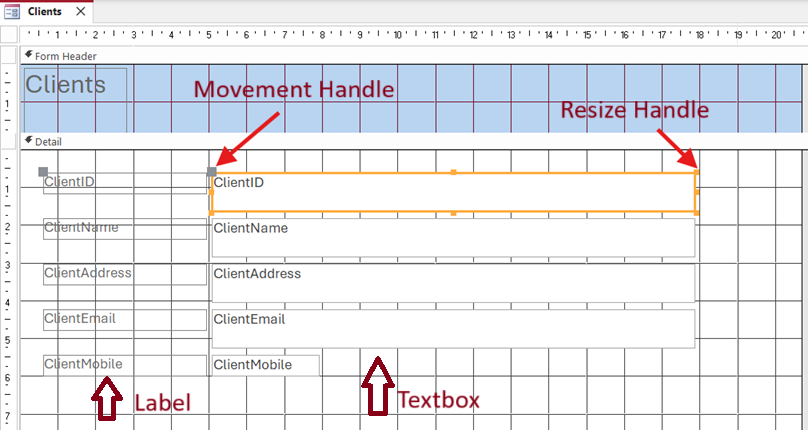
1. Here we need to give name for the form and select the Modify the form’s design to go to form design view. At the end, click finish to create a new form.



1. A new Form Design window will be opened, and it contains all the necessary tools to build a form with full functionality.

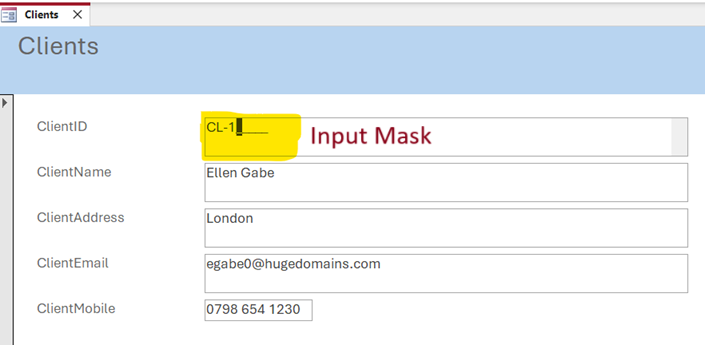


1. We can move, resize using the Handles. The Gray handle is to move the control individually, the corner orange small handle is used for resizing the control. For every Text box, we have a Label which represents what the text box is.



1. Every control in the form has properties. For example, the ClientID has many properties, some are Control Source means, which field it should bind the value to and the Input Mask

A screenshot of a computer

AI-generated content may be incorrect. 

1. Form has three views, Form, Layout and Design View.

A screenshot of a computer

AI-generated content may be incorrect.

1. We can create new buttons to navigate between records, create, save and delete records. We can use button control to do this.

A screenshot of a computer

AI-generated content may be incorrect.

1. After selecting the button control, we need to click and drag the size we want in design view to create a button.

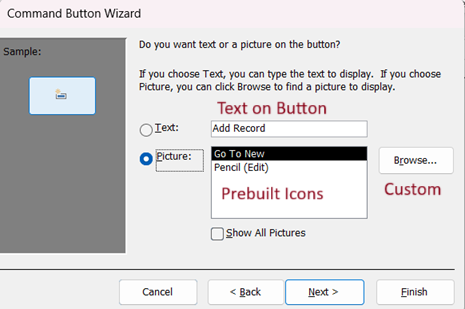
A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

1. After that a menu popup will show up to select which action we want to do when the user clicks a button. There are many options available, but we require only Go to Next/Previous Record, Add New/Save/Delete Record actions. All are available in their categories.
2. Click on Next, and it shows what to display inside the button. We have couple options, Display a pre-built icon or use a custom image from computer or just show the text.



1. After clicking next, it will ask us to give a name for the command, as it stores the login in a command inside Access.

A screenshot of a computer

AI-generated content may be incorrect.

In Form View, it is just a normal view to modify the data. Layout View is used to move the controls around and adjust the layout and position of the elements. Design View is used for Resizing, adding new controls, and all the custom designs and more.

1. Datasheets are also same as forms, but the controls are duplicated in multiple rows to simulate like a table view. To create a Datasheet, we need a data source such as a table or a query. We are using query to generate a datasheet. Select the query/table we want in Objects explorer in left side of window.

A screenshot of a computer

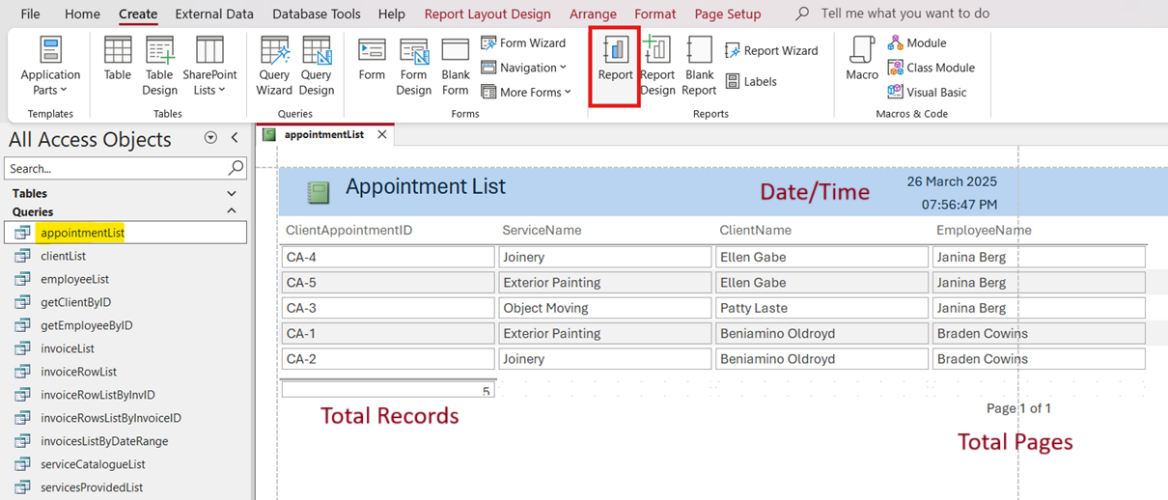
AI-generated content may be incorrect.

1. After clicking on Datasheet, we get something which looks like excel sheet. We can go ahead and save it and give a name.

A screenshot of a computer

AI-generated content may be incorrect.

1. Using the same technique, we can build Reports using queries/tables.



1. The title of the report is editable using double-click on the title.
2. After creating the necessary forms, datasheets and reports. We can combine everything and build a navigation form which can be used as an application.
3. To create a navigation form, we need to go to **Create > Forms > Navigation > Horizontal and Vertical Tabs, Left**. Once created, it should look something like below screenshot.

A screenshot of a computer

AI-generated content may be incorrect.

1. We can add new tabs by double-clicking and giving a name to it.

A screen shot of a computer

AI-generated content may be incorrect.

1. This type of navigation forms works like all horizontal tabs have child vertical tabs associated with it.
2. To add a new entry to vertical tab, We need to drag and drop a form from the Access Object Explorer and drop.

A screenshot of a computer

AI-generated content may be incorrect.

1. To add multiple, we just need to drag and drop each form in that section. We can edit the names of each tab name anything just by double-clicking.

A list of appointment information

AI-generated content may be incorrect.

1. Same rules will be applied for all Forms, Lists and Reports to integrate it in navigation form.

A screenshot of a application

AI-generated content may be incorrect.

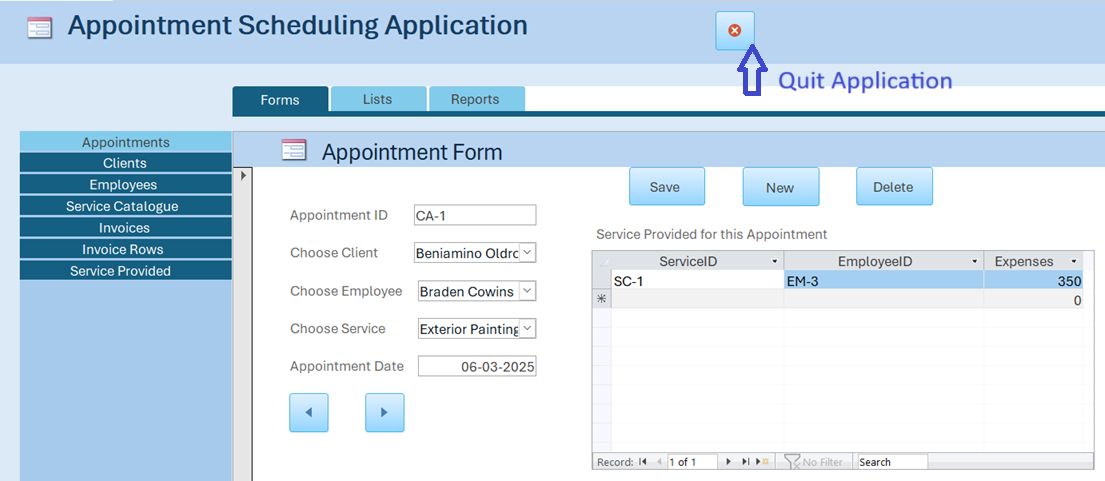
## Appointment Scheduling Database Application

Appointment Scheduling Application is developed using horizontal + vertical navigation form, horizontally we have Forms, Lists and Reports tabs and all of them have their own sub-forms to visualize the data.

For every form, we have Save, New and Delete buttons for ease of use, so user doesn’t have to navigate using built-in controllers which are hard. And to navigate between records, we have next and previous buttons, so user can go to next or previous record easily.

In Forms tab, we have forms to all tables that user can fill.

1. **Appointments Form**: Here for choosing client, we added combo box to select the client using a drop down, instead of entering the Client ID manually. Same process is used for Choose Employee and Choose Service field. A related list is also added which is linked to Services Provided table.



1. **Client Form:** A basic Client Form for modifying, navigating, creating and deleting client information.

A screenshot of a computer application

AI-generated content may be incorrect.

1. **Employee Form:** A basic Employee Form for modifying, navigating, creating and deleting employee information.

A computer screen shot of a application

AI-generated content may be incorrect.

1. **Service Catalogue Form**: A basic Service Catalogue Form for modifying, navigating, creating and deleting Service Catalogue information.

A screenshot of a application

AI-generated content may be incorrect.

1. **Invoices Form:** In Invoices Form, we used related list to get all the Invoice rows which are related to current invoice. This helps for better auditing.

A screenshot of a computer

AI-generated content may be incorrect.

1. **Invoice Rows Form**: In this form, we used combo box to select Invoice and Client Appointment IDs.

A screenshot of a computer application

AI-generated content may be incorrect.

1. **Service Provided Form**: In this form, combo box is used to choose Employee, Service and which Client Appointment is associated with, the Expenses field is formatted to show the pound symbol, so the user can recognize it is a currency field.

A screenshot of a computer

AI-generated content may be incorrect.

In Lists tab, we have data sheets that are generated from the queries that are made in above sections.

1. **Appointments List**: In this list, it shows Service Name, Client Name, Employee Name and Client Appointment ID. So, this gives more useful information than just showing regular IDs that are stored in the table.

A screenshot of a application

AI-generated content may be incorrect.

1. **Clients List**: Lists all the clients in an excel spreadsheet like view.

A screenshot of a computer application

AI-generated content may be incorrect.

1. **Employees List:** Lists all the employees in an excel spreadsheet like view.

**A screenshot of a application

AI-generated content may be incorrect.**

1. **Invoices:** In this list, the InvoiceNumber is shown as it is user-generated instead of InvoiceID. Lists all the invoices in an excel spreadsheet like view.

**A screenshot of a schedule

AI-generated content may be incorrect.**

1. **Services Catalogue:** Lists all the Services Catalogue in an excel spreadsheet like view.

**A screenshot of a computer application

AI-generated content may be incorrect.**

1. **Services Provided:** In this list, actual data is shown instead of IDs.

**A screenshot of a schedule

AI-generated content may be incorrect.**

1. **Invoice Row:** In this list, Service Name and Client Name is taken instead of their IDs for better readability.

**A screenshot of a application

AI-generated content may be incorrect.**

In Reports Tab, we have 2 reports

1. **Invoices**: This gives all the invoice data in one single page with total Invoice value on top right.

A screenshot of a computer

AI-generated content may be incorrect.

1. **Invoices By Month**: This Report is special and gives dynamic data based on user input. When opened, system is prompted to user to give start and end date, so the system can generate an invoice from the start and end date given by the user. Once given, Invoice Report is generated and on top-right, Toval Invoice Value, From Date and To Date is mentioned.

A screenshot of a computer

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer screen

AI-generated content may be incorrect.

# Consideration on Privacy and Security

In the Appointment Scheduling Database for Service-Based SME, the proposed database design does not contain the suggested methods for privacy and security. This presents additional risks