**Lab 1: Creating a Clustered Index**

**Exercise**

1. Create a table named Employees with the following columns:
   * EmployeeID (INT, Primary Key)
   * FirstName (VARCHAR(50))
   * LastName (VARCHAR(50))
   * HireDate (DATE)
2. Create a clustered index on EmployeeID.
3. Insert some sample data and verify the index.

**Solution**

CREATE TABLE Employees (

EmployeeID INT PRIMARY KEY, -- Creates a Clustered Index implicitly

FirstName VARCHAR(50),

LastName VARCHAR(50),

HireDate DATE

);

INSERT INTO Employees VALUES (1, 'John', 'Doe', '2023-01-15'),

(2, 'Jane', 'Smith', '2022-06-20'),

(3, 'Alice', 'Brown', '2021-11-10');

-- Verify the index

EXEC sp\_helpindex 'Employees';

**Lab 2: Creating a Non-Clustered Index**

**Exercise**

1. Create a table Products with columns:
   * ProductID (INT, Primary Key)
   * ProductName (VARCHAR(100))
   * Category (VARCHAR(50))
   * Price (DECIMAL(10,2))
2. Create a non-clustered index on Category and Price.

**Solution**

CREATE TABLE Products (

ProductID INT PRIMARY KEY,

ProductName VARCHAR(100),

Category VARCHAR(50),

Price DECIMAL(10,2)

);

CREATE NONCLUSTERED INDEX IX\_Products\_Category\_Price

ON Products (Category, Price);

-- Verify the index

EXEC sp\_helpindex 'Products';

**Lab 3: Creating a Unique Index**

**Exercise**

1. Create a table Customers with columns:
   * CustomerID (INT, Primary Key)
   * Email (VARCHAR(100), should be unique)
   * Name (VARCHAR(100))
2. Create a unique index on the Email column.

**Solution**

CREATE TABLE Customers (

CustomerID INT PRIMARY KEY,

Email VARCHAR(100) UNIQUE,

Name VARCHAR(100)

);

or

CREATE UNIQUE INDEX IX\_Customers\_Email

ON Customers (Email);

**Lab 4: Creating a Filtered Index**

**Exercise**

1. Create a table Orders with columns:
   * OrderID (INT, Primary Key)
   * CustomerID (INT)
   * Status (VARCHAR(20))
   * TotalAmount (DECIMAL(10,2))
2. Create a filtered index for only Status = 'Completed'.

**Solution**

CREATE TABLE Orders (

OrderID INT PRIMARY KEY,

CustomerID INT,

Status VARCHAR(20),

TotalAmount DECIMAL(10,2)

);

CREATE NONCLUSTERED INDEX IX\_Orders\_Completed

ON Orders (CustomerID, TotalAmount)

WHERE Status = 'Completed';

**Lab 5: Creating a Full-Text Index**

**Exercise**

1. Create a table Articles with columns:
   * ArticleID (INT, Primary Key)
   * Title (VARCHAR(255))
   * Content (TEXT)
2. Create a full-text index on the Content column.

**Solution**

CREATE TABLE Articles (

ArticleID INT PRIMARY KEY,

Title VARCHAR(255),

Content TEXT

);

-- Enable Full-Text Search

EXEC sp\_fulltext\_database 'enable';

-- Create Full-Text Catalog

CREATE FULLTEXT CATALOG ftCatalog AS DEFAULT;

-- Create Full-Text Index

CREATE FULLTEXT INDEX ON Articles (Content)

KEY INDEX PK\_Articles ON ftCatalog;

**Lab 6: Creating an XML Index**

**Exercise**

1. Create a table Documents with columns:
   * DocID (INT, Primary Key)
   * DocData (XML)
2. Create a primary XML index.

**Solution**

CREATE TABLE Documents (

DocID INT PRIMARY KEY,

DocData XML

);

CREATE PRIMARY XML INDEX IX\_Documents\_XML

ON Documents (DocData);

**Lab 7: Creating a Spatial Index**

**Exercise**

1. Create a table Locations with columns:
   * LocationID (INT, Primary Key)
   * GeoData (GEOMETRY)
2. Create a spatial index on GeoData.

**Solution**

CREATE TABLE Locations (

LocationID INT PRIMARY KEY,

GeoData GEOMETRY

);

CREATE SPATIAL INDEX IX\_Locations\_GeoData

ON Locations(GeoData);

**Lab 8: Creating a Columnstore Index**

**Exercise**

1. Create a table Sales with columns:
   * SaleID (INT, Primary Key)
   * ProductID (INT)
   * Amount (DECIMAL(10,2))
   * SaleDate (DATE)
2. Create a columnstore index.

**Solution**

CREATE TABLE Sales (

SaleID INT PRIMARY KEY,

ProductID INT,

Amount DECIMAL(10,2),

SaleDate DATE

);

CREATE CLUSTERED COLUMNSTORE INDEX IX\_Sales\_Columnstore

ON Sales;

**Lab 9: Creating a Composite Index**

**Exercise**

1. Create a table OrdersDetail with columns:
   * OrderID (INT, Foreign Key)
   * ProductID (INT, Foreign Key)
   * Quantity (INT)
   * UnitPrice (DECIMAL(10,2))
2. Create a composite index on OrderID and ProductID.

**Solution**

CREATE TABLE OrdersDetail (

OrderID INT,

ProductID INT,

Quantity INT,

UnitPrice DECIMAL(10,2),

PRIMARY KEY (OrderID, ProductID)

);

or

CREATE NONCLUSTERED INDEX IX\_OrdersDetail\_Order\_Product

ON OrdersDetail (OrderID, ProductID);

**Lab 10: Creating an Indexed View**

**Exercise**

1. Create a table Invoices with columns:
   * InvoiceID (INT, Primary Key)
   * CustomerID (INT)
   * TotalAmount (DECIMAL(10,2))
   * InvoiceDate (DATE)
2. Create a view that calculates the total sales per customer and create an index on it.

**Solution**

CREATE TABLE Invoices (

InvoiceID INT PRIMARY KEY,

CustomerID INT,

TotalAmount DECIMAL(10,2),

InvoiceDate DATE

);

CREATE VIEW vw\_CustomerTotalSales

WITH SCHEMABINDING

AS

SELECT CustomerID, SUM(TotalAmount) AS TotalSales, COUNT\_BIG(\*) AS InvoiceCount

FROM dbo.Invoices

GROUP BY CustomerID;

CREATE UNIQUE CLUSTERED INDEX IX\_vw\_CustomerTotalSales

ON vw\_CustomerTotalSales (CustomerID);