مشروع نظام إدارة حجوزات فندق

شرح تفصيلي عن كيفية انشاء المشروع

اهداف المشروع

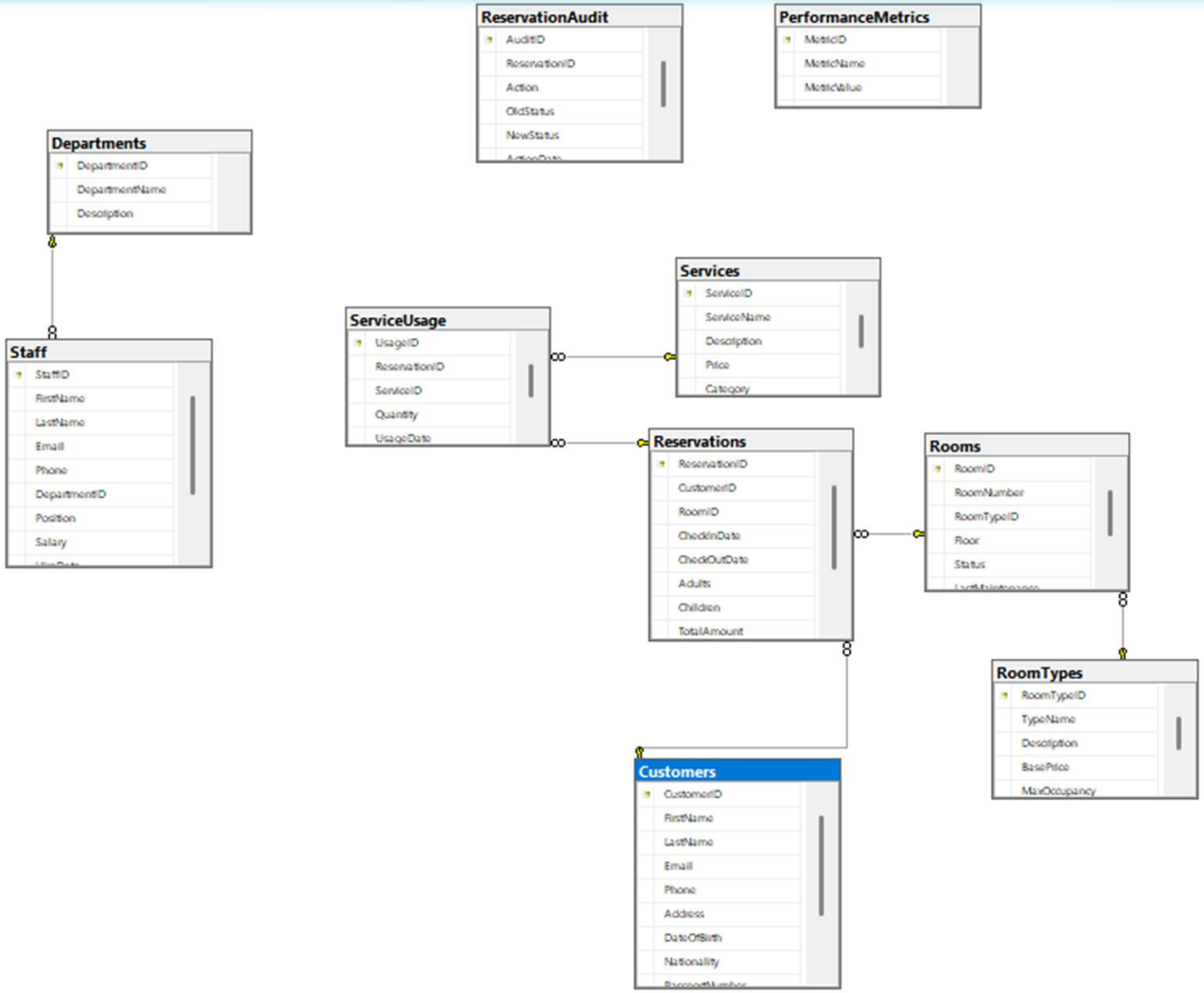
يهدف هذا المشروع إلى تصميم وتطوير نظام متكامل لإدارة حجوزات الفنادق باستخدام قاعدة بيانات احترافية تعتمد على تقنيات SQL SERVER.
تم بناء النظام ليغطي جميع الجوانب التشغيلية الأساسية التي تحتاجها إدارة الفندق، من تسجيل بيانات العملاء والحجوزات إلى إدارة الغرف والخدمات، مع ضمان الأمان، والمرونة، وإمكانية التوسع.
تم تطبيق معايير عالية في التصميم مثل **التطبيع (NORMALIZATION)**، **القيود المرجعية (FOREIGN KEYS)**، **المحفزات (TRIGGERS)**، **الإجراءات المخزنة (STORED PROCEDURES)**، إضافة إلى **الفهارس (INDEXES)** و**التقسيم (PARTITIONING)** لتحقيق أفضل أداء.

يوفر النظام وظائف حيوية تشمل:

- إنشاء وتعديل وحذف الحجوزات والغرف والعملاء.
- التحقق من توفر الغرف تلقائيًا عند الحجز.
- تسجيل الوصول والمغادرة وتحديث حالة الغرفة تلقائيًا.
- حساب تكلفة الإقامة بناءً على نوع الغرفة وعدد الليالي.
- تسجيل تغييرات حالة الحجز في جدول تدقيق (Audit).
- توفير تقارير تحليلية متقدمة مثل تقارير الإيرادات، وتقييم العملاء، واستغلال الغرف.
- دعم الأدوار الوظيفية المختلفة (مدير، موظف استقبال، محاسب) بصلاحيات محددة.
- إعداد خطة نسخ احتياطي شاملة لحماية البيانات.
- دعم المراقبة الآلية لأداء النظام وحجمه.

ملخص

هذا المشروع يمثل نموذجًا حقيقيًا لتطبيق قواعد البيانات في بيئة واقعية، ويجمع بين الجانب النظري (التصميم والبنية) والجانب العملي (الأداء، الأمان، المراقبة). وقد تم اعتماده على قائمة فحص احترافية لضمان جاهزيته للإطلاق في بيئة تشغيل فعلية.


ReservationAudit
AuditID
ReservationID
Action
OldStatus
NewStatus
PerformanceMetrics
MetricID
MetricName
MetricValue
Departments
DepartmentID
DepartmentName
Description
Services
ServiceID
ServiceName
Description
Price
Category
ServiceUsage
UsageID
ReservationID
ServiceID
Quantity
UsageDate
Staff
StaffID
FirstName
LastName
Email
Phone
DepartmentID
Position
Salary
Reservations
ReservationID
CustomerID
RoomID
CheckInDate
CheckOutDate
Adults
Children
TotalAmount
Rooms
RoomID
RoomNumber
RoomTypeID
Floor
Status
RoomTypes
RoomTypeID
TypeName
Description
BasePrice
MaxOccupancy
Customers
CustomerID
FirstName
LastName
Email
Phone
Address
DateOfBirth
Nationality

التحليل الكامل لجميع الأكواد والشرح التفصيلي:

1. إنشاء الجداول (TABLES AND NORMALIZATION)

- **CUSTOMERS (العملاء)**

هذا الجدول يحتوي على معلومات العميل مثل الاسم، البريد الإلكتروني، الهاتف، الجنسية، إلخ.

```
CREATE TABLE Customers (
    CustomerID INT PRIMARY KEY IDENTITY(1,1),
    FirstName NVARCHAR(50) NOT NULL,
    LastName NVARCHAR(50) NOT NULL,
    Email NVARCHAR(100) UNIQUE NOT NULL,
    Phone NVARCHAR(20),
    Address NVARCHAR(200),
    DateOfBirth DATE,
    Nationality NVARCHAR(50),
    PassportNumber NVARCHAR(50),
    CreatedDate DATETIME DEFAULT GETDATE(),
    ModifiedDate DATETIME DEFAULT GETDATE()
);
```

- **RoomTypes (انواع الغرف)**

أنواع الغرف مثل مفردة، مزدوجة، جناح، مع الأسعار والطاقة الاستيعابية.

```
CREATE TABLE RoomTypes (
    RoomTypeID INT PRIMARY KEY IDENTITY(1,1),
    TypeName NVARCHAR(50) NOT NULL,
    Description NVARCHAR(200),
    BasePrice DECIMAL(10,2) NOT NULL,
    MaxOccupancy INT NOT NULL,
    CreatedDate DATETIME DEFAULT GETDATE()
);
```

- **ROOMS (الغرف)**

يمثل كل غرفة في الفندق ومعلوماتها وربطها بنوعها.

```
CREATE TABLE Rooms (
    RoomID INT PRIMARY KEY IDENTITY(1,1),
    RoomNumber NVARCHAR(10) NOT NULL UNIQUE,
    RoomTypeID INT NOT NULL,
    Floor INT NOT NULL,
    Status NVARCHAR(20) DEFAULT 'Available',
    LastMaintenance DATE,
    CreatedDate DATETIME DEFAULT GETDATE(),
    CONSTRAINT FK_Rooms_RoomTypes FOREIGN KEY
(RoomTypeID) REFERENCES RoomTypes(RoomTypeID)
);
```

- **RESERVATIONS (الحجوزات)**

يرتبط بكل عميل وغرفة، ويحتوي على تواريخ الحجز وعدد الأفراد.

```
CREATE TABLE Reservations (
    ReservationID INT PRIMARY KEY IDENTITY(1,1),
    CustomerID INT NOT NULL,
    RoomID INT NOT NULL,
    CheckInDate DATE NOT NULL,
    CheckOutDate DATE NOT NULL,
    Adults INT NOT NULL DEFAULT 1,
    Children INT DEFAULT 0,
    TotalAmount DECIMAL(10,2),
    Status NVARCHAR(20) DEFAULT 'Confirmed',
    CreatedDate DATETIME DEFAULT GETDATE(),
    CONSTRAINT FK_Reservations_Customers FOREIGN KEY
(CustomerID) REFERENCES Customers(CustomerID),
    CONSTRAINT FK_Reservations_Rooms FOREIGN KEY (RoomID)
REFERENCES Rooms(RoomID)
);
```

2. TRIGGERS (المحفزات)

- **تحديث تاريخ تعديل العميل**

يتم تحديث ModifiedDate تلقائيًا عند تعديل بيانات العميل.

```
CREATE TRIGGER tr_UpdateCustomerModifiedDate
ON Customers
AFTER UPDATE
AS
BEGIN
    SET NOCOUNT ON;
    UPDATE Customers
    SET ModifiedDate = GETDATE()
    WHERE CustomerID IN (SELECT CustomerID FROM inserted);
END;
```

- **سجل تغييرات حالة الحجز(Audit)**

يسجل في جدول ReservationAudit أي تغيير في حالة الحجز.

sql

```
CREATE TRIGGER tr_ReservationAudit
ON Reservations
AFTER UPDATE
AS
BEGIN
    SET NOCOUNT ON;
    INSERT INTO ReservationAudit (ReservationID, Action, OldStatus, NewStatus)
    SELECT
        i.ReservationID,
        'UPDATE',
        d.Status,
        i.Status
    FROM inserted i
    JOIN deleted d ON i.ReservationID = d.ReservationID
    WHERE i.Status != d.Status;
END;
```

3. VIEWS (العروض)

- **الحجوزات النشطة**

لعرض الحجوزات ذات الحالة CONFIRMEDأو CHECKEDIN.

```
CREATE VIEW vw_ActiveReservations1
AS
SELECT
    r.ReservationID,
    c.FirstName + ' ' + c.LastName AS CustomerName,
    rm.RoomNumber,
    rt.TypeName AS RoomType,
    r.CheckInDate,
    r.CheckOutDate,
    r.TotalAmount,
    r.Status
FROM Reservations r
JOIN Customers c ON r.CustomerID = c.CustomerID
JOIN Rooms rm ON r.RoomID = rm.RoomID
JOIN RoomTypes rt ON rm.RoomTypeID = rt.RoomTypeID
WHERE r.Status IN ('Confirmed', 'CheckedIn');
```

- **الغرف المتاحة**

لعرض الغرف المتاحة للحجز فقط.

```
CREATE VIEW vw_AvailableRooms1
AS
SELECT
    rm.RoomID,
    rm.RoomNumber,
    rt.TypeName,
    rt.Description,
    rt.BasePrice,
    rt.MaxOccupancy,
    rm.Floor
FROM Rooms rm
JOIN RoomTypes rt ON rm.RoomTypeID = rt.RoomTypeID
WHERE rm.Status = 'Available';
```

4. ADVANCED REPORTS(التقارير المتقدمة)

تقرير استخدام الغرف

```
SELECT
    rm.RoomNumber,
    rt.TypeName,
    COUNT(r.ReservationID) AS TotalBookings,
    SUM(r.TotalAmount) AS TotalRevenue,
    AVG(DATEDIFF(DAY, r.CheckInDate, r.CheckOutDate)) AS
AvgStayDuration
FROM Rooms rm
JOIN RoomTypes rt ON rm.RoomTypeID = rt.RoomTypeID
LEFT JOIN Reservations r ON rm.RoomID = r.RoomID
WHERE r.Status = 'CheckedOut'
GROUP BY rm.RoomNumber, rt.TypeName
ORDER BY TotalRevenue DESC;
```

العملاء الأعلى قيمة

```
SELECT
    c.CustomerID,
    c.FirstName + ' ' + c.LastName AS CustomerName,
    COUNT(r.ReservationID) AS TotalReservations,
    SUM(r.TotalAmount) AS TotalSpent,
    AVG(r.TotalAmount) AS AvgSpentPerVisit,
    MAX(r.CheckInDate) AS LastVisit,
    CASE
        WHEN SUM(r.TotalAmount) > 10000 THEN 'VIP'
        WHEN SUM(r.TotalAmount) > 5000 THEN 'Premium'
        ELSE 'Regular'
    END AS CustomerTier
FROM Customers c
JOIN Reservations r ON c.CustomerID = r.CustomerID
GROUP BY c.CustomerID, c.FirstName, c.LastName
ORDER BY TotalSpent DESC;
```

تقريرلإشغالاليومي

```
SELECT
    CAST(r.CheckInDate AS DATE) AS ReservationDate,
    COUNT(*) AS TotalReservations,
    SUM(r.TotalAmount) AS DailyRevenue,
    AVG(r.TotalAmount) AS AvgReservationValue
FROM Reservations r
WHERE r.Status = 'CheckedIn'
    AND r.CheckInDate >= DATEADD(DAY, -30, GETDATE())
GROUP BY CAST(r.CheckInDate AS DATE)
ORDER BY ReservationDate DESC;
```

5. STORED PROCEDURES (الإجراءات المخزنة)

إنشاء حجز جديد – sp_CreateReservation
يتحقق من توفر الغرفة، يحسب السعر الإجمالي، ثم يُدرج الحجز ويحدث حالة الغرفة.

```
CREATE PROCEDURE sp_CreateReservation
    @CustomerID INT,
    @RoomID INT,
    @CheckInDate DATE,
    @CheckOutDate DATE,
    @Adults INT = 1,
    @Children INT = 0
AS
BEGIN
    SET NOCOUNT ON;

    BEGIN TRANSACTION;

    BEGIN TRY
        -- التحقق من توفر الغرفة
        IF EXISTS (
            SELECT 1 FROM Reservations
            WHERE RoomID = @RoomID
            AND Status IN ('Confirmed', 'CheckedIn')
            AND ((@CheckInDate BETWEEN CheckInDate AND
CheckOutDate)
                OR (@CheckOutDate BETWEEN CheckInDate AND
CheckOutDate))
        )
        BEGIN
            RAISERROR('الغرفة غير متوفرة في التواريخ المحددة', 16, 1);
            ROLLBACK TRANSACTION;
            RETURN;
        END

        -- حساب التكلفة الإجمالية
        DECLARE @TotalAmount DECIMAL(10,2);
        DECLARE @Days INT = DATEDIFF(DAY, @CheckInDate,
@CheckOutDate);

        SELECT @TotalAmount = @Days * rt.BasePrice
        FROM Rooms r
        JOIN RoomTypes rt ON r.RoomTypeID = rt.RoomTypeID
        WHERE r.RoomID = @RoomID;

        -- إدراج الحجز
        INSERT INTO Reservations (CustomerID, RoomID,
CheckInDate, CheckOutDate, Adults, Children, TotalAmount)
        VALUES (@CustomerID, @RoomID, @CheckInDate,
@CheckOutDate, @Adults, @Children, @TotalAmount);

        -- تحديث حالة الغرفة
        UPDATE Rooms SET Status = 'Reserved' WHERE RoomID
= @RoomID;

        COMMIT TRANSACTION;
        PRINT 'تم إنشاء الحجز بنجاح';

    END TRY
    BEGIN CATCH
        ROLLBACK TRANSACTION;
        THROW;
    END CATCH
END;
```

تسجيل الوصول – sp_CheckIn

يغير حالة الحجز إلىCheckedIn ، والغرفة إلى.Occupied

```
CREATE PROCEDURE sp_CheckIn
    @ReservationID INT
AS
BEGIN
    SET NOCOUNT ON;

    BEGIN TRANSACTION;

    BEGIN TRY
        -- التحقق من وجود الحجز
        IF NOT EXISTS (SELECT 1 FROM Reservations WHERE ReservationID = @ReservationID)
        BEGIN
            RAISERROR('الحجز غير موجود', 16, 1);
            ROLLBACK TRANSACTION;
            RETURN;
        END
-- تحديث حالة الحجز
        UPDATE Reservations
        SET Status = 'CheckedIn'
        WHERE ReservationID = @ReservationID;

        -- تحديث حالة الغرفة
        UPDATE Rooms
        SET Status = 'Occupied'
        WHERE RoomID = (SELECT RoomID FROM Reservations WHERE ReservationID =
@ReservationID);

        COMMIT TRANSACTION;
        PRINT 'تم تسجيل الوصول بنجاح';

    END TRY
    BEGIN CATCH
        ROLLBACK TRANSACTION;
        THROW;
    END CATCH
END;
```

sp_MonitorDatabaseSize – مراقبة حجم قاعدة البيانات

يسجل الحجم الحالي لقاعدة البيانات في جدول PerformanceMetrics.

```
CREATE PROCEDURE sp_MonitorDatabaseSize
AS
BEGIN
    INSERT INTO PerformanceMetrics (MetricName, MetricValue)
    SELECT
        'Database Size MB',
        SUM(CAST(size AS DECIMAL(10,2)) * 8 / 1024)
    FROM sys.database_files;
END;
```

sp_CheckOut – تسجيل المغادرة

يحول الحجز إلى CheckedOut ويجعل الغرفة Available.

```
CREATE PROCEDURE sp_CheckOut
    @ReservationID INT
AS
BEGIN
    SET NOCOUNT ON;

    BEGIN TRANSACTION;

    BEGIN TRY
        -- تحديث حالة الحجز
        UPDATE Reservations
        SET Status = 'CheckedOut'
        WHERE ReservationID = @ReservationID;

        -- تحديث حالة الغرفة
        UPDATE Rooms
        SET Status = 'Available'
        WHERE RoomID = (SELECT RoomID FROM Reservations WHERE ReservationID =
@ReservationID);

        COMMIT TRANSACTION;
        PRINT 'تم تسجيل المغادرة بنجاح';

    END TRY
    BEGIN CATCH
        ROLLBACK TRANSACTION;
        THROW;
    END CATCH
END;
```

ROLES AND PERMISSIONS (الأدوار والصلاحيات)

✅ إنشاء الأدوار:

```
CREATE ROLE HotelManager;
CREATE ROLE ReceptionistRole;
CREATE ROLE AccountantRole;
```

✅ منح الصلاحيات للدور HotelManager (مدير الفندق)

```
GRANT SELECT, INSERT, UPDATE, DELETE
ON Reservations TO HotelManager;

GRANT SELECT, INSERT, UPDATE, DELETE
ON Customers TO HotelManager;

GRANT SELECT, INSERT, UPDATE, DELETE
ON Rooms TO HotelManager;

GRANT SELECT ON Staff TO
HotelManager;
```

✅ منح الصلاحيات للدور
RECEPTIONISTROLE(موظف الاستقبال)

```
GRANT SELECT, INSERT, UPDATE ON
Reservations TO ReceptionistRole;

GRANT SELECT, INSERT, UPDATE ON
Customers TO ReceptionistRole;

GRANT SELECT ON Rooms TO
ReceptionistRole;

GRANT SELECT ON Services TO
ReceptionistRole;
```

✅ منح الصلاحيات للدور
ACCOUNTANTROLE(المحاسب)

```
GRANT SELECT ON Reservations TO
AccountantRole;

GRANT SELECT ON ServiceUsage TO
AccountantRole;

GRANT SELECT ON Customers TO
AccountantRole;
```

7. INDEXING (الفهارس)

```
CREATE INDEX IX_Reservations_CheckInDate
ON Reservations(CheckInDate);

CREATE INDEX IX_Reservations_CustomerID ON
Reservations(CustomerID);

CREATE INDEX IX_Rooms_Status ON
Rooms(Status);

CREATE INDEX IX_Customers_Email ON
Customers(Email);
```

لتحسين سرعة البحث والاستعلام:

- على CheckInDate
- على CustomerID
- على Status
- على Email

8. FUNCTIONS

fn_CalculateCustomerDiscount

تحسب خصم للعميل حسب عدد الحجوزات.

```
CREATE FUNCTION fn_CalculateCustomerDiscount(
    @CustomerID INT
)RETURNS DECIMAL(5,2)
AS
BEGIN
    DECLARE @TotalBookings INT;
    DECLARE @DiscountPercent DECIMAL(5,2) = 0;
    SELECT @TotalBookings = COUNT(*)
    FROM Reservations
    WHERE CustomerID = @CustomerID;
    IF @TotalBookings >= 10
        SET @DiscountPercent = 15.0;
ELSE IF @TotalBookings >= 5
        SET @DiscountPercent = 10.0;
    ELSE IF @TotalBookings >= 2
        SET @DiscountPercent = 5.0;

    RETURN @DiscountPercent;
END;
```

9.(ACKUP PLAN)النسخ الاحتياطي

🛡️ خطة النسخ الاحتياطي (Backup and Disaster Recovery Plan)

تُعد خطة النسخ الاحتياطي جزءًا حيويًا من أي نظام قواعد بيانات احترافي، حيث تحمي البيانات من الضياع الناتج عن الأعطال المفاجئة أو الهجمات الإلكترونية أو الأخطاء البشرية. في هذا المشروع، تم اعتماد خطة شاملة تعتمد على 3 أنواع رئيسية من النسخ الاحتياطي:

النسخة الاحتياطية الكاملة (FULL BACKUP)

- **الوصف:** تقوم بنسخ جميع بيانات قاعدة البيانات بالكامل، وهي الأساس لأي خطة نسخ احتياطي.
- **التكرار الموصى به:** مرة يوميًا أو أسبوعيًا حسب حجم البيانات.
- **الكود المستخدم:**

```
BACKUP DATABASE HotelManagement_Main

TO DISK = 'C:\Program Files\Microsoft SQL Server\MSSQL16.MSSQLSERVER\MSSQL\DATA\HotelManagement_Full.bak'

WITH FORMAT, COMPRESSION, CHECKSUM, STATS = 10;
```

- **التفصيل:**
  - FORMAT: يُنشئ ملف نسخ احتياطي جديد تمامًا.
  - COMPRESSION: يقلل من حجم الملف.
  - CHECKSUM: يتحقق من سلامة البيانات أثناء النسخ.
  - STATS = 10: يعرض نسبة التقدم كل 10%.

النسخة الاحتياطية التفاضلية (DIFFERENTIAL BACKUP)

- **الوصف:** تخزن فقط التغييرات التي حدثت منذ آخر **نسخة احتياطية كاملة**.
- **الفائدة:** أسرع وأصغر من النسخ الكاملة.
- **التكرار الموصى به:** عدة مرات يوميًا.
- **الكود المستخدم:**

```
BACKUP DATABASE HotelManagement_Main

TO DISK = 'C:\Program Files\Microsoft SQL Server\MSSQL16.MSSQLSERVER\MSSQL\DATA\HotelManagement_Diff.bak'

WITH DIFFERENTIAL, COMPRESSION, CHECKSUM, STATS = 10;
```

- **التفصيل:**
  - DIFFERENTIAL: يحدد أن النسخة التفاضلية.
  - يجب أن تسبقها نسخة كاملة ليكون لها معنى.

نسخة سجل المعاملات (TRANSACTION LOG BACKUP)

- **الوصف:** تحفظ كل العمليات التي تمت منذ آخر نسخة سجل.
- **الهدف:** تمكين **الاستعادة إلى نقطة زمنية محددة (POINT-IN-TIME RECOVERY)**
- **التكرار الموصى به:** كل ساعة أو أقل في الأنظمة الحساسة.
- **الكود المستخدم:**

```
BACKUP LOG HotelManagement_Main

TO DISK = 'C:\Program Files\Microsoft SQL Server\MSSQL16.MSSQLSERVER\MSSQL\DATA\HotelManagement_Log.trn'

WITH COMPRESSION, CHECKSUM, STATS = 10;
```

- **التفصيل:**
  - يعتمد على أن يكون وضع سجل المعاملات(Recovery Model)مضبوطًا على **FULL**.
  - يُستخدم عند الرغبة في استعادة قاعدة البيانات حتى لحظة قبل حدوث مشكلة.

فوائد الخطة المستخدمة 🧠

النوع	الحجم	السرعة	الغرض
Full Backup	كبير	أبطأ	نسخة رئيسية شاملة
Differential Backup	متوسط	سريع	تتبع التغييرات منذ آخر نسخة كاملة
Log Backup	صغير جدًا	سريع جدًا	استعادة دقيقة حتى لحظة معينة

في ختام هذا المشروع، تم تصميم وتنفيذ نظام متكامل لإدارة حجوزات الفنادق يعتمد على قاعدة بيانات قوية ومترابطة باستخدام SQL SERVER. يغطي هذا النظام جميع الوظائف الأساسية والعمليات اليومية التي يحتاجها الفندق، بما في ذلك إدارة العملاء، الغرف، الحجوزات، التقارير، النسخ الاحتياطي، والتحكم في الصلاحيات.
تميز النظام بالمرونة والكفاءة من خلال استخدام الإجراءات المخزنة، والمشاهدات (VIEWS)، والمحفزات (TRIGGERS)، والفهارس (INDEXES)، بالإضافة إلى تطبيق خطة نسخ احتياطي فعالة لحماية البيانات. كما تم اعتماد منهجية منظمة في بناء الجداول والعلاقات لضمان التكامل المرجعي ودعم التوسع المستقبلي.
إن هذا النظام لا يمثل فقط حلاً تقنيًا، بل يُعد نموذجًا عمليًا يمكن تطويره لاحقًا ليشمل خدمات إضافية مثل الدفع الإلكتروني، أو الربط مع بوابات الحجز العالمية، مما يعزز من كفاءة تشغيل الفندق وتحسين تجربة النزيل.

تم انشاء هذا المشروع بواسطة

احمد الحاج احمد العلو **عبد الباسط قوجه علي** **احمد عمر عباس**