TRAINING360

Tartalomjegyzék

Hallgató neve: Tóth László Balázs

Hallgató e-mail címe: tothlaszlo.b@gmail.com

A választott téma rövid elnevezése: Autókölcsönző rendszer

A téma szöveges leírása

Az adatbázis egy több telephelyes, többnyire nagy értékű autókat bérbeadó autókölcsönző igényeit szolgálja ki.
Az adatbázis tárolja a felhasználók és az alkalmazottak személyes és felhasználói adatait is. A bérelhető járművek adatai több táblán vannak elosztva. Külön táblában találhatóak a főbb adatok, mint a gyártó, modell, rendszám és alvázszám, külön táblában az adott jármű paraméterei, illetve külön táblában azok felszereltsége és kiegészítői. Az adatbázis alkalmas a kliensprogramon keresztül az autókhoz feltöltött képek tárolására is.
A foglalások történhetnek online, böngészős felületen a felhasználók által, illetve meghatározott jogosultsággal a dolgozók is hozzáférnek az adatbázishoz kliensprogramon keresztül így foglalást és kölcsönzést tudnak rögzíteni a telephelyen személyesen is.

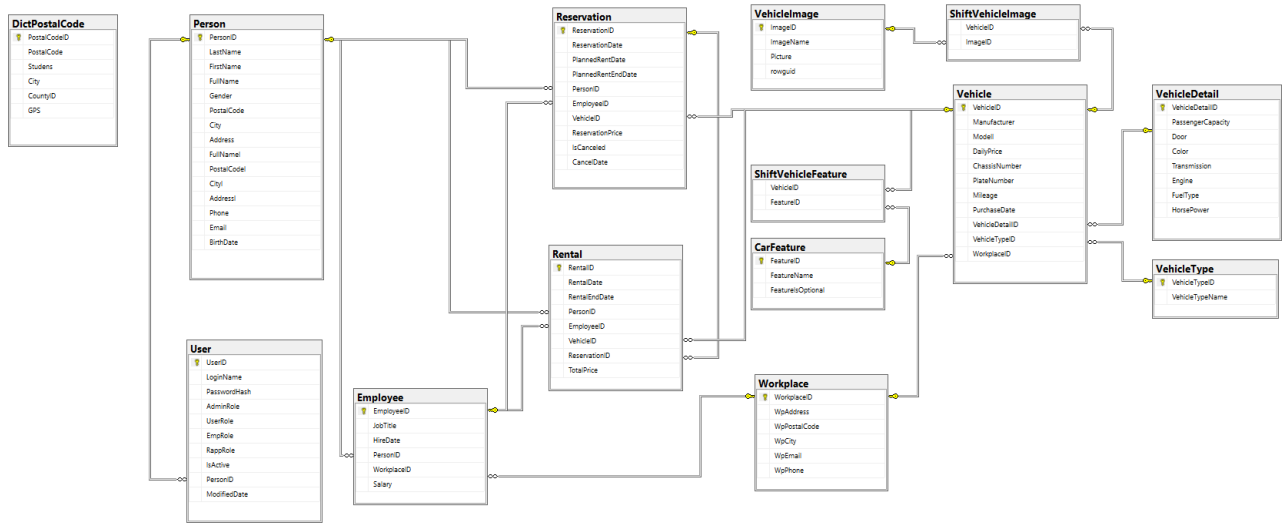
A CarRental adatbázis az alábbi adatokat tárolja:

- A kölcsönzők személyes és céges adatai (Person tábla)
- A felhasználók adatai (User tábla)
- Az alkalmazottak adatai (Employee tábla)
- A telephelyek adatai (Workplace tábla)
- A járművek nyilvántartása (Vehicle tábla)
- Részletes adatok a járművekről (VehicleDetail tábla)
- Különböző járműtípusok nyilvántartása (VehiclyType tábla)
- A járműhöz tartozó kiegészítők és a hozzá kérhető opcionális kiegészítők nyilvántartása (CarFeature tábla)
- Képek az adott járműről (VehicleImage tábla)
- A felhasználók által leadott foglalások (Reservation tábla)
- Kölcsönzések nyilvántartása (Rental tábla)

A CarRental adatbázis az alábbi funkciók ellátását biztosítja:

- A tárolt adatok adatminőségének biztosítása különféle megszorítások és idegen kulcsok révén.
- Adatbiztonság biztosítása a megfelelő jogosultsági rendszer kialakításával, mindenkinek a szükséges minimális jogosultsága van az adatbázishoz.
- Adatkezeléssel kapcsolatos elvárások biztosítása, a törölt felhasználók személyes adatai 5 év után automatikusan törlődnek a rendszerből ütemezett tárolt eljárás segítségével.
- Üzleti elemzéshez szükséges lekérdezések biztosítása a pbi sémán keresztül.
- Kliensprogram és webalkalmazás támogatása tárolt eljárások révén, amely segítségével új foglalás vagy kölcsönzés rögzíthető.

Adatbázis diagramm



Sémák

Emp

Ebbe a sémába azok a tárolt eljárások kerülnek, amelyeket az EmpRole jogosultsággal ellátott alkalmazottak futtatni tudnak. Erre a sémára EXEC joga van az EmpRole adatbázis szerepkörnek.

pbi

Ebbe a sémába azok a nézetek kerültek, amelyek üzleti elemzések készítésére alkalmasak Excel-ben vagy Power BI alkalmazásban.

Rapp

Ebbe a sémába azok a tárolt eljárások kerülnek, amely a RentalWebApp alkalmazás szerepkör számára vannak fenntartva.

Táblák

Dbo.Person

A felhasználók és alkalmazottak személyes adatait tartalmazó tábla.

Mezőleírás

Oszlop neve	Adattípus	Kötelező	Default	Értelmezés	Ellenőrzés
PersonID	int	NOT NULL	(IDENTITY)	Személy egyedi azonosítója	PK
LastName	varchar(50)	NOT NULL		Személy neve	
FirstName	varchar(50)	NOT NULL		Személy neve	
FullName	varchar(100)	NOT NULL	Számított mező	Személy teljes neve Számított mező	
Gender	tinyint	NOT NULL		Személy neme	CK
PostalCode	varchar(8)	NOT NULL		Irányítószám	
City	Varchar(50)	NOT NULL		Város megnevezése	
Address	Varchar(150)	NOT NULL		Cím	
FullNameI	varchar(100)	NOT NULL		Számlázási név	
PostalCodeI	varchar(8)	NULL		Számlázási irányítószám	
CityI	Varchar(50)	NULL		Számlázási város	
AddressI	Varchar(150)	NULL		Számlázási cím	
Phone	Varchar(50)	NOT NULL		Telefonszám	
Email	Varchar(50)	NOT NULL		E-mail cím	CK
BirthDate	Date	NOT NULL		Születési dátum	

Táblaszintű megszorítások

Megszorítás neve	Megszorítás fajtája (trigger, constraint)	Értelmezés
Check_Email	CONSTRAINT	Az Email mezőnek meg kell felelnie a regexre: '%[A-Z0-9][@][A-Z0-9]%[.][A-Z0-9]%'
Check_Gender	CONSTRAINT	Gender értéke 1 vagy 2 lehet

Indexek

Index neve	Oszlop (ok)	Index típusa	Értelmezés
PK_Person_PersonID	PersonID	Clustered (RS)	Elsődleges kulcs
UIX_Person_Email	Email	Unique non-clustered	

Dbo.Employee

Az alkalmazottak céges adatait tartalmazó tábla.

Mezőleírás

Oszlop neve	Adattípus	Kötelező	Default	Értelmezés	Ellenőrzés
EmployeeID	int	NOT NULL	(IDENTITY)	Dolgozó azonosító	PK
JobTitle	varchar(50)			Munkakör	
HireDate	date	NOT NULL		Felvétel dátuma	
PersonID	int	NOT NULL		Személy azonosító, idegen kulcs a Person táblára	FK
WorkPlaceID	int	NOT NULL		Munkahely azonosító, idegenkulcs WorkPlace táblára	FK
Salary	money	NOT NULL		Fizetés	Fizetés > 0

Táblaszintű megszorítások

Megszorítás neve	Megszorítás fajtája (trigger, constraint)	Értelmezés
Check_Salary	CONSTRAINT	Salary > 0

Indexek

Index neve	Oszlop (ok)	Index típusa	Értelmezés
PK_Employee_EmployeeID	EmployeeID	Clustered (RS)	Elsődleges kulcs

Táblakapcsolatok

Idegen kulcs neve	Kapcsolat típusa	Delete és Update szabály
FK_Employee_PersonID_Person_PersonID	1 : 1	DELETE=NO ACTION UPDATE=NO ACTION
FK_Employee_WorkplaceID_Workplace_WorkplaceID	1 : 1	DELETE=NO ACTION UPDATE=NO ACTION

Dbo.User

A felhasználók bejelentkezési adatait és jogosultságait tartalmazó tábla.

Mezőleírás

Oszlop neve	Adattípus	Kötelező	Default	Értelmezés	Ellenőrzés
UserID	int	NOT NULL	(IDENTITY)	Felhasználó azonosító	PK
LoginName	varchar(30)	NOT NULL		Felhasználónév	UIX
PasswordHash	Binary(64)	NOT NULL		Jelszó	
AdminRole	bit	NOT NULL	0	1 = admin jogkör	
EmpRole	bit	NOT NULL	0	1 = dolgozói jogkör	
RappRole	bit	NOT NULL	0	1 = alkalmazás	
IsActive	bit	NOT NULL	0	1 = aktív felhasználó	
PersonID	int	NOT NULL		Idegenkulcs a Person táblára	
UserRole	bit	NOT NULL	1	1 = felhasználói jogkör	
ModifiedDate	datetime2(7)			Felhasználó módosításának dátuma	

Indexek

Index neve	Oszlop (ok)	Index típusa	Értelmezés
PK_User_UserID	UserID	Clustered (RS)	Elsődleges kulcs
UIX_User_LoginName	LoginName	Unique non-clustered	Ne lehessen két egyforma

Triggerek

Trigger neve	Trigger típusa	Értelmezés
trg_IoD_InsteadDeleteUser	DML (DELETE)	A User táblán történő törlés esetén, az IsActive mezőt 0-ra állítja, a jogosultságokat megszünteti és a ModifiedDate mezőbe beírja az aktuális dátumot

Trigger T-SQL kódja

```
CREATE OR ALTER TRIGGER trg_IoD_InsteadDeleteUser
 ON dbo.[User]
 INSTEAD OF DELETE
AS
BEGIN
 SET NOCOUNT ON
 UPDATE [User]
     SET IsActive = 0,
         EmpRole = 0,
         AdminRole = 0,
         RappRole = 0,
         UserRole = 0,
         ModifiedDate = SYSDATETIME()
     FROM [User] U INNER JOIN deleted D ON U.UserID = D.UserID
     WHERE U.UserID = D.UserID
END
```

Táblakapcsolatok

Idegen kulcs neve	Kapcsolat típusa	Delete és Update szabály
FK_User_PersonID_Person_PersonID	1 : 1	DELETE=NO ACTION UPDATE=NO ACTION

Dbo.Vehicle

A járművek alapadatait tartalmazó tábla.

Mezőleírás

Oszlop neve	Adattípus	Kötelező	Default	Értelmezés	Ellenőrzés
VehicleID	int	NOT NULL	(IDENTITY)	Jármű azonosító	PK
ManuFacturer	varchar(50)	NOT NULL		Gyártó	
Modell	varchar(50)	NOT NULL		Model	
DailyPrice	int	NOT NULL		Jármű napi ára	DailyPrice > 0
ChassisNumber	varchar(25)	NOT NULL		Alvázszám	CK, UIX
PlateNumber	varchar(9)	NOT NULL		Rendszám	CK, UIX
Mileage	int	NULL		Km óra állása	
PurchaseDate	date	NULL		Vásárlás dátuma	
VehicleDetailID	int	NOT NULL		Idegen kulcs VehicleDetail táblára	FK
VehicleTypeID	int	NOT NULL		Idegen kulcs VehicleType táblára	FK
WorkPlaceID	int	NOT NULL		Idegen kulcs a WorkPlace táblára	FK

Táblaszintű megszorítások

Megszorítás neve	Megszorítás fajtája (trigger, constraint)	Értelmezés
Check_ChassisNumber	CONSTRAINT	LEN(ChassisNumber) = 17
Check_DailyPrice	CONSTRAINT	DailyPrice > 0
Check_PlateNumber	CONSTRAINT	A rendszámnak meg kell felelnie a regexre '%[A-Z]%[-]%[0-9]%'

Indexek

Index neve	Oszlop (ok)	Index típusa	Értelmezés
PK_Vehicle_VehicleID	VehicleID	Clustered (RS)	Elsődleges kulcs
UIX_Vehicle_ChassisNumber	ChassisNumber	Unique non-clustered	Ne lehessen két egyforma
UIX_Vehicle_PlateNumber	PlateNumber	Unique non-clustered	Ne lehessen két egyforma

Táblakapcsolatok

Idegen kulcs neve	Kapcsolat típusa	Delete és Update szabály
FK_Vehicle_VehicleDetailID_ VehicleDetail_VehicleDetailID	1 : 1	DELETE=NO ACTION UPDATE=NO ACTION
FK_Vehicle_VehicleTypeID_ VehicleType_VehicleTypeID	1 : 1	DELETE=NO ACTION UPDATE=NO ACTION
FK_Vehicle_WorkplaceID_ Workplace_WorkplaceID	1 : 1	DELETE=NO ACTION UPDATE=NO ACTION

Dbo.VehicleType

Mezőleírás

Oszlop neve	Adattípus	Kötelező	Default	Értelmezés	Ellenőrzés
VehicleTypeID	int	NOT NULL	(IDENTITY)	Járműtípus azonosító	PK
VehicleTypeName	varchar(50)			Járműtípus neve	

Indexek

Index neve	Oszlop (ok)	Index típusa	Értelmezés
PK_VehicleType_VehicleTypeID	VehicleTypeID	Clustered (RS)	Elsődleges kulcs

Dbo.VehicleDetail

A járművek részletes tulajdonságait tartalmazó tábla.

Mezőleírás

Oszlop neve	Adattípus	Kötelező	Default	Értelmezés	Ellenőrzés
VehicleDetailID	int	NOT NULL	(IDENTITY)	Jármű részletek azonosító	PK
PassengerCapacity	tinyint	NULL		Ülések száma	
Door	tinyint	NULL		Ajtók száma	
Color	varchar(20)	NULL		Szín	
Transmission	varchar(20)	NULL		Váltó típusa	
Engine	varchar(100)	NULL		Jármű motorjának adatai	
FuelType	varchar(20)	NULL		Üzemanyag típusa	
HorsePower	smallint	NULL		Lóerő	

Indexek

Index neve	Oszlop (ok)	Index típusa	Értelmezés
PK_VehicleDetail_VehicleDetailID	VehicleDetailID	Clustered (RS)	Elsődleges kulcs

Dbo.VehicleImage

A járművekről készült képeket tartalmazó tábla.

Mezőleírás

Oszlop neve	Adattípus	Kötelező	Default	Értelmezés	Ellenőrzés
ImageID	int	NOT NULL	(IDENTITY)	Jármű képének azonosítója	PK
ImageName	varchar(25)	NOT NULL		A kép neve	
Picture	varbinary(max)	NOT NULL		Kép	
rowguid	uniqueidentifier	NOT NULL, UNIQUE		egyedi azonosító	

Indexek

Index neve	Oszlop (ok)	Index típusa	Értelmezés
PK_Image_ ImageID	ImageID	Clustered (RS)	Elsődleges kulcs

Dbo.ShiftVehicleImage

Kapcsolattábla a járművek és a hozzájuk tartozó képekről.

Mezőleírás

Oszlop neve	Adattípus	Kötelező	Default	Értelmezés	Ellenőrzés
VehicleID	int	NOT NULL		Jármű azonosítója	FK
ImageID	int	NOT NULL		Kép azonosítója	FK

Táblakapcsolatok

Idegen kulcs neve	Kapcsolat típusa	Delete és Update szabály
FK_ShiftVehicleImage_ImageID_VehicleImage_ImageID	1 : 1	DELETE=CASCADE UPDATE=CASCADE
FK_ShiftVehicleImage_VehicleID_Vehicle_VehicleID	1 : N	DELETE=NO ACTION UPDATE=NO ACTION

Dbo.CarFeature

A járművek felszereltségét és opcionális kiegészítőket tartalmazó tábla.

Mezőleírás

Oszlop neve	Adattípus	Kötelező	Default	Értelmezés	Ellenőrzés
FeatureID	int	NOT NULL	(IDENTITY)	Jellemző azonosítója	PK
FeatureName	varchar(50)	NOT NULL		Jellemző megnevezése	
FeatureIsOptional	bit	NOT NULL	0	1 = választható	

Indexek

Index neve	Oszlop (ok)	Index típusa	Értelmezés
PK_CarFeature_FeatureID	FeatureID	Clustered (RS)	Elsődleges kulcs

Dbo.Reservation

A foglalások nyilvántartására szolgáló tábla.

Mezőleírás

Oszlop neve	Adattípus	Kötelező	Default	Értelmezés	Ellenőrzés
ReservationID	int	NOT NULL	(IDENTITY)	Foglalás azonosítója	PK
ReservationDate	date	NOT NULL		Foglalás dátuma	
PlannedRentDate	date	NOT NULL		Tervezett kölcsönzési első napja	
PlannedRentEndDate	date	NOT NULL		Tervezett kölcsönzés utolsó napja	
PersonID	int	NOT NULL		Idegenkulcs a Person táblára	FK
EmployeeID	int	NULL		Idegenkulcs az Employee táblára	FK
VehicleID	int	NOT NULL		Idegenkulcs a Vehicle táblára	FK
ReservationPrice	int	NOT NULL	8000	Foglalás költsége	FK
IsCanceled	bit	NOT NULL	0	1 = visszamondott foglalás	
CanceledDate	date	NULL		Visszamondás dátuma	

Indexek

Index neve	Oszlop (ok)	Index típusa	Értelmezés
PK_Reservation_ReservationID	ReservationID	Clustered (RS)	Elsődleges kulcs

Táblakapcsolatok

Idegen kulcs neve	Kapcsolat típusa	Delete és Update szabály
FK_Reservation_EmployeeID_Employee_EmployeeID	1 : N	DELETE=NO ACTION UPDATE=NO ACTION
FK_Reservation_VehicleID_Vehicle_VehicleID	1 : N	DELETE=NO ACTION UPDATE=NO ACTION
FK_Reservation_PersonID_Person_PersonID	1 : N	DELETE=NO ACTION UPDATE=NO ACTION

Dbo.Rental

A kölcsönzések nyilvántartására szolgáló tábla.

Mezőleírás

Oszlop neve	Adattípus	Kötelező	Default	Értelmezés	Ellenőrzés
RentalID	int	NOT NULL	(IDENTITY)	Kölcsönzés azonosítója	PK
RentalDate	date	NOT NULL		Kölcsönzés első napja	
RentalEndDate	date	NOT NULL		Kölcsönzés utolsó napja	
PersonID	int	NOT NULL		Idegenkulcs a Person táblára	FK
EmployeeID	int	NOT NULL		Idegenkulcs az Employee táblára	FK
VehicleID	int	NOT NULL		Idegenkulcs a Vehicle táblára	FK
ReservationID	int	NULL		Idegenkulcs a foglalás táblára (ha volt)	FK
TotalPrice	money	NULL		A kölcsönzés ára	

Indexek

Index neve	Oszlop (ok)	Index típusa	Értelmezés
PK_Rental_RentalID	RentalID	Clustered (RS)	Elsődleges kulcs

Táblakapcsolatok

Idegen kulcs neve	Kapcsolat típusa	Delete és Update szabály
FK_Rental_PersonID_Person_PersonID	1 : N	DELETE=NO ACTION UPDATE=NO ACTION
FK_Rental_EmployeeID_Employee_EmployeeID	1 : N	DELETE=NO ACTION UPDATE=NO ACTION
FK_Rental_VehicleID_Vehicle_VehicleID	1 : N	DELETE=NO ACTION UPDATE=NO ACTION
FK_Rental_ReservationID_Reservation_ReservationID	1 : N	DELETE=NO ACTION UPDATE=NO ACTION

Triggerek

Trigger neve	Trigger típusa	Értelmezés
trg_AI_CalcTotalPrice	DML (INSERT)	A Reservation táblából vagy közvetlenül új sor beszúrásakor a TotalPrice mező értékét kiszámolja

Trigger T-SQL kódja

```
CREATE OR ALTER TRIGGER dbo.trg_AI_CalcTotalPrice
 ON dbo.Rental
 AFTER INSERT
AS
BEGIN
 SET NOCOUNT ON
     DECLARE @DailyPrice money
     DECLARE @Day int
     SELECT @DailyPrice = DailyPrice FROM dbo.Vehicle V INNER JOIN inserted I ON
 V.VehicleID = I.VehicleID WHERE V.VehicleID = I.VehicleID
     SELECT @Day = DATEDIFF(d, inserted.RentalDate, inserted.RentalEndDate) FROM
inserted
     UPDATE Rental
         SET TotalPrice = @DailyPrice * @Day
         FROM Rental R
         INNER JOIN inserted I ON R.VehicleID = I.VehicleID
         WHERE R.VehicleID = I.VehicleID AND R.RentalID = I.RentalID
END
```

Dbo.DictCounty

Az irányítószámokat és városok neveit tartalmazó szótár tábla.

Mezőleírás

Oszlop neve	Adattípus	Kötelező	Default	Értelmezés	Ellenőrzés
PostalCodeID	int	NOT NULL		Egyedi azonosító	PK
PostalCode	nvarchar(4)	NULL		Irányítószám	
Studens	nvarchar(2)	NULL			
City	nvarchar(60)	NULL		Város	
CountyID	smallint	NULL		Megye azonosító	
GPS	geography	NULL		GPS polygon	

Indexek

Index neve	Oszlop (ok)	Index típusa	Értelmezés
PK_DictPostalCode_PostalCodeID	PostalCodeID	Clustered (RS)	Elsődleges kulcs

Dbo.Workplace

A telephelyek nyilvántartására szolgáló tábla.

Mezőleírás

Oszlop neve	Adattípus	Kötelező	Default	Értelmezés	Ellenőrzés
WorkPlaceID	int	NOT NULL	(IDENTITY)	Telephely azonosító	PK
WpAddress	varchar(150)	NOT NULL		Telephely címe	
WpPostalCode	varchar(8)	NOT NULL		Telephely irányítószám	
WpCity	varchar(50)	NOT NULL		Telephely városa	
WpEmail	varchar(50)	NOT NULL		Telephely e-mail címe	
WpPhone	varchar(50)	NOT NULL		Telephely telefonszáma	

Táblaszintű megszorítások

Megszorítás neve	Megszorítás fajtája (trigger, constraint)	Értelmezés
Check_EmailWp	CONSTRAINT	A WpEmail mezőnek meg kell felelnie a regexre: '%[A-Z0-9][@][A-Z0-9]%[.][A-Z0-9]%"

Indexek

Index neve	Oszlop (ok)	Index típusa	Értelmezés
PK_Workplace_WorkplaceID	WorkplaceID	Clustered (RS)	Elsődleges kulcs

Nézetek

dbo.PerformedLettingPerEmployee

Nézet leírása

Ebben a nézetben a dolgozók által sikeresen elvégzett kölcsönzési feladatok jellennek meg.

Nézet T-SQL kódja

```
CREATE OR ALTER VIEW [dbo].[PerformedLettingPerEmployee]
AS
 SELECT P.FullName, W.WpCity, E.JobTitle, E.HireDate, E.Salary,  COUNT(1) AS
'PerformedLetting'
 FROM Rental R
     INNER JOIN Employee E ON R.EmployeeID = E.EmployeeID
     INNER JOIN Workplace W ON W.WorkplaceID = E.WorkplaceID
     INNER JOIN Person P ON P.PersonID = E.PersonID
 GROUP BY P.FullName, W.WpCity, E.JobTitle, E.HireDate, E.Salary
GO
```

dbo.ReturningRenter

Nézet leírása

Ebben a nézetben azok visszatérő kölcsönzők jelennek meg, akik több, mint 8 alkalommal kölcsönöztek már autót. Felhasználható személyes kedvezmények kiküldésére.

Nézet T-SQL kódja

```
CREATE OR ALTER VIEW [dbo].[ReturningRenter]
AS
       SELECT P.FullName, P.PostalCode, P.City, P.Address, P.Email, COUNT(1) AS
       'RentalCount'
 FROM Person P
     INNER JOIN Rental R ON P.PersonID = R.PersonID
 GROUP BY P.FullName, P.PostalCode, P.City, P.Address, P.Email
 HAVING COUNT(1) > 8
GO
```

pbi.RentalForCompanies

Nézet leírása

A nézet kigyűjti azokat a kölcsönzéseket, amelyek cégek által történtek, megadja, hogy adott cég mekkora összegben, hány alkalommal kölcsönzött. A nézet azon felhasználók számára készült, akiknek SELECT jogosultsága van a pbi sémára, így Excel-ből vagy Power BI-ból tudnak kimutatásokat készíteni.

Nézet T-SQL kódja

```
CREATE OR ALTER   VIEW [pbi].[RentalForCompanies]
AS
       SELECT P.FullNameI, P.PostalCodeI, P.CityI, P.AddressI, COUNT(1) AS 'RentalNo',
       SUM(R.TotalPrice) AS 'TotalPrice'
 FROM Person P
 INNER JOIN Rental R ON P.PersonID = R.PersonID
 WHERE P.FullNameI IS NOT NULL
 GROUP BY P.FullNameI, P.PostalCodeI, P.CityI, P.AddressI
GO
```

pbi.QuarterlyRevenue

Nézet leírása

A nézet negyedéves pénzügyi adatok kigyűjtésére szolgál telephelyenként. A nézet azon felhasználók számára készült, akiknek SELECT jogosultsága van a pbi sémára, így Excel-ből vagy Power BI-ból tudnak kimutatásokat készíteni.

Nézet T-SQL kódja

```
CREATE OR ALTER VIEW [pbi].[QuarterlyRevenue]
AS
       SELECT YEAR(RentalDate) AS 'Year', DATEPART(Q, RentalDate) AS 'Quarter',
       W.WpCity, SUM(TotalPrice) AS 'TotalPrice' FROM Rental R
     INNER JOIN Employee E ON R.EmployeeID = E.EmployeeID
     INNER JOIN Workplace W ON W.WorkplaceID = E.WorkplaceID
 GROUP BY YEAR(RentalDate), DATEPART(Q, RentalDate), W.WpCity
GO
```

Függvények

dbo.GetTotalRentPricePerCar

A függvény funkcionalitásának leírása

A skalár függvény bemeneti paraméternek egy jármű azonosítót vár, amely alapján visszaadja az adott jármű mekkora bevételt termelt ki. A függvény felhasználható az adatbázishoz fejlesztett alkalmazás számára, ahol a felületen összegző információkat jelenít meg egyes járművekről.

Függvény paraméterei és visszaadott értéke

Paraméter neve	Adattípus	Típus	Alapértelmezés
@VehicleID	int	Input	Nincs
@Price	int	ReturnValue	
Return érték	int	ReturnValue	-1 = Nem adott meg azonosítót

A függvény T-SQL kódja

```
CREATE OR ALTER FUNCTION GetTotalRentPricePerCar(@VehicleID int)
    RETURNS money
AS
BEGIN
    IF @VehicleID IS NULL
        RETURN -1
    DECLARE @Price as money
    SELECT @Price = SUM(TotalPrice) FROM Rental R
    WHERE VehicleID = @VehicleID

    IF (@Price IS NULL)
        SET @Price = 0

    RETURN @Price
    END
```

dbo.NoOfRentalDays

A függvény funkcionalitásának leírása

A skalár függvény bemeneti paraméternek egy jármű azonosítót vár, amely alapján visszaadja, hogy az adott jármű összesen hány napot töltött kölcsönzésben. A függvény felhasználható az adatbázishoz fejlesztett alkalmazás számára, ahol a felületen összegző információkat jelenít meg egyes járművekről.

Függvény paraméterei és visszaadott értéke

Paraméter neve	Adattípus	Típus	Alapértelmezés
@VehicleID	int	Input	Nincs
@day	int	ReturnValue	
Return érték	int	ReturnValue	-1 = Nem adott meg azonosítót

A függvény T-SQL kódja

```
CREATE OR ALTER FUNCTION NoOfRentalDays(@VehicleID int = NULL)
    RETURNS int
AS
BEGIN
    IF @VehicleID IS NULL
        RETURN -1
    DECLARE @day int
    SELECT @day = SUM(DATEDIFF(d, RentalDate, RentalEndDate))
    FROM Rental WHERE VehicleID = @VehicleID

    IF (@day IS NULL)
        SET @day = 0

    RETURN @day
    END
```

dbo.VerifyHunPhoneNumber

A függvény funkcionalitásának leírása

A skalár függvény a bemeneti paraméterben megadott mobil hívószám helyességét ellenőrzi. Kezeli a NULL értéket, ellenőrzi a hosszát, számokon kívül tartalmaz-e más karaktert, illetve azt, hogy magyar szolgáltatónál található-e az adott szám. Továbbá ellenőrzi, hogy magyar országhívóval kezdődik-e a szám. Ez kliensoldalról garantálható lehet, ha a hívószám megadásakor országhívó alapértelmezett értéke +36 a továbbiak pedig legördülő menüből kiválaszthatóak.

Függvény paraméterei és visszaadott értéke

Paraméter neve	Adattípus	Típus	Alapértelmezés
@PhoneNumber	varchar(15)	Input	Nincs
Return érték	bit	ReturnValue	0 = A megadott hívószám érvényes -1 = Nem adott meg hívószámot -2 = A hívószám nem magyar hívószám -3 = A hívószám nem megfelelő hosszúságú -4 = A hívószám nem csak számokat tartalmaz -5 = A hívószám nem mobil hívószám

A függvény T-SQL kódja

```
CREATE OR ALTER FUNCTION VerifyHunPhoneNumber(@PhoneNumber varchar(15) = NULL)
    RETURNS int
AS
BEGIN
    IF @PhoneNumber Is NULL
        RETURN -1
    IF (LEFT(@PhoneNumber, 3) NOT LIKE '+36' OR LEFT(@PhoneNumber, 2) NOT LIKE '06')
        SET @PhoneNumber = REPLACE(@PhoneNumber, '+36', '06')
    ELSE
        RETURN -2

    IF(LEN(@PhoneNumber) <> 11)
        RETURN -3
    ELSE IF TRY_PARSE(RIGHT(@PhoneNumber, LEN(@PhoneNumber)-2) AS int) IS NULL
        RETURN -4
    ELSE IF(SELECT 1 WHERE SUBSTRING(@PhoneNumber, 3, 2) NOT IN ('20', '30', '31', '50',
'70')) = 1
        RETURN -5

    RETURN 0
    END
```

Tárolt eljárások

dbo.procCheckVIN

A tárolt eljárás funkcionalitásának leírása

A tárolt eljárás megvizsgálja, hogy a paraméterként megadott 17 karakteres alvázszám az észak-amerikai szabványnak megfelel-e. A szabvány szerint az alvázszám 9. karaktere egy ellenőrzőszám. Az alvázszámban minden karakternek és annak pozíciójának is különböző értéke van, ennek szorzatát kell karakterenként összeadni és moduláris osztást végezni rajta 11-gyel. A kapott szám lesz az ellenőrző szám. Ha az ellenőrzőszámmal módosított alvázszám megegyezik a paraméternek megadott alvázszámmal, akkor az érvényes.
Az ellenőrző szám számításának leírása:
https://en.wikibooks.org/wiki/Vehicle_Identification_Numbers_(VIN_codes)/Check_digit

Tárolt eljárás paraméterei és visszaadott értéke

Paraméter neve	Adattípus	Típus	Alapértelmezés és egyéb értékek
@VIN	varchar(17)	Input	Nincs
Return érték	int	ReturnValue	0 = A megadott alvázszám érvényes -1 = Nem adott meg bemeneti paramétert -2 = Karakterszám nem egyenlő 17-tel -3 = A bemeneti paraméterben érvénytelen karakter szerepel -4 = Az alvázszám érvénytelen az ÉA-i szabvány szerint

A tárolt eljárás T-SQL kódja

```
CREATE OR ALTER PROCEDURE rapp.procCheckVIN (@VIN AS varchar(17) = NULL)
AS
    BEGIN
        SET NOCOUNT ON
        IF @VIN IS NULL
            RETURN -1
        IF(LEN(@VIN) <> 17)
            RETURN -2

        DECLARE @Transliteration table(
            ID int IDENTITY(1,1) not null,
            TKey char,
            TValue smallint
        )
        INSERT INTO @Transliteration VALUES('a', 1)
        INSERT INTO @Transliteration VALUES('b', 2)
        INSERT INTO @Transliteration VALUES('c', 3)
        INSERT INTO @Transliteration VALUES('d', 4)
        INSERT INTO @Transliteration VALUES('e', 5)
        INSERT INTO @Transliteration VALUES('f', 6)
        INSERT INTO @Transliteration VALUES('g', 7)
        INSERT INTO @Transliteration VALUES('h', 8)
        INSERT INTO @Transliteration VALUES('j', 1)
        INSERT INTO @Transliteration VALUES('k', 2)
        INSERT INTO @Transliteration VALUES('l', 3)
        INSERT INTO @Transliteration VALUES('m', 4)
        INSERT INTO @Transliteration VALUES('n', 5)
        INSERT INTO @Transliteration VALUES('p', 7)
        INSERT INTO @Transliteration VALUES('r', 9)
        INSERT INTO @Transliteration VALUES('s', 2)
        INSERT INTO @Transliteration VALUES('t', 3)
        INSERT INTO @Transliteration VALUES('u', 4)
        INSERT INTO @Transliteration VALUES('v', 5)
        INSERT INTO @Transliteration VALUES('w', 6)
        INSERT INTO @Transliteration VALUES('x', 7)
        INSERT INTO @Transliteration VALUES('y', 8)
        INSERT INTO @Transliteration VALUES('z', 9)

        DECLARE @Weight table (
            Position int IDENTITY(1,1) NOT NULL,
            PositionWeight smallint
        )
        INSERT INTO @Weight VALUES(8)
        INSERT INTO @Weight VALUES(7)
        INSERT INTO @Weight VALUES(6)
        INSERT INTO @Weight VALUES(5)
        INSERT INTO @Weight VALUES(4)
        INSERT INTO @Weight VALUES(3)
        INSERT INTO @Weight VALUES(2)
        INSERT INTO @Weight VALUES(10)
        INSERT INTO @Weight VALUES(0)
        INSERT INTO @Weight VALUES(9)
        INSERT INTO @Weight VALUES(8)
        INSERT INTO @Weight VALUES(7)
        INSERT INTO @Weight VALUES(6)
        INSERT INTO @Weight VALUES(5)
        INSERT INTO @Weight VALUES(4)
        INSERT INTO @Weight VALUES(3)
        INSERT INTO @Weight VALUES(2)

        DECLARE @Cnt tinyint = 1
        DECLARE @CheckSUM smallint = 0
```

```
    DECLARE @CheckDigit tinyint = 0
    SET @VIN = LOWER(@VIN)
    DECLARE @ModifiedVIN AS varchar(17) = @VIN

    WHILE @Cnt <= 17
        BEGIN
            IF(SUBSTRING(@VIN, @Cnt, 1) IN ('i', 'o', 'q'))
            BEGIN
                RETURN -3
            END
            SET @CheckSUM +=
                CASE
                    WHEN TRY_PARSE(SUBSTRING(@VIN, @Cnt, 1) AS int) IS NOT NULL
                        THEN PARSE(SUBSTRING(@VIN, @Cnt, 1) AS int)
                        * (SELECT PositionWeight FROM @Weight WHERE Position = @Cnt)
                        ELSE
                            (SELECT TValue FROM @Transliteration WHERE TKey LIKE
                            SUBSTRING(@VIN, @Cnt, 1))
                            * (SELECT PositionWeight FROM @Weight WHERE Position = @Cnt)
                END
            SET @Cnt+=1
        END
    SET @CheckDigit = @CheckSUM % 11
    IF(@CheckDigit = 10)
        BEGIN
            SET @ModifiedVIN = STUFF(@ModifiedVIN, 9, 1, 'x')
        END
    ELSE
        BEGIN
            SET @ModifiedVIN = STUFF(@ModifiedVIN, 9, 1, @CheckDigit)
        END

    IF (@ModifiedVIN LIKE @VIN)
        BEGIN
            RETURN 0
        END
    ELSE
        BEGIN
            RETURN -4
        END
END
```

dbo.procDeletePersonalData

A tárolt eljárás funkcionalitásának leírása

A tárolt eljárás a Person táblában található személyes adatok értékét NULL-ra állítja azoknál a felhasználóknál, akiknek a User táblában az IsActive mező értéke 0 (tehát törölt) és ez a módosítás több, mint 1826 napja történt (5 éve). A tárolt eljárás job-bal akár minden este lefuttatható.

Tárolt eljárás paraméterei és visszaadott értéke

Paraméter neve	Adattípus	Típus	Alapértelmezés és egyéb értékek
Return érték	int	ReturnValue	0 = a módosításokat hiba nélkül elvégezte

A tárolt eljárás T-SQL kódja

```
CREATE OR ALTER PROC dbo.procDeletePersonalData
AS
BEGIN
    SET NOCOUNT ON
    UPDATE Person
        SET FirstName = NULL,
            LastName = NULL,
            FullName = NULL,
            Address = NULL,
            Phone = NULL,
            Email = NULL,
            BirthDate = NULL
    FROM Person P INNER JOIN [User] U ON P.PersonID = U.PersonID
    WHERE IsActive = 0 AND DATEDIFF(DAY, U.ModifiedDate, SYSDATETIME()) > 1826
    RETURN 0
END
```

emp.procAddNewReservation

A tárolt eljárás funkcionalitásának leírása

A tárolt eljárás segítségével a dolgozók új foglalást adhatnak hozzá a Reservation táblához anélkül, hogy közvetlenül INSERT utasítást adnának ki arra.

Tárolt eljárás paraméterei és visszaadott értéke

Paraméter neve	Adattípus	Típus	Alapértelmezés és egyéb értékek
@PlannedRentDate	date	INPUT	NULL
@PlannedRentEndDate	date	INPUT	NULL
@PersonID	int	INPUT	NULL
@EmployeeID	int	INPUT	NULL
@VehicleID	int	INPUT	NULL
Return érték	int	ReturnValue	0 = A foglalás rögzítése sikeres volt -1 = Hiányzó bemeneti paraméter -2 = A foglalás utolsó napja nem lehet korábban, mint az első nap -3 = A foglalás első napja nem lehet korábban a mai napnál -4 = Az adott időintervallumban már létezik rögzített foglalás az adott járműre

A tárolt eljárás T-SQL kódja

```
CREATE OR ALTER PROC emp.procAddNewReservation
    @PlannedRentDate date = NULL,
    @PlannedRentEndDate date = NULL,
    @PersonID int = NULL,
    @EmployeeID int = NULL,
    @VehicleID int = NULL
AS
BEGIN
    SET NOCOUNT ON
    IF(@PlannedRentDate IS NULL OR @PlannedRentEndDate IS NULL OR @PersonID IS NULL OR
    @EmployeeID IS NULL OR @VehicleID IS NULL)
        RETURN -1
    ELSE IF(@PlannedRentEndDate < @PlannedRentDate)
        RETURN -2
    ELSE IF(@PlannedRentDate < SYSDATETIME())
        RETURN -3
    ELSE IF (SELECT COUNT(1) FROM Reservation R
                WHERE (@PlannedRentDate >= R.PlannedRentDate AND @PlannedRentDate <=
                R.PlannedRentEndDate
                OR @PlannedRentEndDate >= R.PlannedRentDate AND @PlannedRentEndDate <=
                R.PlannedRentEndDate
                OR R.PlannedRentDate >= @PlannedRentDate AND R.PlannedRentDate <=
                @PlannedRentEndDate
                OR R.PlannedRentEndDate >= @PlannedRentDate AND R.PlannedRentEndDate <=
                @PlannedRentEndDate)
                AND R.VehicleID = @VehicleID
            )  > 0
        RETURN -4
    ELSE
        BEGIN
            INSERT INTO Reservation(ReservationDate, PlannedRentDate,PlannedRentEndDate,
            PersonID, EmployeeID,VehicleID)
            VALUES (SYSDATETIME(), @PlannedRentDate, @PlannedRentEndDate, @PersonID,
            @EmployeeID, @VehicleID)
            RETURN 0
        END
END
```

emp.procAddNewRental

A tárolt eljárás funkcionalitásának leírása

A tárolt eljárás segítségével a dolgozók új foglalást adhatnak hozzá a Reservation táblába anélkül, hogy közvetlenül INSERT utasítást adnának ki arra.

Tárolt eljárás paraméterei és visszaadott értéke

Paraméter neve	Adattípus	Típus	Alapértelmezés és egyéb értékek
@RentalDate	date	INPUT	NULL
@RentalEndDate	date	INPUT	NULL
@PersonID	int	INPUT	NULL
@EmployeeID	int	INPUT	NULL
@VehicleID	int	INPUT	NULL
Return érték	int	ReturnValue	0 = A kölcsönzés rögzítése sikeres volt -1 = Hiányzó bemeneti paraméter -2 = A kölcsönzés utolsó napja nem lehet korábban, mint az első napja -3 = Az adott időintervallumban már létezik rögzített kölcsönzés az adott járműre

A tárolt eljárás T-SQL kódja

```
CREATE OR ALTER PROC emp.procAddNewRental
    @RentalDate date = NULL,
    @RentalEndDate date = NULL,
    @PersonID int = NULL,
    @EmployeeID int = NULL,
    @VehicleID int = NULL
AS
BEGIN
    SET NOCOUNT ON
    IF(@RentalDate IS NULL OR @RentalEndDate IS NULL OR @PersonID IS NULL OR @EmployeeID
    IS NULL OR @VehicleID IS NULL)
        RETURN -1
    ELSE IF(@RentalEndDate < @RentalDate)
        RETURN -2
    ELSE IF (SELECT COUNT(1) FROM Rental R
                WHERE (@RentalDate >= R.RentalDate AND @RentalDate <= R.RentalEndDate
                OR @RentalEndDate >= R.RentalDate AND @RentalEndDate <= R.RentalEndDate
                OR R.RentalDate >= @RentalDate AND R.RentalDate <= @RentalEndDate
                OR R.RentalEndDate >= @RentalDate AND R.RentalEndDate <= @RentalEndDate)
                AND R.VehicleID = @VehicleID
            )  > 0
        RETURN -3

    INSERT INTO Rental(RentalDate, RentalEndDate, PersonID, EmployeeID,VehicleID)
    VALUES (@RentalDate, @RentalEndDate, @PersonID, @EmployeeID, @VehicleID)
END
```

emp.procInsertRentalFromReservation

A tárolt eljárás funkcionalitásának leírása

A tárolt eljárás segítségével az alkalmazottak egy megvalósult foglalást tudnak rögzíteni a Rental táblában a foglalás azonosítójának megadásával. A tárolt eljárás a Reservation táblában megkeresi a paraméterben megadott azonosítónak megfelelő sort és a szükséges adatokat beilleszti a Rental táblába.

Tárolt eljárás paraméterei és visszaadott értéke

Paraméter neve	Adattípus	Típus	Alapértelmezés és egyéb értékek
@ReservationID	int	INPUT	NULL
Return érték	int	ReturnValue	0 = A kölcsönzés rögzítése sikeres volt -1 = Hiányzó bemeneti paraméter -2 = Ilyen foglalás kóddal már szerepel kölcsönzés a Rental táblában

A tárolt eljárás T-SQL kódja

```
CREATE OR ALTER PROC emp.procInsertRentalFromReservation(@ReservationID int = NULL)
AS
BEGIN
    IF @ReservationID IS NULL
        RETURN -1
    ELSE IF @ReservationID IN (SELECT ReservationID FROM Rental)
        RETURN -2
    ELSE
    BEGIN
        INSERT INTO Rental(RentalDate, RentalEndDate, PersonID, EmployeeID, VehicleID,
        ReservationID)
        SELECT PlannedRentDate, PlannedRentEndDate, PersonID, EmployeeID, VehicleID,
        @ReservationID FROM Reservation
        WHERE ReservationID = @ReservationID
    RETURN 0
    END
END
```

dbo.procAllIndexRebuild

A tárolt eljárás funkcionalitásának leírása

A tárolt eljárás egy kurzorral végigmegy a sys.tables táblán és soronként elvégzi az indexek újraépítését az ALTER INDEX ALL ON *táblanév* REBUILD paranccsal.

Tárolt eljárás paraméterei és visszaadott értéke

Paraméter neve	Adattípus	Típus	Alapértelmezés és egyéb értékek
Return érték	int	ReturnValue	0

A tárolt eljárás T-SQL kódja

```
CREATE OR ALTER PROCEDURE [dbo].[procAllIndexRebuild]
AS
    DECLARE curTable CURSOR FOR SELECT name FROM sys.tables
    DECLARE @name varchar(200), @S varchar(1000)
    OPEN curTable
    FETCH NEXT FROM curTable INTO @name
    WHILE @@FETCH_STATUS = 0
        BEGIN
            SET @S = 'ALTER INDEX ALL ON ' +'['+ @name +']' + ' REBUILD'
            EXEC (@S)
            FETCH NEXT FROM curTable INTO @name
        END
    CLOSE curTable
    DEALLOCATE curTable
```

Jogosultsági rendszer

Az autókölcsönzői adatbázist a tervek alapján a következő kliens rendszerek érik el:

- **Windows alapú kliensprogram**, amelyet a cég alkalmazottjai használnak.
  - AdmCarRental – Teljeskörű adatbázis eléréssel rendelkezik.
  - EmpCarRental – Korlátozott hozzáférés az adatbázishoz.
- **Kölcsönzői webalkalmazás**. PHP alapú webes alkalmazás, ahol regisztráció után a felhasználók foglalásokat adhatnak le az autókra. Megkezdett foglalás esetén az autó 10 percre zárolásra kerül más felhasználók elől, ennyi ideje van az adott felhasználónak véglegesítenie a foglalását, amennyiben ezt nem teszi meg, újra elérhető lesz más felhasználók számára az autó. A felhasználók nyomon tudják követni a korábbi foglalásaikat és kölcsönzéseiket. Az autókat a különböző keresési paraméterek megadásával szűrve is meg tudják jeleníteni a felületen.
- **Power BI és Excel**. Az üzleti elemzéseket ezekben az alkalmazásokban készítik el az alkalmazottak.

Login objektumok

- AdmCarRental SQL Login (bulkadmin, dbcreator szerepkör, SQLAgentReaderRole szerepkör)
- EmpCarRental SQL Login (bulkadmin szerepkör)
- BACarRental

User objektumok

- AdmCarRental – db_owner adatbázis szerepkör
- EmpCarRental – EmpRole adatbázis szerepkör
- BACarRental – BARole adatbázis szerepkör

Database role és Application role objektumok

A CarRental adatbázis jelenleg két egyénileg létrehozott adatbázis szerepkört és egy alkalmazás szerepkört tartalmaz.

- **EmpRole** szerepkör felhasználói EXEC jogot kapnak az emp sémában található tárolt eljárásokra. Továbbá SELECT jogot kapnak a személyek, járművek, foglalások és a kölcsönzések adatait tartalmazó táblákra.
- **BARole** szerepkör tagjai SELECT jogot kapnak a pbi sémában található nézetekre
- **RentalWebApp** alkalmazás szerepkör, amelynek EXEC joga van a Rapp sémára és két tárolt eljárásra, amely az Emp sémában található

Telepítés

Person tábla feltöltése

1. **CreateDictTableFromHunSpatialDB** tárolt eljárás futtatása, amely létrehozza a DictCounty és DictPostalCode táblákat az adatbázisban a HunSpatialDB-ből másolva

Tárolt eljárás T-SQL kódja

```
CREATE OR ALTER PROC CreateDictTableFromHunSpatialDB
AS
    SET NOCOUNT ON
    SELECT *
        INTO CarRental.dbo.DictCounty
    FROM HunSpatial.dbo.County

    SELECT *
        INTO CarRental.dbo.DictPostalCode
    FROM HunSpatial.dbo.PostalCode

    ALTER TABLE DictPostalCode
    ADD CONSTRAINT PK_DictPostalCode_PostalCodeID PRIMARY KEY (PostalCodeID)

    ALTER TABLE DictCounty
    ADD CONSTRAINT PK_DictCounty_ID PRIMARY KEY (ID)
```

2. **FillPersonTableRandom** tárolt eljárás futtatása, amely a paraméterben megadott értéknek megfelelő számú személyt hoz létre a Person táblában véletlenszerűen generált nevekkel és címekkel. 1123 különböző családnév kerül betöltésre, amely Wikipédiáról származik. Az MTA Nyelvtudományi Intézet által publikált összes női és férfinév is betöltésre kerül külön ideiglenes táblákba, ami több, mint 4000 különböző keresztnevet jelent. A Posta weboldalán korábban fellelhető Excel-ből kinyertem az összes budapesti utcanevet, a tárolt eljárásban a különbözőek leválogatásra kerülnek, így nagyjából 7000 különböző utcanév kerül előállításra. A házszámok véletlenszámok generálásával állnak elő. Futtatás előtt a UNIQUE megszorítást el kell távolítani az Email mezőről, mivel ez a mező később kerül feltöltésre.

Tárolt eljárás T-SQL kódja

```
CREATE OR ALTER PROC FillPersonTableRandom
@Count int
AS
SET NOCOUNT ON
DROP TABLE IF EXISTS #LN
CREATE TABLE #LN (LastName varchar(30))
BULK INSERT #LN
    FROM '\\DESKTOP-SR0EERQ\C$\Training360 anyagok\_Adatbázis
üzemeltető\Vizsgaremek\Adatfájlok\Családnevek.txt'
    WITH(CODEPAGE = '65001')

UPDATE #LN SET LastName = LEFT(LastName,1) + LOWER(SUBSTRING(LastName, 2, 100))

--Női keresztnevek és gender beállítás
DROP TABLE IF EXISTS #FNF
CREATE TABLE #FNF (FirstName varchar(30))
BULK INSERT #FNF
    FROM '\\DESKTOP-SR0EERQ\C$\Training360 anyagok\_Adatbázis
üzemeltető\Vizsgaremek\Adatfájlok\osszesnoi.txt'
    WITH(CODEPAGE = '65001'
       , ROWTERMINATOR = '0x0A')

ALTER TABLE #FNF ADD Gender tinyint NOT NULL DEFAULT 1
```

```
-- Férfi keresztnevek és gender beállítás
CREATE TABLE #FNM (FirstName varchar(30))
BULK INSERT #FNM
    FROM '\\DESKTOP-SR0EERQ\C$\\Training360 anyagok\_Adatbázis
üzemeltető\Vizsgaremek\Adatfájlok\osszesffi.txt'
    WITH(CODEPAGE = '65001'
    , ROWTERMINATOR = '0x0A')

ALTER TABLE #FNM ADD Gender tinyint NOT NULL DEFAULT 2

SELECT *
INTO #AllFN
FROM
(SELECT SUBSTRING(FirstName, 1, LEN(FirstName)-1) AS FirstName, Gender FROM #FNF
    UNION
    SELECT SUBSTRING(FirstName, 1, LEN(FirstName)-1) AS FirstName, Gender FROM #FNM) X

-- személynév előállítása a családnevek és a keresztnevek alapján
DROP TABLE IF EXISTS #LF
SELECT ROW_NUMBER() OVER(ORDER BY (SELECT NULL)) PersonID, LastName, FirstName, Gender
    INTO #LF
    FROM
        (SELECT TOP (@Count) LastName, FirstName, Gender
        FROM #LN
        CROSS APPLY #AllFN
        ORDER BY NEWID()
        ) X

--Utcanevek betöltése
DROP TABLE IF EXISTS #Street
DROP TABLE IF EXISTS #TempStreet
CREATE TABLE #TempStreet (Street varchar(50))
BULK INSERT #TempStreet
    FROM '\\DESKTOP-SR0EERQ\C$\\Training360 anyagok\_Adatbázis
üzemeltető\Vizsgaremek\Adatfájlok\utcanevek.txt'
    WITH(CODEPAGE = '65001'
    , ROWTERMINATOR = '0x0A')

--Különböző utcanevek leválogatása
SELECT * INTO #Street
    FROM (SELECT DISTINCT Street FROM #TempStreet) X
DROP TABLE #TempStreet

DROP TABLE IF EXISTS #A
SELECT ROW_NUMBER() OVER(ORDER BY(SELECT NULL)) PersonID, PostalCode, City,
CONCAT(Street,FLOOR(((ABS(CHECKSUM(NEWID())) % 100001)
        + ((ABS(CHECKSUM(NEWID())) % 100001) * 00000.1) )/ 1100)) AS Street
    INTO #A
    FROM (SELECT TOP (@Count) PostalCode, City, Street
            FROM #Street
            CROSS APPLY DictPostalCode
            ORDER BY NEWID() ) X

--Person tábla feltöltése
SET IDENTITY_INSERT Person ON
INSERT Person (PersonID, LastName, FirstName, FullName, Gender, PostalCode, City, Address,
BirthDate)
SELECT #LF.PersonID, #LF.LastName, #LF.FirstName, #LF.LastName+' '+#LF.FirstName, #LF.Gender,
#A.PostalCode, #A.City, #A.Street,
    DATEADD(DAY, ABS(CHECKSUM(NEWID()) % 19001), '1950-01-01')
```

```
FROM #A
INNER JOIN #LF ON #LF.PersonID = #A.PersonID
SET IDENTITY_INSERT Person OFF
```

3. **CreateEmailForPerson** tárolt eljárás e-mail címeket állít elő a felhasználó nevéből, születési dátumából, illetve véletlenszerűen választott e-mail szolgáltatóból. Futtatás után a UNIQUE megszorítás visszaállítható az Email mezőre.

Tárolt eljárás T-SQL kódja

```
CREATE OR ALTER PROC CreateEmailForPerson
AS
    SET NOCOUNT ON
    ;WITH Random
    AS
    (
    SELECT
          p1.PersonID,
          p1.[FirstName],
          p1.[LastName],
          p1.BirthDate,
          CAST(RAND(CHECKSUM(NEWID()))*5 as int) randomemail
      FROM [dbo].Person p1
    )
    UPDATE Person
    SET Email = R.email
    FROM (
            SELECT PersonID,
            email=
            CASE
                when randomemail = 0 then
                    lower(left(FirstName,3)+[LastName])+(CAST(YEAR(BirthDate) as
                    varchar))+'@hotmail.com'
                when randomemail = 1 then
                    lower(left(FirstName,3)+[LastName])+(CAST(YEAR(BirthDate) as
                    varchar))+'@gmail.com'
                when randomemail = 2 then
                    lower(left(FirstName,3)+[LastName])+(CAST(YEAR(BirthDate) as
                    varchar))+'@freemail.hu'
                when randomemail = 3 then
                    lower(left(FirstName,3)+[LastName])+(CAST(YEAR(BirthDate) as
                    varchar))+'@outlook.com'
                else
                    lower(left(FirstName,3)+[LastName])+(CAST(YEAR(BirthDate) as
                    varchar))+'@yahoo.com'
            END
            FROM Random) R
    INNER JOIN
    Person P ON P.PersonID = R.PersonID

    Update Person SET Email = REPLACE(Email,'á','a')
    Update Person SET Email = REPLACE(Email,'é','e')
    Update Person SET Email = REPLACE(Email,'í','i')
    Update Person SET Email = REPLACE(Email,'ó','o')
    Update Person SET Email = REPLACE(Email,'ö','o')
    Update Person SET Email = REPLACE(Email,'ő','o')
    Update Person SET Email = REPLACE(Email,'ú','u')
    Update Person SET Email = REPLACE(Email,'ü','u')
    Update Person SET Email = REPLACE(Email,'ű','u')
```

4. **CreatePhoneForPerson** tárolt eljárás a felhasználók számára véletlenszerűen generált telefonszámokat hoz létre.

Tárolt eljárás T-SQL kódja

```
CREATE OR ALTER PROC CreatePhoneForPerson
AS
SET NOCOUNT ON
;WITH Random
AS
(
SELECT
      p1.PersonID,
      CAST(RAND(CHECKSUM(NEWID()))*3 as int) randomphone
  FROM [dbo].Person p1
)
UPDATE Person
SET Phone = R.Phone
FROM (
        SELECT PersonID,
        Phone =
        CASE
            when randomphone = 0 then
            '+3620'+SUBSTRING(CAST(ABS(CHECKSUM(NEWID())) as varchar),1,7)
            when randomphone = 1 then
            '+3630'+SUBSTRING(CAST(ABS(CHECKSUM(NEWID())) as varchar),1,7)
            else
            '+3670'+SUBSTRING(CAST(ABS(CHECKSUM(NEWID())) as varchar),1,7)
        END
        FROM Random) R
INNER JOIN
Person P ON P.PersonID = R.PersonID
```

Workplace tábla feltöltése

INSERT INTO segítségével 3 telephely kerül feltöltésre a táblába.

Táblafeltöltés T-SQL kódja

```
INSERT INTO Workplace VALUES ('Jókai u. 25', '6723', 'Szeged',
'carrentszeged@fakemail.com', '302223344')
INSERT INTO Workplace VALUES ('Keresztúri út 13', '1106', 'Budapest',
'carrentbp@fakemail.com', '302223355')
INSERT INTO Workplace VALUES ('Tihanyi Árpád út 59', '9023', 'Győr',
'carrentgyor@fakemail.com', '302223366')
```

Employee tábla feltöltése

INSERT INTO segítségével 7 dolgozó kerül feltöltésre a táblába.

Táblafeltöltés T-SQL kódja

```
INSERT INTO Employee VALUES('Ügyintéző', '2011-11-06', 16, 2, 350000)
INSERT INTO Employee VALUES('Ügyintéző', '2011-12-01', 98, 2, 350000)
INSERT INTO Employee VALUES('Ügyintéző', '2011-10-21', 139, 2, 350000)
INSERT INTO Employee VALUES('Ügyintéző', '2011-11-10', 883, 3, 320000)
INSERT INTO Employee VALUES('Ügyintéző', '2011-11-03', 1099, 3, 320000)
INSERT INTO Employee VALUES('Ügyintéző', '2011-10-10', 419, 1, 320000)
INSERT INTO Employee VALUES('Ügyintéző', '2011-10-28', 1243, 1, 320000)
```

User tábla feltöltése

1. **CreateUsersRandom** tárolt eljárás a személyek neveiből és véletlenszerű számokból felhasználó neveket generál és CRYPT_GEN_RANDOM függvény segítségével jelszavakat. Egyre állítja azon felhasználók EmpRole értékét, amelyek szerepelnek az Employee táblában.

Tárolt eljárás T-SQL kódja

```
CREATE OR ALTER PROC dbo.CreateUsersRandom
AS
SET NOCOUNT ON
INSERT INTO [User] (LoginName, PasswordHash, PersonID)
            SELECT LOWER(SUBSTRING(FirstName,1,3)+LastName)+
            CAST(FLOOR((RAND(CHECKSUM(NEWID()))*1000) as varchar)
            , CONVERT(varbinary(64), CRYPT_GEN_RANDOM(64))
            , PersonID
            FROM Person

    UPDATE [User]
    SET EmpRole = 1,
        UserRole = 0
    WHERE PersonID IN (SELECT PersonID FROM Employee)

    UPDATE [User] SET LoginName = REPLACE(LoginName,'á','a')
    UPDATE [User] SET LoginName = REPLACE(LoginName,'é','e')
    UPDATE [User] SET LoginName = REPLACE(LoginName,'í','i')
    UPDATE [User] SET LoginName = REPLACE(LoginName,'ó','o')
    UPDATE [User] SET LoginName = REPLACE(LoginName,'ö','o')
    UPDATE [User] SET LoginName = REPLACE(LoginName,'ő','o')
    UPDATE [User] SET LoginName = REPLACE(LoginName,'ú','u')
    UPDATE [User] SET LoginName = REPLACE(LoginName,'ü','u')
    UPDATE [User] SET LoginName = REPLACE(LoginName,'ű','u')
```

VehicleType tábla feltöltése

INSERT INTO segítségével 6 autótípus kerül feltöltésre a táblába.

Táblafeltöltés T-SQL kódja

```
    INSERT INTO VehicleType VALUES ('car')
    INSERT INTO VehicleType VALUES ('minibus')
    INSERT INTO VehicleType VALUES ('bus')
    INSERT INTO VehicleType VALUES ('suv')
    INSERT INTO VehicleType VALUES ('pickup')
```

VehicleDetail tábla feltöltése

Bulk Insert segítségével egy csv fájlból tölti fel a VehicleDetail tábla adatait.

Táblafeltöltés T-SQL kódja

```
    BULK INSERT VehicleDetail
        FROM '\\DESKTOP-SR0EERQ\C$\\Training360 anyagok\_Adatbázis
        üzemeltető\Vizsgaremek\Adatfájlok\VehicleDetail.csv'
        WITH(FORMAT = 'CSV'
        , FIRSTROW = 2
        , CODEPAGE = '65001'
        , FIELDTERMINATOR = ';'
        , ROWTERMINATOR = '0x0A')
```

Carfeature tábla feltöltése

Bulk Insert segítségével egy csv fájlból tölti fel a CarFeature tábla adatait.

Táblafeltöltés T-SQL kódja

```
BULK INSERT CarFeature
    FROM '\\DESKTOP-SR0EERQ\C$\\Training360 anyagok\_Adatbázis
    üzemeltető\Vizsgaremek\Adatfájlok\carfeature.csv'
    WITH(FORMAT = 'CSV'
    , FIRSTROW = 2
    , CODEPAGE = '65001'
    , FIELDTERMINATOR = ';'
    , ROWTERMINATOR = '\n')
```

ShiftVehicleFeature tábla feltöltése

FillShiftVehicleFeature tárolt eljárás egy INSERT INTO…SELECT segítségével véletlenszerűen feltölti a kapcsolattáblát.

Tárolt eljárás T-SQL kódja

```
CREATE OR ALTER PROC FillShiftVehicleFeature
AS
    INSERT INTO ShiftVehicleFeature (VehicleID, FeatureID)
    SELECT TOP 1000 VehicleID, FeatureID FROM Vehicle V
    CROSS JOIN CarFeature
    ORDER BY NEWID()
```

Vehicle tábla feltöltése

Bulk Insert segítségével egy csv fájlból feltölti a táblát a járművek adataival.

Táblafeltöltés T-SQL kódja

```
BULK INSERT Vehicle
    FROM '\\DESKTOP-SR0EERQ\C$\\Training360 anyagok\_Adatbázis
    üzemeltető\Vizsgaremek\Adatfájlok\Vehicle.csv'
    WITH(FORMAT = 'CSV'
    , FIRSTROW = 2
    , CODEPAGE = '65001'
    , FIELDTERMINATOR = ';'
    , ROWTERMINATOR = '0x0A')
```

Reservation tábla feltöltése

1. Ideiglenes táblába véletlenszerű kapcsolatokat készítek a személyek és a járművek között telephelyenként külön.

Adatgenerálás T-SQL kódja

Egy kapcsolat létrehozása a #T ideiglenes táblába, ahol a WorkplaceID = 1, tehát a foglalás a szegedi telephelyen található járműre történt.

```
CREATE TABLE #T (ResDate date, StartDate date, EndDate date, PersonID int, VehicleID int,
EmployeeID int)

SET NOCOUNT ON
DECLARE @ResDate as smalldatetime
DECLARE @StartDate as smalldatetime
DECLARE @EndDate as smalldatetime

SELECT @ResDate = DATEADD(DAY, FLOOR(RAND()*2555+1), '2013-01-01')
SELECT @StartDate = DATEADD(DAY, FLOOR(RAND()*12+1), @ResDate)
SELECT @EndDate = DATEADD(DAY, FLOOR(RAND()*8+1), @StartDate)

INSERT INTO #T
```

```
SELECT  @ResDate, @StartDate, @EndDate, P.PersonID, V.VehicleID, E.EmployeeID
FROM (SELECT TOP 1 Person.PersonID FROM Person ORDER BY NEWID()) P
CROSS APPLY
    (SELECT TOP 1 VehicleID, WorkplaceID FROM Vehicle WHERE WorkplaceID = 1 ORDER BY
NEWID()) V
CROSS APPLY
    (SELECT TOP 1 EmployeeID, WorkplaceID FROM Employee WHERE WorkplaceID = 1 ORDER BY
NEWID()) E
WHERE V.WorkplaceID = E.WorkplaceID
GO
```

2. Egy CTE segítségével törlöm azokat a sorokat a #T ideiglenes táblából, ahol átfedés található a foglalási dátumok között.

A CTE T-SQL kódja

```
;WITH DelDateOverlap
AS
(
SELECT T1.ResDate, T1.StartDate,T1.EndDate, T1.PersonID, T1.VehicleID, T1.EmployeeID FROM
#T T1
INNER JOIN #T T2 ON (T2.StartDate > T1.StartDate AND T2.StartDate < T1.EndDate
OR T2.StartDate > T1.StartDate AND T2.StartDate <T1.EndDate
OR T1.StartDate > T2.StartDate AND T1.StartDate < T2.StartDate
OR T1.EndDate > T2.StartDate AND T1.EndDate < T2.StartDate)
AND T1.VehicleID = T2.VehicleID
)
DELETE T FROM #T T INNER JOIN DelDateOverlap D ON T.PersonID = D.PersonID AND T.VehicleID
= D.VehicleID AND T.EmployeeID = D.EmployeeID AND T.StartDate = D.StartDate
GO
```

3. A generált adatokkal feltöltöm a Reservation táblát.

Táblafeltöltés T-SQL kódja

```
INSERT Reservation
SELECT ResDate, StartDate, EndDate, T.PersonID, T.EmployeeID, T.VehicleID, 8000, 0, NULL
FROM #T T
INNER JOIN Vehicle V ON V.VehicleID = T.VehicleID
ORDER BY ResDate
```

4. **UpdateCanceledReservationRandom** tárolt eljárás segítségével a foglalások 5%-át visszamondott állapotúvá alakítja
(Reservation tábla feltöltése.sql)

Tárolt eljárás T-SQL kódja

```
CREATE OR ALTER PROC UpdateCanceledReservationRandom
AS
    UPDATE Reservation
    SET IsCanceled = 1,
        CancelDate = DATEADD(DAY, ABS(CHECKSUM(NEWID()) % DATEDIFF(day, ReservationDate,
        PlannedRentDate)) + 1, ReservationDate)
    WHERE Reservation.ReservationID IN
    (
        SELECT TOP 5 PERCENT ReservationID FROM Reservation
        ORDER BY NEWID()
    )
```

Rental tábla feltöltése

FillRentalFromReservation tárolt eljárás a Reservation táblából az összes megvalósult foglalást elhelyezi a Rental táblában, a TotalPrice mezőt feltöltés közben kiszámolja a jármű napiárának és a kölcsönzött napok szorzatával

Tárolt eljárás T-SQL kódja

```
CREATE OR ALTER PROC FillRentalFromReservation
AS
    INSERT Rental
        SELECT PlannedRentDate, PlannedRentEndDate, PersonID, EmployeeID, R.VehicleID,
        ReservationID
        , DATEDIFF(d, R.PlannedRentDate, R.PlannedRentEndDate) * V.DailyPrice
    FROM Reservation R INNER JOIN Vehicle V ON R.VehicleID = V.VehicleID
    WHERE IsCanceled = 0
```

VehicleImage tábla feltöltése

INSERT..INTO segítségével néhány kép betöltése a VehicleImage táblába.

Táblafeltöltés T-SQL kód

```
INSERT INTO VehicleImage (ImageName, Picture, rowguid)
    SELECT 'Chevrolet Camaro',
    bulkcolumn, NEWID() FROM OPENROWSET(
        BULK 'C:\Training360 anyagok\_Adatbázis
        üzemeltető\Vizsgaremek\Adatfájlok\image\Chevrolet Camaro.jpeg', SINGLE_BLOB) AS x

INSERT INTO VehicleImage (ImageName, Picture, rowguid)
    SELECT '2009 Audi A4 Sedan 3.2B',
    bulkcolumn, NEWID() FROM OPENROWSET(
        BULK 'C:\Training360 anyagok\_Adatbázis
        üzemeltető\Vizsgaremek\Adatfájlok\image\2009 Audi A4 Sedan 3.2B.jpeg',
        SINGLE_BLOB) AS x

INSERT INTO VehicleImage (ImageName, Picture, rowguid)
    SELECT '2010 Ford Ranger XLT',
    bulkcolumn, NEWID() FROM OPENROWSET(
        BULK 'C:\Training360 anyagok\_Adatbázis
        üzemeltető\Vizsgaremek\Adatfájlok\image\2010 Ford Ranger XLT.jpg', SINGLE_BLOB)
        AS x
```

ShiftVehicleImage tábla feltöltése

Néhány kapcsolat létrehozása az adatbázisba betöltött képek és néhány autó között.

Táblafeltöltés T-SQL kód

```
INSERT INTO ShiftVehicleImage
    SELECT VehicleID, 2 FROM Vehicle WHERE Modell LIKE '2009 Audi A4 Sedan 3.2'

INSERT INTO ShiftVehicleImage
    SELECT VehicleID, 1 FROM Vehicle WHERE Modell LIKE '%Camaro%'

INSERT INTO ShiftVehicleImage
    SELECT VehicleID, 3 FROM Vehicle WHERE Modell LIKE '%Ford Ranger%'
```

Mentési stratégia

A CarRental adatbázis igényeit az egyre növekvő felhasználó szám, a növekvő géppark miatt az SQL Express nem szolgálná ki, mivel az egyes autók képeit is az adatbázisban tároljuk egy Filestream filegroup-ban, ezért érdemes Standard vagy Enterprise verziót alkalmazni, amelynek nagy előnye, hogy tartalmazza az SQL Server Agent szolgáltatást.
Az adatbázis Full recovery modellt használ, amelynek mentése job-ok segítségével valósul meg. Teljes mentés történik hetente minden szombat éjfélkor, 4 óránként differential mentés, továbbá fél óránként a logok mentése valósul meg.
Ahhoz, hogy a mentések és a bennük lévő adatok konzisztenciáját garantálni lehessen, érdemes a *Verify backup when finished* és *Perform cheksum before writing to media* opciókat bejelölni.
A *RESTORE VERIFYONLY* beállítás miatt annak a felhasználónak, akinek a nevében futnak a mentésért felelős job-ok, dbcreator szerver szerepkört kell biztosítani, különben az ellenőrzést nem tudja elvégezni jogosultság hiányában.
A mentések 14 napig tárolódnak a szerveren, utána felülíródnak. A mentések biztonsága tovább növelhető szalagos egység segítségével, ha minden heti mentést bizonyos ideig még megtartunk kazettán is.

Full Backup T-SQL kódja

```
BACKUP DATABASE [CarRental] TO  DISK = N'C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\CarRentalFullBackup' WITH NOFORMAT, RETAINDAYS = 14,
NOINIT,  NAME = N'CarRental-Full Database Backup', SKIP, NOREWIND, NOUNLOAD,  STATS = 10,
CHECKSUM
GO
declare @backupSetId as int
select @backupSetId = position from msdb..backupset where database_name=N'CarRental' and
backup_set_id=(select max(backup_set_id) from msdb..backupset where
database_name=N'CarRental' )
if @backupSetId is null begin raiserror(N'Verify failed. Backup information for database
''CarRental'' not found.', 16, 1) end
RESTORE VERIFYONLY FROM  DISK = N'C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\CarRentalFullBackup.bak' WITH  FILE = @backupSetId,
NOUNLOAD,  NOREWIND
GO
```

Differential Backup T-SQL kódja

```
BACKUP DATABASE [CarRental] TO  DISK = N'C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\CarRentalDiffBackup.bak' WITH  DIFFERENTIAL ,
RETAINDAYS = 14, NOFORMAT, NOINIT,  NAME = N'CarRental-Differential Database Backup', SKIP,
NOREWIND, NOUNLOAD,  STATS = 10, CHECKSUM
GO
declare @backupSetId as int
select @backupSetId = position from msdb..backupset where database_name=N'CarRental' and
backup_set_id=(select max(backup_set_id) from msdb..backupset where
database_name=N'CarRental' )
if @backupSetId is null begin raiserror(N'Verify failed. Backup information for database
''CarRental'' not found.', 16, 1) end
RESTORE VERIFYONLY FROM  DISK = N'C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\CarRentalDiffBackup.bak' WITH  FILE = @backupSetId,
NOUNLOAD,  NOREWIND
GO
```

Log Backup T-SQL kódja

```
BACKUP LOG [CarRental] TO  DISK = N'C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\CarRentalLogBackup.bak' WITH  NO_TRUNCATE ,
RETAINDAYS = 14, NOFORMAT, NOINIT,  NAME = N'CarRental-Log Database Backup', SKIP, NOREWIND,
NOUNLOAD,  NORECOVERY ,  STATS = 10, CHECKSUM
GO
declare @backupSetId as int
select @backupSetId = position from msdb..backupset where database_name=N'CarRental' and
backup_set_id=(select max(backup_set_id) from msdb..backupset where
database_name=N'CarRental' )
if @backupSetId is null begin raiserror(N'Verify failed. Backup information for database
''CarRental'' not found.', 16, 1) end
RESTORE VERIFYONLY FROM  DISK = N'C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\CarRentalLogBackup.bak' WITH  FILE = @backupSetId,
NOUNLOAD,  NOREWIND
GO
```

Rendszer adatbázisok mentése

A mentési stratégia akkor teljes, ha a rendszeradatbázisok (master, model, msdb) is mentésre kerülnek, legalább akkora gyakorisággal, mint a saját adatbázis teljes mentése. Ezek az adatbázisok rendszerkonfigurációkat és SQL Server Job információkat tárolnak, amelyek elengedhetetlenek egy teljes rendszer visszaállítás során.

Master adatbázis mentésének T-SQL kódja

```
BACKUP DATABASE [master] TO  DISK = N'C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\master.bak' WITH  RETAINDAYS = 14, NOFORMAT, NOINIT,
NAME = N'master-Full Database Backup', SKIP, NOREWIND, NOUNLOAD,  STATS = 10, CHECKSUM
GO
declare @backupSetId as int
select @backupSetId = position from msdb..backupset where database_name=N'master' and
backup_set_id=(select max(backup_set_id) from msdb..backupset where database_name=N'master' )
if @backupSetId is null begin raiserror(N'Verify failed. Backup information for database
''master'' not found.', 16, 1) end
RESTORE VERIFYONLY FROM  DISK = N'C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\master.bak' WITH  FILE = @backupSetId,  NOUNLOAD,
NOREWIND
GO
```

Model adatbázis mentésének T-SQL kódja

```
BACKUP DATABASE [model] TO  DISK = N'C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\model.bak' WITH RETAINDAYS = 14, NOFORMAT, NOINIT,
NAME = N'model-Full Database Backup', SKIP, NOREWIND, NOUNLOAD,  STATS = 10, CHECKSUM
GO
declare @backupSetId as int
select @backupSetId = position from msdb..backupset where database_name=N'model' and
backup_set_id=(select max(backup_set_id) from msdb..backupset where database_name=N'model' )
if @backupSetId is null begin raiserror(N'Verify failed. Backup information for database
''model'' not found.', 16, 1) end
RESTORE VERIFYONLY FROM  DISK = N'C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\model.bak' WITH  FILE = @backupSetId,  NOUNLOAD,
NOREWIND
GO
```

Msdb adatbázis mentésének T-SQL kódja

```
BACKUP DATABASE [msdb] TO  DISK = N'C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\msdb.bak' WITH  RETAINDAYS = 14, NOFORMAT, NOINIT,
NAME = N'msdb-Full Database Backup', SKIP, NOREWIND, NOUNLOAD,  STATS = 10, CHECKSUM
GO
declare @backupSetId as int
select @backupSetId = position from msdb..backupset where database_name=N'msdb' and
backup_set_id=(select max(backup_set_id) from msdb..backupset where database_name=N'msdb' )
if @backupSetId is null begin raiserror(N'Verify failed. Backup information for database
''msdb'' not found.', 16, 1) end
RESTORE VERIFYONLY FROM  DISK = N'C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\msdb.bak' WITH  FILE = @backupSetId,  NOUNLOAD,
NOREWIND
GO
```

Job-ok létrehozása és ütemezése

```
USE [msdb]
GO
DECLARE @jobId BINARY(16)
EXEC  msdb.dbo.sp_add_job @job_name=N'CarRentalFullBackup',
        @enabled=1,
        @notify_level_eventlog=0,
        @notify_level_email=2,
        @notify_level_page=2,
        @delete_level=0,
        @category_name=N'[Uncategorized (Local)]',
        @owner_login_name=N'DESKTOP-SR0EERQ\sysadmin', @job_id = @jobId OUTPUT
select @jobId
GO
EXEC msdb.dbo.sp_add_jobserver @job_name=N'CarRentalFullBackup', @server_name = N'DESKTOP-
SR0EERQ'
GO
USE [msdb]
GO
EXEC msdb.dbo.sp_add_jobstep @job_name=N'CarRentalFullBackup', @step_name=N'FullBackup',
        @step_id=1,
        @cmdexec_success_code=0,
        @on_success_action=1,
        @on_fail_action=2,
        @retry_attempts=0,
        @retry_interval=0,
        @os_run_priority=0, @subsystem=N'TSQL',
        @command=N'BACKUP DATABASE [CarRental] TO  DISK = N''C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\CarRentalFullBackup.bak'' WITH NOFORMAT, RETAINDAYS =
14, NOINIT,  NAME = N''CarRental-Full Database Backup'', SKIP, NOREWIND, NOUNLOAD,  STATS =
10, CHECKSUM
GO
declare @backupSetId as int
select @backupSetId = position from msdb..backupset where database_name=N''CarRental'' and
backup_set_id=(select max(backup_set_id) from msdb..backupset where
database_name=N''CarRental'' )
if @backupSetId is null begin raiserror(N''Verify failed. Backup information for database
''''CarRental'''' not found.'', 16, 1) end
RESTORE VERIFYONLY FROM  DISK = N''C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\CarRentalFullBackup.bak'' WITH  FILE = @backupSetId,
NOUNLOAD,  NOREWIND
GO
',
        @database_name=N'CarRental',
        @flags=0
```

```
GO
USE [msdb]
GO
EXEC msdb.dbo.sp_update_job @job_name=N'CarRentalFullBackup',
        @enabled=1,
        @start_step_id=1,
        @notify_level_eventlog=0,
        @notify_level_email=2,
        @notify_level_page=2,
        @delete_level=0,
        @description=N'',
        @category_name=N'[Uncategorized (Local)]',
        @owner_login_name=N'DESKTOP-SR0EERQ\sysadmin',
        @notify_email_operator_name=N'',
        @notify_page_operator_name=N''
GO
USE [msdb]
GO
DECLARE @schedule_id int
EXEC msdb.dbo.sp_add_jobschedule @job_name=N'CarRentalFullBackup', @name=N'W_CarRental_Full',
        @enabled=1,
        @freq_type=8,
        @freq_interval=64,
        @freq_subday_type=1,
        @freq_subday_interval=0,
        @freq_relative_interval=0,
        @freq_recurrence_factor=1,
        @active_start_date=20210813,
        @active_end_date=99991231,
        @active_start_time=0,
        @active_end_time=235959, @schedule_id = @schedule_id OUTPUT
select @schedule_id
GO

USE [msdb]
GO
DECLARE @jobId BINARY(16)
EXEC  msdb.dbo.sp_add_job @job_name=N'CarRentalDiffBackup',
        @enabled=1,
        @notify_level_eventlog=0,
        @notify_level_email=2,
        @notify_level_page=2,
        @delete_level=0,
        @category_name=N'[Uncategorized (Local)]',
        @owner_login_name=N'DESKTOP-SR0EERQ\sysadmin', @job_id = @jobId OUTPUT
select @jobId
GO
EXEC msdb.dbo.sp_add_jobserver @job_name=N'CarRentalDiffBackup', @server_name = N'DESKTOP-
SR0EERQ'
GO
USE [msdb]
GO
EXEC msdb.dbo.sp_add_jobstep @job_name=N'CarRentalDiffBackup', @step_name=N'DiffBackup',
        @step_id=1,
        @cmdexec_success_code=0,
        @on_success_action=1,
        @on_fail_action=2,
        @retry_attempts=0,
        @retry_interval=0,
        @os_run_priority=0, @subsystem=N'TSQL',
```

```
        @command=N'BACKUP DATABASE [CarRental] TO  DISK = N''C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\CarRentalDiffBackup.bak'' WITH  DIFFERENTIAL ,
RETAINDAYS = 14, NOFORMAT, NOINIT,  NAME = N''CarRental-Differential Database Backup'', SKIP,
NOREWIND, NOUNLOAD,  STATS = 10, CHECKSUM
GO
declare @backupSetId as int
select @backupSetId = position from msdb..backupset where database_name=N''CarRental'' and
backup_set_id=(select max(backup_set_id) from msdb..backupset where
database_name=N''CarRental'' )
if @backupSetId is null begin raiserror(N''Verify failed. Backup information for database
''''CarRental'''' not found.'', 16, 1) end
RESTORE VERIFYONLY FROM  DISK = N''C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\CarRentalDiffBackup.bak'' WITH  FILE = @backupSetId,
NOUNLOAD,  NOREWIND
GO

',
        @database_name=N'CarRental',
        @flags=0
GO
USE [msdb]
GO
EXEC msdb.dbo.sp_update_job @job_name=N'CarRentalDiffBackup',
        @enabled=1,
        @start_step_id=1,
        @notify_level_eventlog=0,
        @notify_level_email=2,
        @notify_level_page=2,
        @delete_level=0,
        @description=N'',
        @category_name=N'[Uncategorized (Local)]',
        @owner_login_name=N'DESKTOP-SR0EERQ\sysadmin',
        @notify_email_operator_name=N'',
        @notify_page_operator_name=N''
GO
USE [msdb]
GO
DECLARE @schedule_id int
EXEC msdb.dbo.sp_add_jobschedule @job_name=N'CarRentalDiffBackup',
@name=N'4H_CarRental_Diff',
        @enabled=1,
        @freq_type=4,
        @freq_interval=1,
        @freq_subday_type=8,
        @freq_subday_interval=4,
        @freq_relative_interval=0,
        @freq_recurrence_factor=1,
        @active_start_date=20210813,
        @active_end_date=99991231,
        @active_start_time=0,
        @active_end_time=235959, @schedule_id = @schedule_id OUTPUT
select @schedule_id
GO

USE [msdb]
GO
DECLARE @jobId BINARY(16)
EXEC  msdb.dbo.sp_add_job @job_name=N'CarRentalLogBackup',
        @enabled=1,
        @notify_level_eventlog=0,
```

```
        @notify_level_email=2,
        @notify_level_page=2,
        @delete_level=0,
        @category_name=N'[Uncategorized (Local)]',
        @owner_login_name=N'DESKTOP-SR0EERQ\sysadmin', @job_id = @jobId OUTPUT
select @jobId
GO
EXEC msdb.dbo.sp_add_jobserver @job_name=N'CarRentalLogBackup', @server_name = N'DESKTOP-
SR0EERQ'
GO
USE [msdb]
GO
EXEC msdb.dbo.sp_add_jobstep @job_name=N'CarRentalLogBackup', @step_name=N'LogBackup',
        @step_id=1,
        @cmdexec_success_code=0,
        @on_success_action=1,
        @on_fail_action=2,
        @retry_attempts=0,
        @retry_interval=0,
        @os_run_priority=0, @subsystem=N'TSQL',
        @command=N'BACKUP LOG [CarRental] TO  DISK = N''C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\CarRentalLogBackup.bak'' WITH  RETAINDAYS = 14,
NOFORMAT, NOINIT,  NAME = N''CarRental-Full Database Backup'', SKIP, NOREWIND, NOUNLOAD,
STATS = 10, CHECKSUM
GO
declare @backupSetId as int
select @backupSetId = position from msdb..backupset where database_name=N''CarRental'' and
backup_set_id=(select max(backup_set_id) from msdb..backupset where
database_name=N''CarRental'' )
if @backupSetId is null begin raiserror(N''Verify failed. Backup information for database
''''CarRental'''' not found.'', 16, 1) end
RESTORE VERIFYONLY FROM  DISK = N''C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\CarRentalLogBackup.bak'' WITH  FILE = @backupSetId,
NOUNLOAD,  NOREWIND
GO
',
        @database_name=N'CarRental',
        @flags=0
GO
USE [msdb]
GO
EXEC msdb.dbo.sp_update_job @job_name=N'CarRentalLogBackup',
        @enabled=1,
        @start_step_id=1,
        @notify_level_eventlog=0,
        @notify_level_email=2,
        @notify_level_page=2,
        @delete_level=0,
        @description=N'',
        @category_name=N'[Uncategorized (Local)]',
        @owner_login_name=N'DESKTOP-SR0EERQ\sysadmin',
        @notify_email_operator_name=N'',
        @notify_page_operator_name=N''
GO
USE [msdb]
GO
DECLARE @schedule_id int
EXEC msdb.dbo.sp_add_jobschedule @job_name=N'CarRentalLogBackup',
@name=N'30min_CarRental_Log',
        @enabled=1,
```

```
        @freq_type=4,
        @freq_interval=1,
        @freq_subday_type=4,
        @freq_subday_interval=30,
        @freq_relative_interval=0,
        @freq_recurrence_factor=1,
        @active_start_date=20210813,
        @active_end_date=99991231,
        @active_start_time=0,
        @active_end_time=235959, @schedule_id = @schedule_id OUTPUT
select @schedule_id
GO
```

Job kezelések átruházása más felhasználóra

Ehhez a felhasználó Login-ját ki kell vezetni az msdb adatbázisba és legalább SQLAgentUserRole-al kell rendelkeznie.

```
USE [msdb]
GO
EXEC msdb.dbo.sp_update_job @job_id=N'3f050e06-01a9-48e3-8b1c-c2c26acd93b5',
        @owner_login_name=N'AdmCarRental'
GO

USE [msdb]
GO
EXEC msdb.dbo.sp_update_job @job_id=N'f7459fb3-8fa0-451c-89e5-1c6d80499038',
        @owner_login_name=N'AdmCarRental'
GO

USE [msdb]
GO
EXEC msdb.dbo.sp_update_job @job_id=N'b69b8efb-695f-4373-8524-109546853102',
        @owner_login_name=N'AdmCarRental'
GO
```

Adatbázis visszaállításának menete

1. Az adatbázis Single User módba helyezése
2. Tail-log backup készítése
3. Az utolsó Full backup visszaállítása *WITH NORECOVERY* opcióval
4. Differential backup visszaállítás *WITH NORECOVERY* opcióval
5. Transaction Log backup visszaállítása *WITH NORECOVERY* opcióval
6. A korábban készített Tail-log backup visszaállítása *WITH NORECOVERY* opcióval
7. Az adatbázis helyreállítása *RESTORE DATABASE <dbname> WITH RECOVERY* parancs kiadásával
8. Az adatbázis Multi User módba helyezése

A visszaállítás T-SQL kódja

```
USE master
ALTER DATABASE CarRental SET SINGLE_USER WITH ROLLBACK IMMEDIATE

BACKUP LOG CarRental TO DISK = N'C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\Taillog.bak'
   WITH NORECOVERY, NO_TRUNCATE;

RESTORE DATABASE CarRental FROM DISK = N'C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\CarRentalFullBackup.bak'
   WITH NORECOVERY;

   RESTORE DATABASE CarRental FROM DISK = N'C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\CarRentalDiffBackup.bak'
   WITH NORECOVERY;

RESTORE LOG CarRental FROM DISK = N'C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\CarRentalLogBackup.bak'
   WITH NORECOVERY;

RESTORE LOG CarRental FROM DISK = N'C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\Backup\Taillog.bak'
   WITH NORECOVERY;

RESTORE DATABASE CarRental
WITH RECOVERY;

ALTER DATABASE CarRental SET MULTI_USER WITH ROLLBACK IMMEDIATE
```