Aflevering uge 5
- Videreudvikling af FanzyPublish -

Jeg startede med at slette min pris-kolonne så jeg kan lave den mere fleksibel med en pristabel.

```
ALTER TABLE [dbo].[Book]
DROP COLUMN Price;
```

Så oprettede jeg pristabellen. Den sætter automatisk Id.

```
IF OBJECT_ID('[dbo].[Price]', 'U') IS NOT NULL
DROP TABLE [dbo].[Price]
GO

CREATE TABLE [dbo].[Price](
    [Id] INT IDENTITY (1,1) NOT NULL PRIMARY KEY, -- Primary Key column
    [Price] DECIMAL(6,2) NOT NULL
);
GO
```

Så indsatte jeg forskellige priser i pristabellen.

```
INSERT INTO [dbo].[Price]
VALUES(9.99),(19.99),(29.99),(39.99),(49.99),(59.99),(69.99),(79.99),
(89.99),(99.99);
```

Så oprettede jeg bog-pris-relationstabellen, for at have muligheden for flere priser på den samme bog til forskellige tider. Hvis man ikke skriver en specifik dato ved insert i denne tabel skriver den automatisk dags dato. Jeg indførte et Id, da bog + pris Id´er jo kan optræde flere gange. Man kunne muligvis have brugt bogId + prisId + dato, men jeg synes det var mere clean sådan her.

```
IF OBJECT_ID('[dbo].[BookPriceRelation]', 'U') IS NOT NULL
DROP TABLE [dbo].[BookPriceRelation]
GO

CREATE TABLE [dbo].[BookPriceRelation](
    [Id] INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
    [BookId]INT NOT NULL,
    [PriceId] INT NOT NULL,
    [PriceDate] DATE NOT NULL,
    FOREIGN KEY (BookId) REFERENCES Book (Id),
    FOREIGN KEY (PriceId) REFERENCES Author (Id)
);
GO

ALTER TABLE [dbo].[BookPriceRelation] ADD CONSTRAINT
[DF_BookPriceRelation_PriceDate]
    DEFAULT (GETDATE()) FOR [PriceDate];
GO
```

Så indsatte jeg værdier i bog-pris-relationstabellen. Jeg afprøvede de 2 muligheder med og uden dato. Det var nødvendigt for at få en tidligere dato.

```
INSERT INTO [dbo].[BookPriceRelation]
    (BookId, PriceId)
VALUES
    (2,6),
    (3,4),
    (4,5);
GO

INSERT INTO [dbo].[BookPriceRelation]
    (BookId, PriceId, PriceDate)
VALUES
    (2,8,'2020-02-02'),
    (4,6,'2020-03-03'),
    (7,9,'2021-12-31');
GO
```

Opgave 3:

Så har jeg lavet en query på bøger med ISBN, Title, pris, gennemsnitspris og foregående pris.

```
SELECT
    ISBN,
    Title,
    p.Price AS 'CurrentPrice',
    CAST(AVG(p.Price) OVER(PARTITION BY b.Id) AS DECIMAL(6,2)) AS
AverageBookPrice,
    LAG(p.Price) OVER(PARTITION BY b.Id ORDER BY p.price) AS
PreviousPrice
FROM Book as b
LEFT JOIN BookPriceRelation as bp on bp.BookId = b.Id
LEFT JOIN Price as p on p.Id = bp.PriceId
ORDER BY ISBN;
```

Resultatet af ovenstående som CSV-fil.

```
ISBN,Title,CurrentPrice,AverageBookPrice,PreviousPrice
978-1-4842-3017-6,Pro C# 7,39.99,39.99,NULL
978-1-4842-6605-2,Beginning T-SQL,NULL,NULL,NULL
978-1-491-98765-0,C# 7.0,59.99,69.99,NULL
978-1-491-98765-0,C# 7.0,79.99,69.99,59.99
978-1-529-35541-3,The Institute,89.99,89.99,NULL
978-1-61729-627-1,Unit Testing,49.99,54.99,NULL
978-1-61729-627-1,Unit Testing,59.99,54.99,49.99
978-87-02-08254-8,Mange sære ting for,NULL,NULL,NULL
978-87-02-30420-6,Fremtidsspejl,NULL,NULL,NULL
978-87-400-451-30,Natrium Chlorid,NULL,NULL,NULL
978-87-400-6521-3,Blodtørst,NULL,NULL,NULL
```

Opgave 4:

Søgning på en bog med flere parametre i Where clausen og et LIKE.

```
SELECT Title, FirstName + ' ' + LastName AS Fullname, ISBN, Price,
PriceDate
FROM Book as b
JOIN AuthorBookRelation abr on abr.BookId = b.Id
JOIN Author a on a.Id = abr.AuthorId
JOIN BookPriceRelation bpr on bpr.BookId = b.Id
JOIN Price p on p.Id = bpr.PriceId
WHERE Price > 40 AND PriceDate = '2022-03-28' AND Title LIKE '%C%#%';
```

Resultatet af opgave 4 som CSV.

```
Title,Fullname,ISBN,Price,PriceDate
C# 7.0,Joseph Albahari,978-1-491-98765-0,59.99,2022-03-28
C# 7.0,Ben Albahari,978-1-491-98765-0,59.99,2022-03-28
```

Opgave 5:

Her har jeg lavet query med en subquery for at finde de bøger der er sat en prisseddel på.

```
SELECT Title AS 'Books with a price'
FROM Book b
WHERE EXISTS(
    SELECT PriceId FROM BookPriceRelation as bpr
    WHERE b.Id = bpr.BookId);
```

Resultatet af opgave 5 som CSV.

```
Books with a price
C# 7.0
Pro C# 7
Unit Testing
The Institute
```

Fremstillet af

Janus Mogensen