# SYNERGY 2015

SEATTLE, WA, USA

SYNERGY2015

SEATTLE, WA, USA

# Applied SQL

Presented by Eddy Kleinjan

Take advantage of SQL Server capabilities and use them in your DataFlex project.

# SQL


## SQL Server Options

# SQL Server Options

**SQL Server Versions**

- 2000
- 2005
- 2008
- 2008 R2
- 2012
- 2014

**SQL Server Editions (2014)**

- Express
- Web
- Standard
- Business Intelligence
- Enterprise
- LocalDB

# SQL Server Scale

| Feature Name | Enterprise | Business Intelligence | Standard | Web | Express |
|---|---|---|---|---|---|
| Maximum Compute Capacity Used by a Single Instance (SQL Server Database Engine)[1] | Operating System maximum | Limited to lesser of 4 Sockets or 16 cores | Limited to lesser of 4 Sockets or 16 cores | Limited to lesser of 4 Sockets or 16 cores | Limited to lesser of 1 Socket or 4 cores |
| Maximum Compute Capacity Used by a Single Instance (Analysis Services, Reporting Services) [1] | Operating system maximum | Operating system maximum | Limited to lesser of 4 Sockets or 16 cores | Limited to lesser of 4 Sockets or 16 cores | Limited to lesser of 1 Socket or 4 cores |
| Maximum memory utilized (SQL Server Database Engine) | Operating system maximum | 64 GB | 64 GB | 64 GB | 1 GB |
| Maximum memory utilized (Analysis Services) | Operating system maximum | Operating system maximum | 64 GB | N/A | N/A |
| Maximum memory utilized (Reporting Services) | Operating system maximum | Operating system maximum | 64 GB | 64 GB | N/A |
| Maximum relational Database size | 524 PB | 524 PB | 524 PB | 524 PB | 10 GB |

Source: http://msdn.microsoft.com/en-us/library/cc645993.aspx

# SQL Server Nice to Know…

- Multiple Databases on an Instance
- SQL Server Versions side by side
- Restore only on Same or Newer

# Benefits from SQL

- Standardized Data Format
- Server Side Filtering
- Complex Reports made easier and faster
- Share Data
- Online Backup

# SQL

## Connectivity Kit What's New

# CK What's New

- Updated Supported data types
  - Microsoft SQL Server 2012
  - IBM DB2 10.1
- Auto Reconnect
- Improved Restructure Logic

# Data Types

- Support for new Data Types like
  - Date, DateTime2, Time
  - Supported on MS SQL, DB2 and ODBC

```
DEFAULT_MAP_DF_TO_SQL_TYPE_SCHEMA
```

# DEFAULT_MAP_DF_TO_SQL_TYPE_SCHEMA

- Mapping Schema's

- MAP_DF_TO_SQL_TYPE_CK5
- MAP_DF_TO_SQL_TYPE_SQL2000
- MAP_DF_TO_SQL_TYPE_SQL2005
- MAP_DF_TO_SQL_TYPE_SQL2008
- MAP_DF_TO_SQL_TYPE_SQL2012
- MAP_DF_TO_SQL_TYPE_CK6

| ` | CK5 | SQL2000 | SQL2005 | SQL2008 | SQL2012 | CK6 |
|---|---|---|---|---|---|---|
| DF_ASCII | Char | Char | Char | Char | Char | Char |
| DF_DATE | Datetime | Datetime | Datetime | Date | Date | Date |
| DF_DATETIME | Datetime | Datetime | Datetime | Datetime2 | Datetime2 | Datetime2 |
| DF_TEXT | Text | Text | Varchar(max) | Varchar(max) | Varchar(max) | Varchar(max) |
| DF_BINARY | Binary/Image | Binary/image | Varbinary(max) | Varbinary(max) | Varbinary(max) | Varbinary(max) |

# CK Data Types: nchar & nvarchar

- Unicode Data Type
- Maps to Clients Code Page
- Data might get lost when writing back

# Auto Reconnect

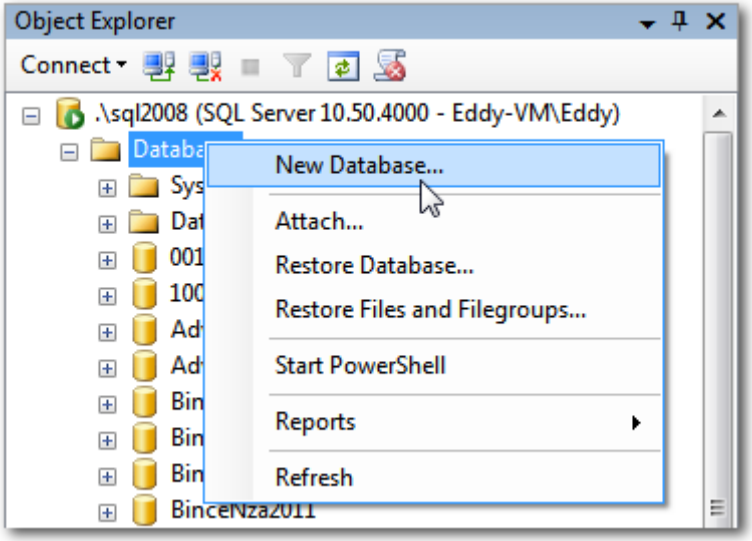
- Reconnect when connection is lost
- Open and Find will reconnect
- Supported on MS SQL, DB2 and ODBC
- AUTO_RECONNECT 1

# SQL

How to…

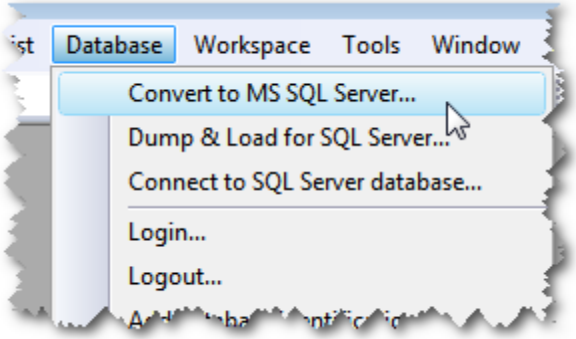
# Create Database

- SQL Server Management Studio
- Use SQL defaults
- Mind collating



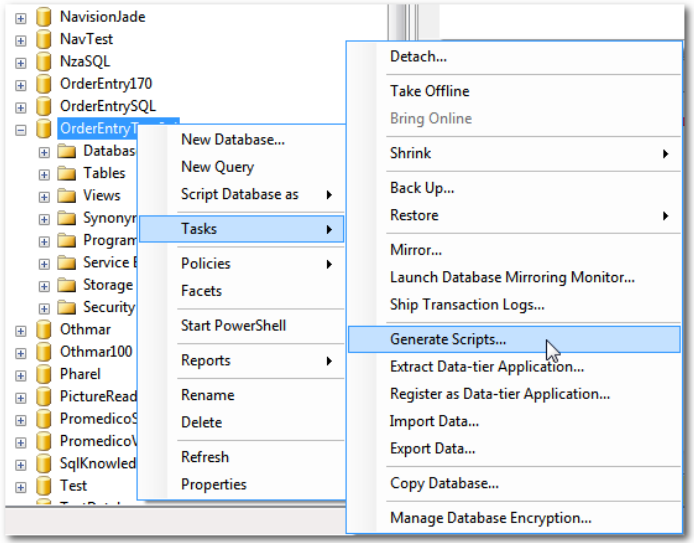
# Convert Existing to SQL

- Database Builder converts data
- Standard vs Recnum Tables
  - 18.1 can create clustered indexes
- Structure: DataFlex Studio Leading



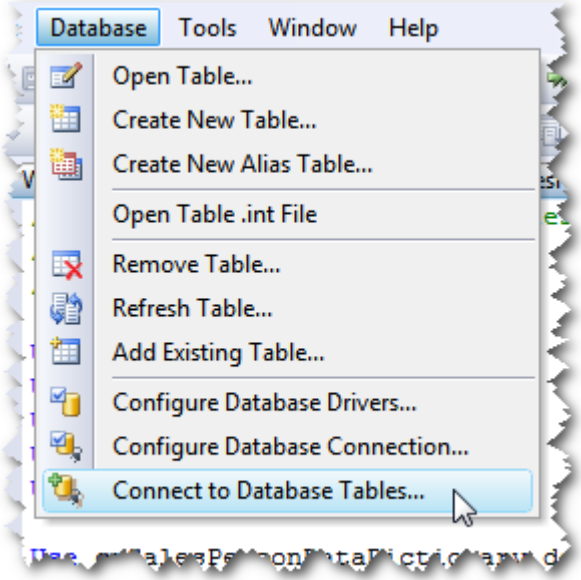
# SQL Tasks > Generate Scripts

- Easy way to copy between versions
- Structure only, Data only, or Both
- See Advanced Options button



# Attach to Existing SQL

- Setup Database Connection
- Connect to Tables from Studio
- Structure: SQL is leading



# Are you enjoying the view?

- Join data from different tables databases
- Include complex filters
- Limit the number of columns
- Database structure independency

```
CREATE VIEW [dbo].[vwDebtors] AS

    SELECT
        [cicmpy].[debnr]
        ,[cicmpy].[cmp_name] AS [debnaam]
        ,[cicmpy].[textfield1] AS [SalespersonName]
    FROM
        [100].[dbo].[cicmpy]
    WHERE
        [cicmpy].[debnr] IS NOT NULL
```

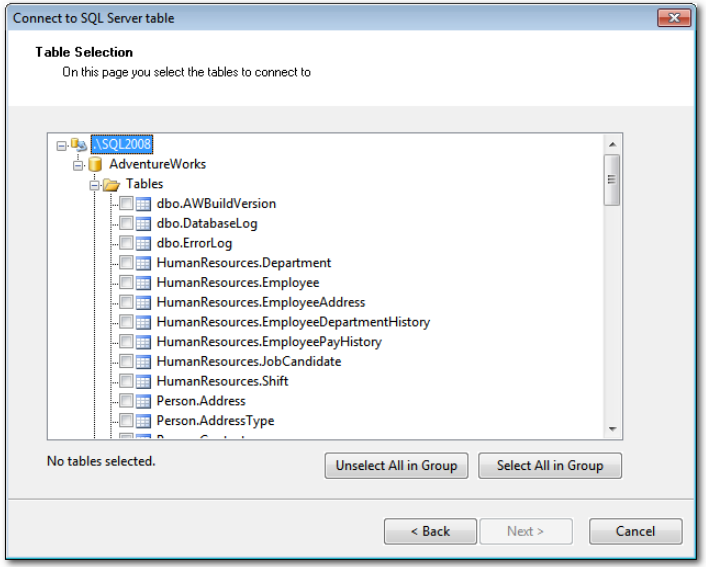
# Create a View

```
CREATE VIEW [vwCustomerLast120Days] AS
SELECT    Customer_Number
FROM      Customer
WHERE     EXISTS (  SELECT Order_Number
                    FROM OrderHea
                    WHERE OrderHea.Customer_Number = Customer.Customer_Number
                              AND DATEDIFF(dd, Order_Date, GETDATE()) < 120)
```

- Benefits: Treat like a table, Reusable
- Downside: Distributed code

# How to connect

- Connect to views
  - Views are not tables
  - Views can be updatable



# SQL Side Filter

- Works in driver version 4 or higher
- DF_FILE_SQL_FILTER
- DF_FILE_SQL_FILTER_ACTIVE

```
EXISTS (
        SELECT Order_Number FROM OrderHea
        WHERE OrderHea.Customer_Number = Customer.Customer_Number
                AND DATEDIFF(dd, Order_Date, GETDATE()) < 120
)`
```

# SQL Side Filter

- Use Data Dictionaries
  - psSQLFilter
  - pbUseDDSQLFilters
  - pbApplyGlobalSQLFilters (defaults to FALSE!)

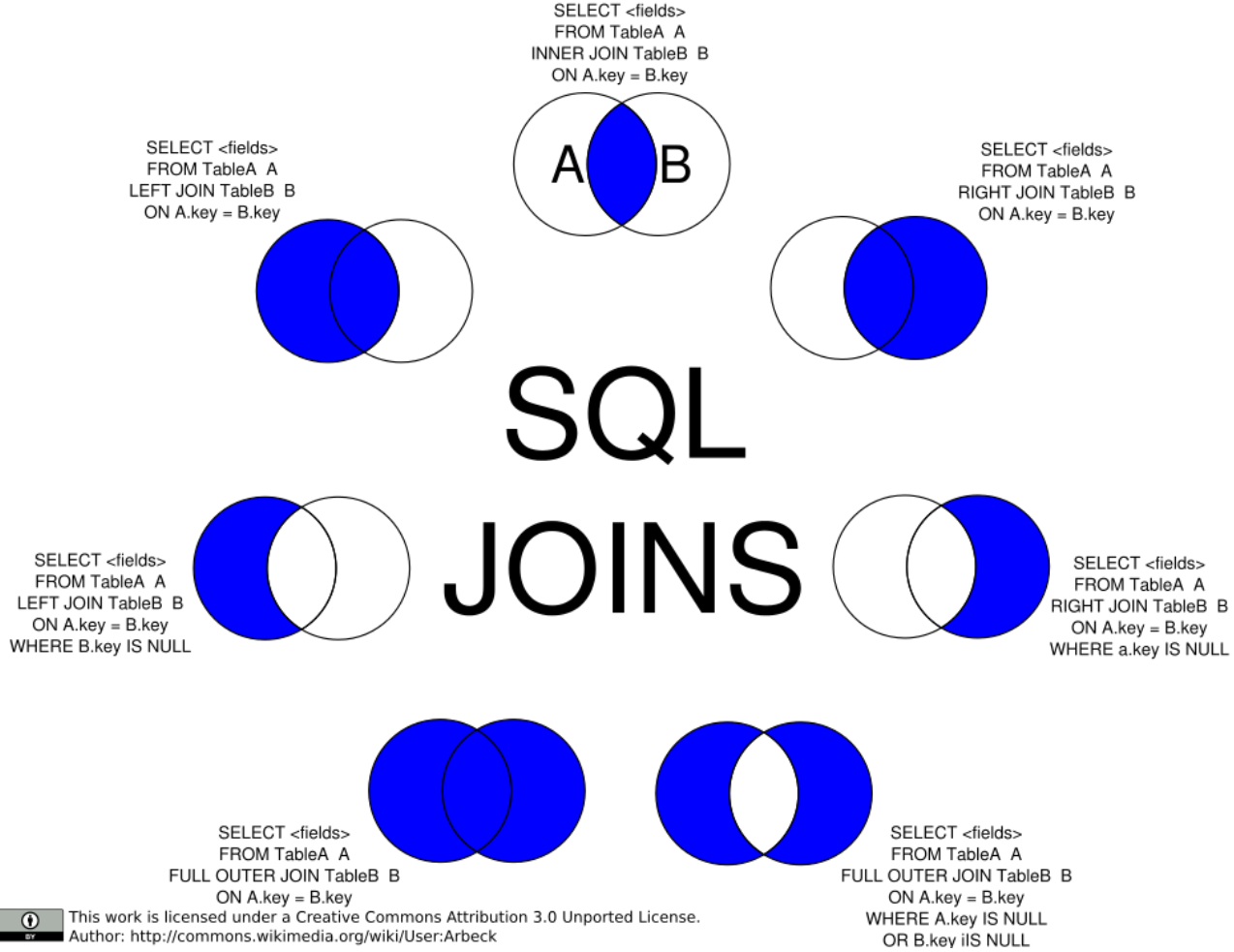
```
Object oCustomer_DD is a Customer_DataDictionary

    Set pbUseDDSQLFilters to True
    Set pbApplyGlobalSQLFilters to True
    Procedure OnConstrain
        Set psSQLFilter to "(LEFT([State], 1) = 'C')"
    End_Procedure

End_Object
```

# SQL

Some SQL Tricks…



# SQL Tricks: Back-up a table

```
-- Create backup
SELECT *
INTO CustomerBackup
FROM Customer

-- Empty table
TRUNCATE TABLE Customer
-- Alternative: DELETE FROM Customer

-- Copy backup data back into table
INSERT INTO Customer
SELECT * FROM CustomerBackup

-- Drop the backup table
DROP TABLE CustomerBackup
```

# SQL Tricks: UNION

- Concatenate result sets
- First result set defines names and data types

```
SELECT 'Customer' AS [Source], CAST([Customer_Number] AS varchar) AS [ID], [Name] AS [Description] FROM [Customer]
UNION ALL
SELECT 'Invt', [Item_ID], [Description] FROM [Invt]
UNION ALL
SELECT 'SalesP', [ID] ,[Name] FROM [SalesP]
UNION ALL
SELECT 'Users', [LoginName], [Full_Name] FROM [Users]
UNION ALL
SELECT 'Vendor', CAST([ID] AS varchar), [Name] FROM [Vendor]
```

# SQL Tricks: DISTINCT

- SELECT unique rows
- Cannot have aggregates; use GROUP BY for that

```
SELECT DISTINCT
      [Item_ID]
      ,[Qty_Ordered]
      ,[Price]
FROM
      [OrderDtl]
```

# SQL Trick: CASE

- Great to put in conditionals
- Use it anywhere; even in joins

```
SELECT
    [Customer].[Customer_Number]
    ,[Customer].[Name]
    ,[CustomerOrderTotal].[Order_Total]
    ,CASE
        WHEN [CustomerOrderTotal].[Order_Total] IS NULL THEN 'Call!'
        ELSE 'Ok'
        END AS [Status]
FROM
    [Customer]
    LEFT OUTER JOIN (
        SELECT
```

# SQL Trick: Four-Way Path

- Join Data from different servers
- Register the external server

```
SELECT
        *
FROM
        [Server].[Database].[Schema].[Table]
```

```
SELECT
    *
FROM
    [eddy-vm\sql2008].[OrderEntryTestSql].[dbo].[Customer]
```

# SQL

Create some samples