

Synergetic

Integration Guide

To install Wonde at your school you will need access to the SQL database, administrator privileges to install and two serial keys provided by Wonde.

If you do not have these keys please email support@wonde.com with your school's name, your request and which SIS you use (Synergetic).

Preparing the database

Wonde requires a user account on the MSSQL server to access school data. Follow the steps below to create a user with the correct permissions:

It is advised to make sure that you have sufficient backups of your database before performing the following steps with Wonde or any other provider.

1. Log in to Microsoft SQL Server Management Studio as an administrator.
2. Expand databases and highlight your SIS database within the **Object Explorer**.
3. Click the **New Query** button.
4. Paste in the contents of the **Wonde-Synergetic.sql** section at the end of this document into the query area.
5. At the top of the script, populate the password variable with a random 15 character long password and keep a note as you will need this password again later (Please avoid symbol characters as these may not transfer properly for Wonde).
6. Make sure you have your SIS database selected while you click the **Execute** button. This will add our user and give a confirmation message that the script was successful.

Installing the Wonde client

1. Install the Wonde software using the **WondeInstaller.msi** file, provided at <https://wonde.com/wonde-installer>, by going through the installer wizard.
2. Go to the installation file path for Wonde and open the Wonde client (Wonde.exe).
3. Click **Settings** and then **Site licence**.
4. Add the **Site key** provided to yourselves and click Save settings.
5. Click on **File** and then **Add school**.
6. Add the **Licence key** provided to yourselves and tick the MSSQL option
7. Add your database credentials:
 - a. **Server name**. This is the name of the server that your Synergetic MSSQL database is hosted on.
 - b. **Database name**. This is the name of your Synergetic database.
 - c. **Username**. This should be "wonde" unless you modified while preparing the database user.
 - d. **Password**. This is the password you created previously.
8. Click on **Add School** and this will then display your school details in the Wonde client.
9. Go to the **services.msc** and start up the **Wonde Comms Service** and **Wonde Recovery Service**. It's sometimes necessary to terminate the processes for the service to restart if it is already running.

Verification

A Wonde engineer will be required to test that your school's installation was successful so please let us know when this has been completed by filling in the following Google form: <https://goo.gl/forms/xInviaiame0BzkzK2>.

If you are unable to access the Google form then please email into support@wonde.com with your name, school name and postcode to let us know.

Wonde-Synergetic.sql

-- Consts

```
DECLARE @UserName VARCHAR(MAX) = 'wonde';
```

```
DECLARE @Password VARCHAR(MAX) = '**REPLACE_WITH_STRONG_PASSWORD**';
```

```
DECLARE @RoleName VARCHAR(MAX) = 'Syn_Wonde_dbo';
```

-- Variables

```
DECLARE @SQL VARCHAR(MAX);
```

```
BEGIN TRY
```

-- Create Wonde configuration group

```
IF NOT EXISTS (SELECT * FROM luConfigGroup WHERE Code = 'Wonde')
```

```
BEGIN
```

```
    INSERT luConfigGroup (Code, Description)
```

```
    SELECT 'Wonde', 'Wonde'
```

```
END
```

-- Create Login and Database User

```
IF NOT EXISTS (
```

```
    SELECT *
```

```
    FROM sys.server_principals p
```

```
    WHERE p.[type] = 'S'
```

```
        AND p.[name] = @UserName
```

```
)
```

```
BEGIN
```

```

SET @SQL = CONCAT('CREATE LOGIN ', @UserName, ' WITH PASSWORD = "',
@Password, '" , DEFAULT_DATABASE=[master], CHECK_EXPIRATION=OFF,
CHECK_POLICY=OFF');
PRINT @SQL
EXEC(@SQL);
END

IF NOT EXISTS (
SELECT *
FROM sys.database_principals p
WHERE p.[type] = 'S'
AND p.[name] = @UserName
)
BEGIN
SET @SQL = CONCAT('CREATE USER ', @UserName, ' FOR LOGIN ', @UserName);
PRINT @SQL
EXEC(@SQL);
END

```

```

-- Create Database Role / Recreate if Exists

```

```

IF EXISTS (
SELECT *
FROM sys.database_principals p
WHERE p.[type] = 'R'
AND p.[name] = @RoleName
)
BEGIN
SET @SQL = CONCAT('DROP ROLE ', @RoleName);
PRINT @SQL
EXEC(@SQL);
END

SET @SQL = CONCAT('CREATE ROLE ', @RoleName);
PRINT @SQL

```

```
EXEC(@SQL);
```

```
-- Assign User to Role
```

```
IF NOT EXISTS(
  SELECT TOP (1) dp.name
  FROM sys.database_role_members dr
  INNER JOIN sys.database_principals dp ON (dp.principal_id =
dr.role_principal_id)
  INNER JOIN sys.database_principals su ON (su.principal_id =
dr.member_principal_id)
  WHERE dp.name = @RoleName
  AND su.name = @UserName)
AND EXISTS(SELECT * FROM sys.database_principals dp2 WHERE dp2.name =
@UserName)
BEGIN
  SET @SQL = CONCAT('ALTER ROLE ', @RoleName, ' ADD MEMBER ', @UserName);
  PRINT @SQL
  EXEC(@SQL)
END
```

```
-- Grant Select Permissions to Role
-- To add new permissions, add table name to list and run the script
```

```
IF OBJECT_ID('tempdb..#DatabasePermissions') IS NOT NULL
  DROP TABLE #DatabasePermissions
```

```
CREATE TABLE #DatabasePermissions
(
  UserRole VARCHAR(255),
  ObjectName VARCHAR(255),
  ExecFlag BIT DEFAULT(0),
```

```
SelectFlag BIT DEFAULT(0),
InsertFlag BIT DEFAULT(0),
UpdateFlag BIT DEFAULT(0),
DeleteFlag BIT DEFAULT(0),
object_id INT NULL,
schema_id INT NULL
)
```

```
INSERT INTO #DatabasePermissions
```

```
(
  UserRole,
  ObjectName,
  SelectFlag,
  object_id,
  schema_id
)
```

```
SELECT
```

```
  @RoleName,
  ss.name + '.' + QUOTENAME(so.name),
  SelectFlag = 1,
  so.object_id,
  ss.schema_id
```

```
FROM sys.objects so
```

```
  INNER JOIN sys.schemas ss ON (ss.schema_id = so.schema_id)
```

```
WHERE so.name IN (
```

```
  'Addresses',
  'AttendanceMaster',
  'Community',
  'FileSemesters',
  'MedicalCondition',
  'MedicalAllergy',
  'Staff',
  'StudentClasses',
  'Students',
  'StudentSemester',
  'SubjectAssessments',
```

'SubjectClasses',
'Timetable',
'TimetableSubstitutes',

'fnDateRange',
'luBoardingHouse',
'luDepartment',
'luTimetableGroup',
'luForm',
'luHouse',
'luTutor',
'luCampus',
'luLanguage',
'luRelationship',
'luReligion',
'luRoom',
'luStaffCategory',
'luYearLevel',
'luMedicalConditionType',

'pvTimetableDefinition',
'pvTimetableDefinitionAll',
'tAttendances',
'tMedicalIncidents',

'vStudentsAll',
'vCommunityAddresses',
'vstudentcontactaddress',
'vPhotos',
'vStaff',
'vStudentClasses',
'vStudents',
'vSubjectClassesStaff',
'vStudentContactsRelationOnly',
'vFutureStudents',
'vFutureContactAddress'


```

)
AND ss.name = 'dbo'

DECLARE @ExecFlag BIT = 0,
@ObjectName VARCHAR(255)

DECLARE PermissionCursor CURSOR LOCAL FORWARD_ONLY READ_ONLY
FOR
SELECT
    dp.UserRole,
    dp.ObjectName
FROM #DatabasePermissions dp
ORDER BY UserRole,ObjectName

OPEN PermissionCursor
FETCH NEXT FROM PermissionCursor INTO
@RoleName,
@ObjectName

WHILE @@FETCH_STATUS = 0 BEGIN
SET @SQL = 'GRANT SELECT,'

SET @SQL = SUBSTRING(@SQL,0,LEN(@SQL)) + ' ON ' + @ObjectName + ' TO ' +
@RoleName

PRINT @SQL
EXEC(@SQL)
FETCH NEXT FROM PermissionCursor INTO
@RoleName,
@ObjectName
END
CLOSE PermissionCursor
DEALLOCATE PermissionCursor

-----
-- Execute permissions for stored procedures

```

```
SET @SQL = 'GRANT EXECUTE ON fnTimetableGroupDayNumber TO ' + @RoleName;  
PRINT @SQL;  
EXEC(@SQL);
```

```
SET @SQL = 'GRANT EXECUTE ON fnDateOnly TO ' + @RoleName;  
PRINT @SQL;  
EXEC(@SQL);
```

```
SET @SQL = 'GRANT EXECUTE ON sptxiAttendances TO ' + @RoleName;  
PRINT @SQL;  
EXEC(@SQL);
```

```
SET @SQL = 'GRANT SELECT,INSERT,DELETE ON xiAttendances TO ' + @RoleName;  
PRINT @SQL;  
EXEC(@SQL);
```

```
END TRY  
BEGIN CATCH  
    THROW;  
END CATCH
```