MSSQL and MySQL Database Integration

You can integrate your knowledgebase with MSSQL and MySQL databases to keep your records synchronized. This integration uses an external system adapter (ESA) to sync Agiloft with the database. This article uses two example tables to illustrate the configuration, but you can sync more or fewer tables as needed.

The integration setup is almost identical for MSSQL and MySQL, but the examples differ slightly.

Prerequisites

First, make sure that you have all the necessary credentials to access the database:

- Username
- Password
- Database name
- Server address and port

Preparing Database Tables

First, evaluate the content in your KB and your database and identify what entities you need to sync, and the nature of that synchronization. In this example, consider a database with two tables, ext_parent_table and ext_child_table, that must be kept in sync with two corresponding tables in Agiloft. Every time a change is made in either system, it must be reflected in the other.

For the database tables, consider this example code for both MSSQL and MySQL:

MSSQL Example

```
SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO

CREATE TABLE [dbo].[ext_parent_table](

[id] [int] IDENTITY(1,1) NOT NULL,

[name] [nchar](255) NULL,

[lmd] [datetime] NULL,

CONSTRAINT [ext_parent_table_pk] PRIMARY KEY NONCLUSTERED
(

[id] ASC

)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

/****** Object: Table [dbo].[ext_child_table] Script Date: 9/6/2021 2:41:02 AM
******/

SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO

CREATE TABLE [dbo].[ext_child_table](

[id] [int] IDENTITY(1,1) NOT NULL,

[ext_parent_id] [int] NULL,

[name] [nchar](255) NULL,

[lmd] [datetime] NULL,

CONSTRAINT [ext_child_table_pk] PRIMARY KEY NONCLUSTERED

(

[id] ASC

)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO



ALTER TABLE [dbo].[ext_child_table] WITH CHECK ADD CONSTRAINT
[ext_child_table_ext_parent_table_id_fk] FOREIGN KEY([ext_parent_id])

REFERENCES [dbo].[ext_parent_table] ([id])
```

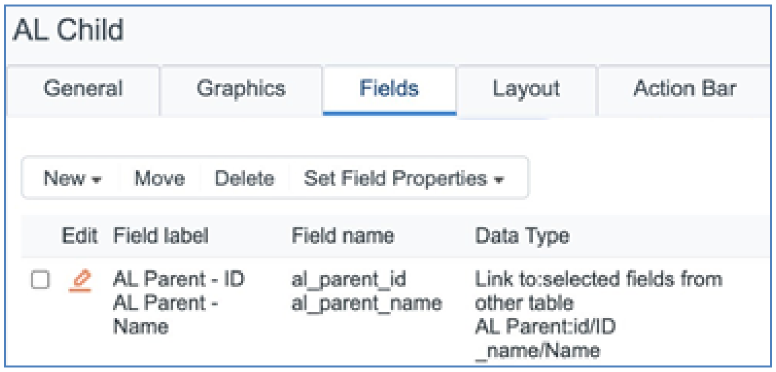
```
GO



ALTER TABLE [dbo].[ext_child_table] CHECK CONSTRAINT
[ext_child_table_ext_parent_table_id_fk]

GO
```

MySQL Example

```
create table ext_parent_table(

id int auto_increment        primary key,

name varchar(255) null,

lmd timestamp default CURRENT_TIMESTAMP not null);

create table ext_child_table(

id int auto_increment        primary key,

name varchar(255) null,

ext_parent_id int null,

lmd timestamp default CURRENT_TIMESTAMP not null,

constraint ext_child_table_ext_parent_table_id_fk

foreign key (ext_parent_id) references ext_parent_table (id));
```

In both cases, ext_child_table is related to ext_parent_table using the ext_parent_id field.

In Agiloft, you must identify a corresponding table to sync with the database table. To follow the example above, consider matching tables in Agiloft, AL Parent and AL Child. In the AL Parent table, there is a custom Text field called Name; in the AL Child table, there is another Text field called Name, and a linked set pointing to the AL Parent table's ID and Name fields.



AL Child

Edit	Field label	Field name	Data Type
	AL Parent - ID AL Parent - Name	al_parent_id al_parent_name	Link to:selected fields from other table AL Parent:id/ID _name/Name

AL Child table's linked set to AL Parent

Before these tables can be synced, every table must have its own primary key field, which is not null and increments automatically, and a DATETIME field, which represents the date and time of the last change to the record. In Agiloft, the ID and Last Modified Date fields serve these purposes. To add a last modification timestamp, a mechanism must be created in the MSSQL or MySQL database to populate that timestamp every time a record is created or updated. Consider these example triggers, one for MSSQL and one for MySQL:

MSSQL Example

```
CREATE TRIGGER tr_insert_ext_parent_table ON ext_parent_table
AFTER INSERT
AS
BEGIN
UPDATE ext_parent_table
SET lmd = GETDATE()
FROM inserted
WHERE ext_parent_table.id = inserted.id;
END
GO


CREATE TRIGGER tr_update_ext_parent_table ON ext_parent_table
AFTER UPDATE
AS
BEGIN
UPDATE ext_parent_table
SET lmd = GETDATE()
FROM INSERTED
WHERE ext_parent_table.id = INSERTED.id;
END
GO
------
CREATE TRIGGER tr_insert_ext_child_table ON ext_child_table
AFTER INSERT
AS
BEGIN
UPDATE ext_child_table
SET lmd = GETDATE()
FROM inserted
WHERE ext_child_table.id = inserted.id;
END
GO
CREATE TRIGGER tr_update_ext_child_table ON ext_child_table
AFTER UPDATE
AS
BEGIN
UPDATE ext_child_table
SET lmd = GETDATE()
FROM INSERTED
WHERE ext_child_table.id = INSERTED.id;
END
GO
```

MySQL Example

```
CREATE TRIGGER before_ext_parent_table_update
    BEFORE UPDATE
    ON ext_parent_table FOR EACH ROW
BEGIN
            SET new.lmd = now();
END

CREATE TRIGGER before_ext_parent_table_insert
    BEFORE INSERT
  ON ext_parent_table FOR EACH ROW
BEGIN
            SET new.lmd = now();
END


CREATE TRIGGER before_ext_child_table_update
    BEFORE UPDATE
    ON ext_child_table FOR EACH ROW
BEGIN
            SET new.lmd = now();
END

CREATE TRIGGER before_ext_child_table_insert
    BEFORE INSERT
                      ON ext_child_table FOR EACH ROW
BEGIN
            SET new.lmd = now();
END
```

For every additional field you want to sync between Agiloft and the database, make sure it has a counterpart in the other system. These are mapped later in the process.

Creating a Sync Configuration

With the tables in both systems ready to go, you can create a sync configuration in Agiloft.

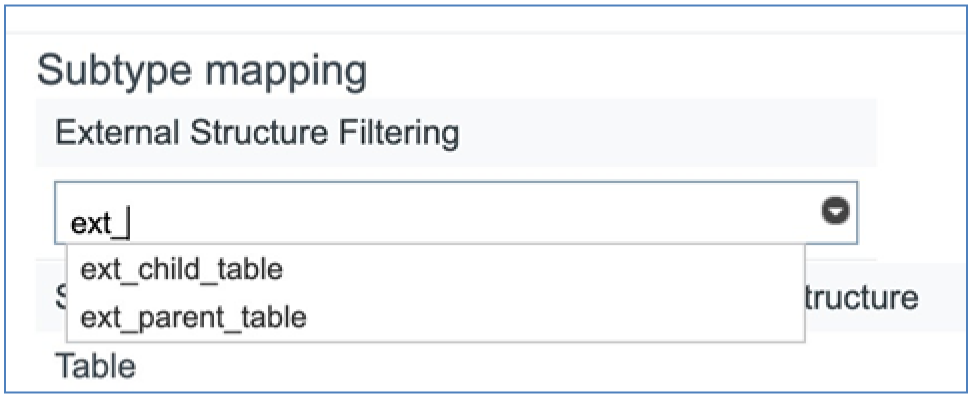
1. In Agiloft, go to **Setup > Sync > External Sync** and click New.
2. Give the configuration a clear name.
3. In External System Type, select Database.
4. Select or clear the Allow Multi to Multi Table Mappings checkbox. If you don't need to map multiple tables, clear the checkbox to enable two-way sync.

5. Select the Direction for the sync. Two-way sync is good for keeping both systems up to date, but it only works for one table in each system. If you want to sync more than one table, you can use one-way sync. One-way sync is helpful if one system is only being used to look at information, not change it. You can also use two one-way sync configurations if you need to send information back and forth between the two systems .
6. Configure the remaining settings, particularly the Conflicts section, and then click Next.
7. Set the Database type to MSSQL or MySQL, to match your database.
8. Complete the rest of the fields with the database credentials.
9. Click Next. If the credentials were entered correctly, the test connection is successful and the Mappings tab opens.

Mapping Tables

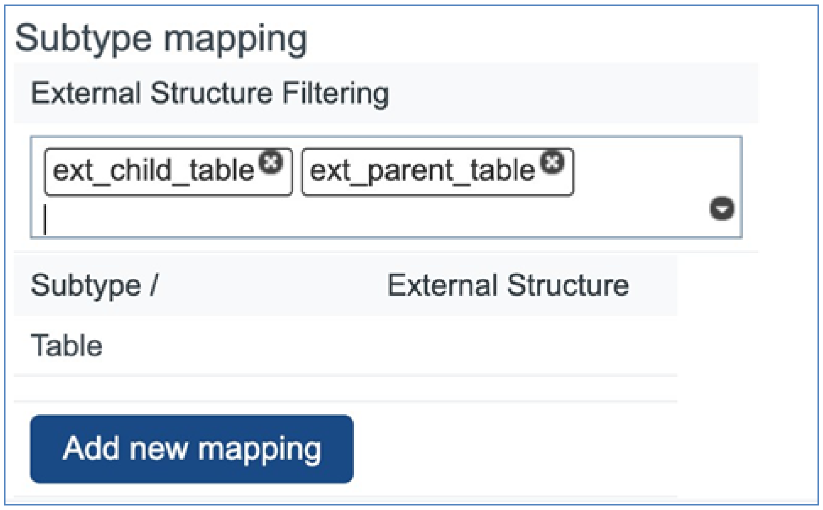
Make sure you completed the steps in Preparing Database Tables so that every table in the database has a primary key field and a DATETIME column that stores the time stamp of the record's most recent modification. These two fields are required for sync because they are used to match records across systems and to identify which record is the more recent version.

1. On the Mapping tab, search the database table name in the External Structure Filtering field and select it.



2. Continue to search and select each database table you want to sync until they are all selected.
3. For each pair of tables you want to sync:

a. Click "Add new mapping" to add a new drop-down pair to the list.



b. Use the drop-downs to select the Agiloft table and the MSSQL or MySQL table that should be mapped.

c. Click Map to open the Field Mapping tab.

Mapping Fields

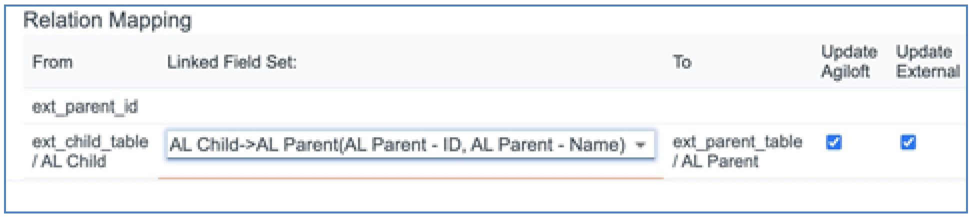
On the Field Mappings tab, you can configure settings for the specific tables being mapped, and set which fields in the tables are synced.

1. In the Record Modification Timestamp Field, select the field that stores the date and time of the last update to the record. This field is compared between systems to determine which record's values are synced to the other system.
2. For System Update Type, choose the sync direction for the table. You can choose a different sync direction for individual tables than you chose for the overall sync on the General tab, but you can't contradict the General tab setting. For example, if you chose to Update Agiloft Only on the General tab, you can't choose the Export option here to send data to the database.
   - **Synchronize**: Updates records in both systems based on the timestamp field.
   - **Export**: Updates records in the MSSQL or MySQL database to match field values in Agiloft.
   - **Import**: Updates records in Agiloft to match field values in the MSSQL or MySQL database.
3. In Allowed Operations, choose which sync operations are allowed during the sync.
4. Determine how the record ID is generated during the sync. If you generate record IDs with Agiloft, you must also specify a prefix and a table column to store the generated IDs.
5. If desired, select any actions you want to run after a successful sync.
6. Choose whether to include records in related tables. Selecting this option can impact performance.
7. Map the primary key, or ID, from the database to the ID of the Agiloft table and select Identifying. Clear the Update in Agiloft and Update in External checkboxes, to prevent this field from being overwritten.

8. Map the last modification date from the database to the Date Created field of the Agiloft table and select Last Modified. Clear the Update in Agiloft and Update in External checkboxes, to prevent this field from being overwritten.
9. For each additional database field you want to sync, select the corresponding Agiloft field from the drop-down. Then, choose whether the field value is updated in Agiloft, the database, or both. Update options are disabled if the sync direction does not permit them. For fields with multiple choice values, complete the dialog box that opens to map individual choice fields.

   > Use this tab to map links to single fields, but not complete linked sets. Linked sets should instead be mapped below, on the Relation Mapping tab.

10. Use the Identifying column to select fields to identify matching records between the systems. If you want to match records only when all the Identifying fields match, select the "Use strict match for identification" option. Otherwise, the system first attempts to identify matches by all fields, and if no match is found, it then narrows down the Identifying fields one at a time until a match is found.
11. Click Next.
12. On the Filters tab, if you need to prevent records created or updated in one system from being synced to the other, create a saved search to filter the desired records.
13. Click Finish. The Field Mapping wizard closes and returns you to the Mapping tab.
14. Click Next to proceed to the Relation Mapping tab.
15. On this tab, use the drop-down to map linked relationships, and then use the checkboxes to control where this information is updated. In the given example, here, you would map the linked relationship between AL Child and AL Parent.



Relation Mapping

From	Linked Field Set:	To	Update Agiloft	Update External
ext_parent_id				
ext_child_table / AL Child	AL Child->AL Parent(AL Parent - ID, AL Parent - Name)	ext_parent_table / AL Parent	✓	✓

16. On the Running tab, choose whether synchronization is initiated manually or by actions and rules.
17. On the Export tab, customize any export settings related to the sync configuration. This allows you to transfer your sync configuration to another KB, if desired.
18. Click Finish.

Testing the Integration

Test the integration by creating records in one system, running the sync, and then checking the other system to see if those records were duplicated as expected. Then, for two-way sync, test in reverse, creating records in the other system and syncing them back to the first.