Using Dream Report with Ignition® Data Logging

Ignition® SCADA by Inductive Automation provides a few different ways to log process data into SQL database. One of the methods is to simply select tags to be historized for trending, and they will be logged to a set of “generic” tables in the project’s database. Another method is to create “Transaction” groups in Ignition Designer, where you select the items/tags to be logged, storage rate, and the specific SQL table into which the data will be logged. Dream Report can access either of these data table types, but with slightly different approaches.

I. Tag History Logging

In the Ignition configuration environment, the project database is configured by creating a connection to a specific database, on a specific database platform (MySQL, MS SQL Server, etc.)

Then, under the “Tags – History” configuration, you can specify how the data is “partitioned” – i.e., the mechanism by which new tables are created periodically in which the data will be logged:
The default data partition length is set for 1 Month. What this means is that every month, a new table will be created in the SQL database, and the tag history will be logged to that month’s table. From a trending (or query) efficiency standpoint, smaller tables of data can be queried, allowing for faster data retrieval. The downside to the data partitioning is that from reporting standpoint, it is difficult to know which table name(s) to query. To address this, you can configure a very log data partition – say, 10 years – or, simply not check the “Enable Data Partitioning” option. This is preferable for reporting, as you will always have a specific set of tables to work with, but may result in slower historical trend updates in Ignition.

Once the historical logging has been configured in Ignition, the project’s database will get created, with the necessary tables. The image below shows a database for the Ignition “Demo” project, created in the MySQL database system. With no data partitioning configured, a single table, named sqlth_1_data below, is created, and all data will be logged to that table. If partitioning were enabled, multiple tables (named like sql_data_1_2014_10 below – containing data for the month of Oct, 2014) would be created. We will assume that a single table will be used.
Looking at the contents of the table, we see that the data is not easily understood at first glance:
A few things stand out here, which will pose a challenge for reporting on the data:

- Tagnames are not logged with the data values, only a numeric tagid
- There are separate fields for integer, float, and string data types
- The datestamp is shown in Unix time format

This table separate table will need to be joined with the sqlth_te table to cross-reference the tag ID’s with the actual tagnames:

Fortunately, we can create a SQL “view” to organize the data into a human-readable form. We actually need to create two views – one to display the numeric data (integer, float and discrete data points, and one for string tags.) The two views are scripted at follows, but please note that the syntax will vary slightly from one database package to another:

**View: v_ignitiondata**

```sql
CREATE
ALGORITHM = UNDEFINED
DEFINER = `root`@`localhost`
SQL SECURITY DEFINER
VIEW `v_ignitiondata` AS
select
  from_unixtime(`d`.`t_stamp` / 1000) AS `DateTime`,
  `t`.`tagpath` AS `tagpath`,
  `d`.`intvalue` AS `Value`
from
  (`sqlth_1_data` `d`
  join `sqlth_te` `t` ON ((`d`.`tagid` = `t`.`id`)))
where
```

- 4 -
union select
  from_unixtime((`d`.'t_stamp' / 1000)) AS 'DateTime',
  `t`.'tagpath' AS 'tagpath',
  `d`.'floatvalue' AS 'Value'
from
  (`sqlth_1_data` `d` join `sqlth_te` `t` ON ((`d`.'tagid' = `t`.'id')))
where
  (`d`.'floatvalue' is not null)
order by 'DateTime'

View:  v_ignitiondata_strings

CREATE
  ALGORITHM = UNDEFINED
  DEFINER = `root`@`localhost`
  SQL SECURITY DEFINER
VIEW 'v_ignitiondata_strings' AS
  select
    from_unixtime((`d`.'t_stamp' / 1000)) AS 'DateTime',
    `t`.'tagpath' AS 'tagpath',
    `d`.'stringvalue' AS 'Value'
from
  (`sqlth_1_data` `d` join `sqlth_te` `t` ON ((`d`.'tagid' = `t`.'id')))
where
  (`d`.'stringvalue' is not null)
order by 'DateTime'

When we query the v_ignitiondata view, for instance, we see data nicely displayed for reporting. We will reference this view from Dream Report when creating reports:
Note: For more information configuring Ignition data logging, please refer to the Ignition product documentation.

II. Transaction Groups

Transaction Groups are created in Ignition Designer:

In the example below, we created a transaction group named “Logging Group”, selected a set of tags to include in that group, and set the storage rate to be every 10 seconds. Additionally, we specified the table name (into which the data will be logged) as “Dream Report_data_log”.

The resulting table created in the (MySQL) database is easily understood, showing a field for each logged tag, and a timestamp field showing the local date and time:
We’re now ready to create reports in Dream Report with the Ignition-logged data...

**Dream Report Communication Driver Configuration**

In Dream Report we will use the ODBC Communication Driver, to connect the Ignition SQL database, and then will connect to either the SQL view(s) or transaction group tables in the database. The following steps assume you have first used the Windows ODBC Manager (32-bit) to create an ODBC DSN (Data Source Name) to the specific Ignition database being used.

To configure the connections:

1. Open the “Communication Configuration Wizard” in Dream Report Studio, and select the “ODBC Historical Values” driver from *Open Communication Protocols > ODBC > ODBC Historical Values*
2. Enter a **Logical Name** for the driver, and then click “Configure”

3. Under the “Database Definition” section, Select the previously-created ODBC DSN from the list of DSN’s

4. Enter a valid SQL username and password, and click “Connect”

5. Once a connection is established to the database, all available tables and views will be displayed in the “Item History Data” section

6. Since the SQL view and the transaction group table have different table structures, we need to configure the ODBC driver appropriately for each connection type:
   a. SQL View – this is a “generic” table structure that does not use the tagnames as the field names. So, we need to select the “Database Type” to be “AnyDB structure”, and then specify below the Table Fields that specify the Item Names, Item Values and Date:

   ![Database Configuration Screen](image)

   - Table Field for Item Name
   - Table Field for Item Value
   - Table Field for Date (e.g., “DateTime”)

   b. Transaction Group Table – each of these tables are structured with each item/tagname being a column in the table. As such, we need to select the “Database Type” to be “Column-Item structure”, and simply select the “DateTime” field from the list for the “Table Field for Date” setting:
7. Once the driver has been configured, click "OK", and then "Add", to add it to the “Defined Drivers List”

8. Click “OK” to close the “Communication Configuration Wizard”

At this point, you are ready to start building reports. Create a new report, select any reporting object (Table, Chart, Expression, etc.), and simply browse for the tag(s) you want to add to that object. For example, to add Ignition tags to a Dream Report Line Chart object, select the “External History Server” option from the “Select Data Item” section, select the driver logical name created earlier, and a list of logged Ignition tags will be displayed:
And the resulting report:

![Moving Averages - Last 1 Hour chart]

- Accum Level
- Disch. Pressure
- Discharge Temp.

<table>
<thead>
<tr>
<th>Tagname</th>
<th>Max</th>
<th>Min</th>
<th>Avg</th>
<th>Std. Dev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accum. Level</td>
<td>100.00</td>
<td>0.00</td>
<td>42.46</td>
<td>29.38</td>
</tr>
<tr>
<td>Disch. Pressure</td>
<td>60.00</td>
<td>2.00</td>
<td>36.18</td>
<td>17.83</td>
</tr>
<tr>
<td>Disch. Temp</td>
<td>212.00</td>
<td>0.00</td>
<td>104.84</td>
<td>69.34</td>
</tr>
</tbody>
</table>