The purpose of this project is to illustrate how you can integration legacy systems that lack APIs and must be integrated at the beginning via database tables and JMS queues in the short term.

Note that longer term solution should always be based on building managed APIs around the systems for scalability, maintainability, and functionality. More details can be found here: <http://www.mulesoft.com/platform/api/manager>

In the repository, there are three different Mule projects:

1. demo-mock-jms-endpoint: this is a Mule app that emulates a legacy system that can only be integrated via JMS messages. This application was built with inbound and outbound messages traveling on different queues but can easily be changed to use the same queue. This app can serve as a template for integrating with other systems using only JMS.
2. demo-from-db: This is a Mule app that emulates the process of picking up changed records from a legacy system via database connection, then send the records via JMS queue to be enriched.
3. Demo-to-db: This is a Mule app that emulates the process of receiving the enriched records from a JMS queue then insert them into a legacy system via database connection.

The basic use case is this:

1. In system A, employee records were changed and need to be enriched by an external system, B. The enriched records will then need to be sent over to system C.
2. Interfacing with systems A & C have to be done via database connections.
3. Interfacing with system B can only be done via JMS queues.
4. Changes records will be picked up from A, sent over to B to be enriched, returned by B then inserted into C.

Notes:

1. The Mule apps are written on Mule 3.4.2 but will work with 3.4.x in general as well as Mule 3.5.
2. For this example, ActiveMQ and MySQL were used to represent JMS and Database since they are open source and readily available, but the architecture will work with *any* JMS and database platforms such as Oracle AQ and Oracle database.
3. The projects also use spring property placeholders to manage connection credentials, i.e. the mule.properties files in the src/main/resources folders. This is the recommended best practice.