

FROM HARD-CODED CONDITIONS TO RULES

- ▶ The benefits:
 - Allow decision makers to directly introduce their business rules with minimum translation involved

- ▶ You will learn about JBoss Drools and Guvnor GUI
 - With abbreviated example of Product Marketing

- ▶ Look for introduction to semantic enhancements helping to bridge business language and rules
 - With a brief exposure to Decision Tables and Business Architecture Sandbox for Enterprise (BASE)

JBoss Drools and Guvnor Introduction From Code to Rule-Based Development

**if(person.getAge() > 18)
offers.add(creditCard)**



```
<rule name="person age more than 18">  
  <parameter identifier="person">  
    <class>org.drools.examples.simple.Person</class>  
  </parameter>  
  
  <java:condition>person.getAge() > 18</java:condition>  
  <java:consequence>  
    offers.add(creditCard)  
  </java:consequence>  
  
</rule>
```

FROM A RULE TO A RULE-SET

```
<rule-set name="offer rules" xmlns="http://drools.org/rules"
xmlns:java="http://drools.org/semantics/java"
xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"
xs:schemaLocation="http://drools.org/rules rules.xsd
http://drools.org/semantics/java java.xsd">
```

```
<rule name="person age more than 18">
  <parameter identifier="person">
    <class>org.drools.examples.simple.Person</class>
  </parameter>

  <java:condition>person.getAge() > 18</java:condition>
  <java:consequence>
    offers.add(creditCard)
  </java:consequence>
```

```
</rule>
```

```
</rule-set>
```

THE RULE IN A DRL FILE

```
import org.drools.examples.simple.Person
import org.drools.examples.simple.Offers
import org.drools.examples.simple.CreditCard
```

Rule: person age more than 18

when

person.age > 18

then

offers.add(creditCard);

end

EXECUTE RULES IN JAVA PROGRAM

//load the rules from the rules.java.drl file

```
RuleBase ruleBase = RuleBaseLoader.loadFromStream(  
this.getClass().getResourceAsStream( "rules.java.drl" ) );
```

```
WorkingMemory workingMemory =  
ruleBase.newWorkingMemory( );
```

```
workingMemory.assertObject(person); // get data object
```

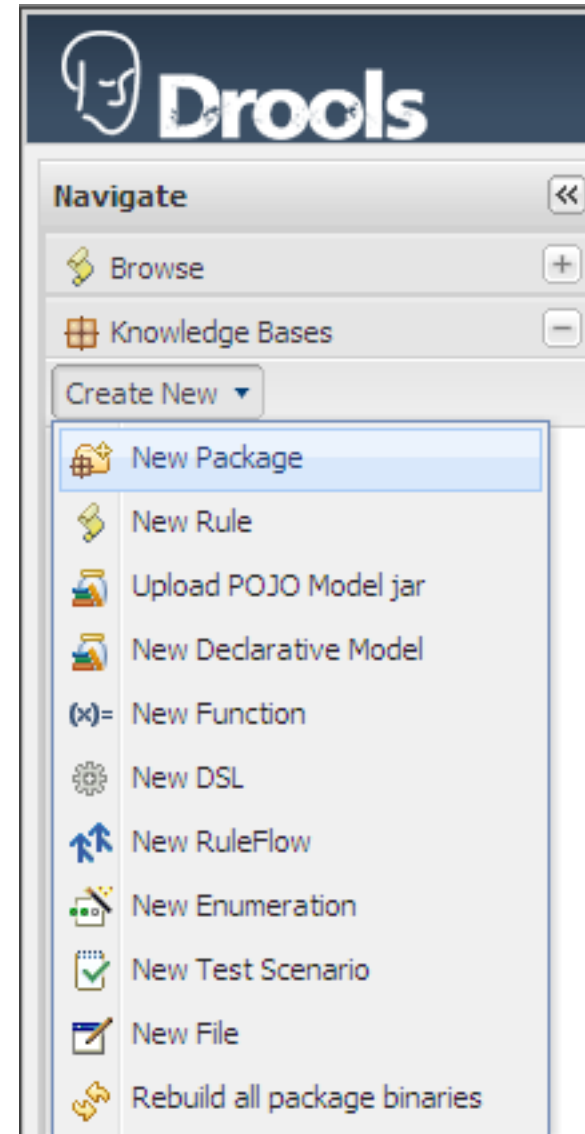
```
workingMemory.fireAllRules( );
```

GUVNOR (DROOLS ON THE WEB)

LOGIN, THEN CREATE CATEGORY AND PACKAGE

<http://server:8080/drools-guvnor>

Before creating any rules we need to establish a Category and a Package where we plan to create the rules.



<http://ITUniversity.us>

GUVNOR (DROOLS ON THE WEB)

CREATE A NEW RULE

Click on the **Browse** control

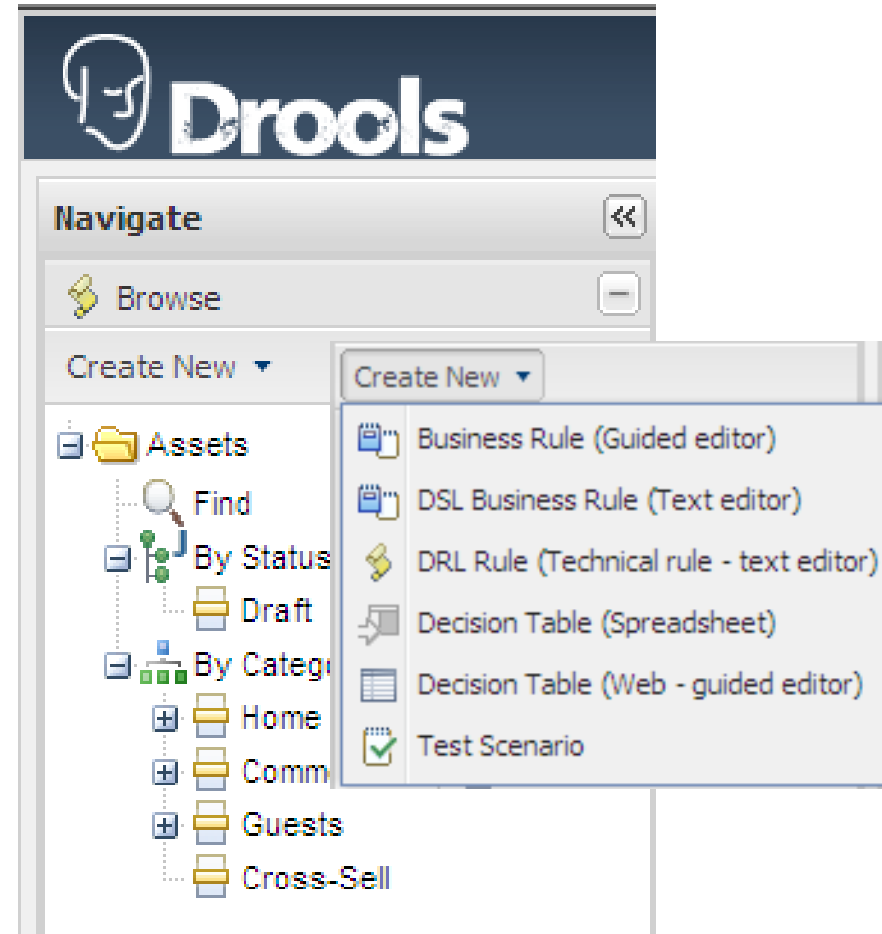
See **Assets**

Each Rule is an **Asset**

Select
Create New – Business Rule

or

Create New – DRL Rule



GUVNOR (DROOLS ON THE WEB)

NEW BUSINESS RULE (GUIDED EDITOR)

Select Cross-Sell
Category

Select Marketing
Package

Enter the Business
Rule **Name**

Press OK

New Business Rule (Guided editor)

New Business Rule (Guided editor)

Name:

Initial category:
 Home Mortgage
 Commercial Mortgage
 Guests
 Cross-Sell

Package: marketing

Initial description:

GVNOR (DROOLS ON THE WEB)

RULE CONDITION AND CONSEQUENCE

The screenshot displays the GUVNOR web interface for configuring a rule. At the top, there is a toolbar with options: Save changes, Copy, Archive, Delete, Change status, and Status: [Draft]. The main area is divided into sections for rule configuration. On the left, the 'WHEN' section is active, showing the rule condition: `when person age more than 18`. Below this, the 'THEN' section is visible with the consequence: `then offers.add(creditCard)`. A 'View source' and 'Validate' button are present. In the center, a tree view shows the 'Assets' folder containing 'Find', 'By Status', 'Draft', and 'By Category'. On the right, a 'Title' field contains the text: `[person age more than 18]`. Below the title field, there is a button labeled 'Add a condition to this rule. ...'. A large text overlay in the center reads: 'Click on the "+" control', with an arrow pointing to the '+' icon in the 'Add a condition to this rule. ...' button. At the bottom, a yellow warning message is displayed: 'Note: No model has been defined. Tip: You will want to import or define a model for this user interface to work!'. Below the warning is a dialog box with an 'OK' button and a list of options: 'There is no ...', 'There exists ...', and 'Any of ...'. An arrow points from the text 'If no data model has been defined, you will see this yellow warning' to the warning message.

Save changes | Copy | Archive | Delete | Change status | Status: [Draft]

WHEN
THEN
(options)

Click on the "+" control

Title: [person age more than 18]
Add a condition to this rule. ...

View source | Validate

```
when person age more than 18  
then offers.add(creditCard)
```

Assets
Find
By Status
Draft
By Category

Add a condition to the rule...

Note: No model has been defined.
Tip: You will want to import or define a model for this user interface to work!

OK

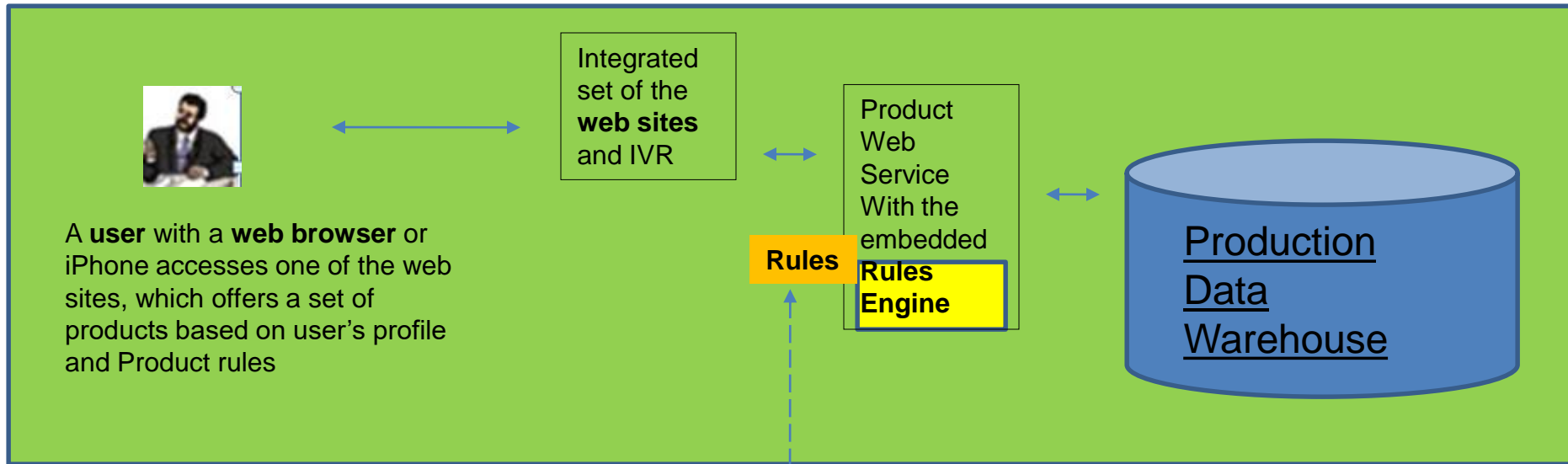
There is no ...
There exists ...
Any of ...

View source | Validate

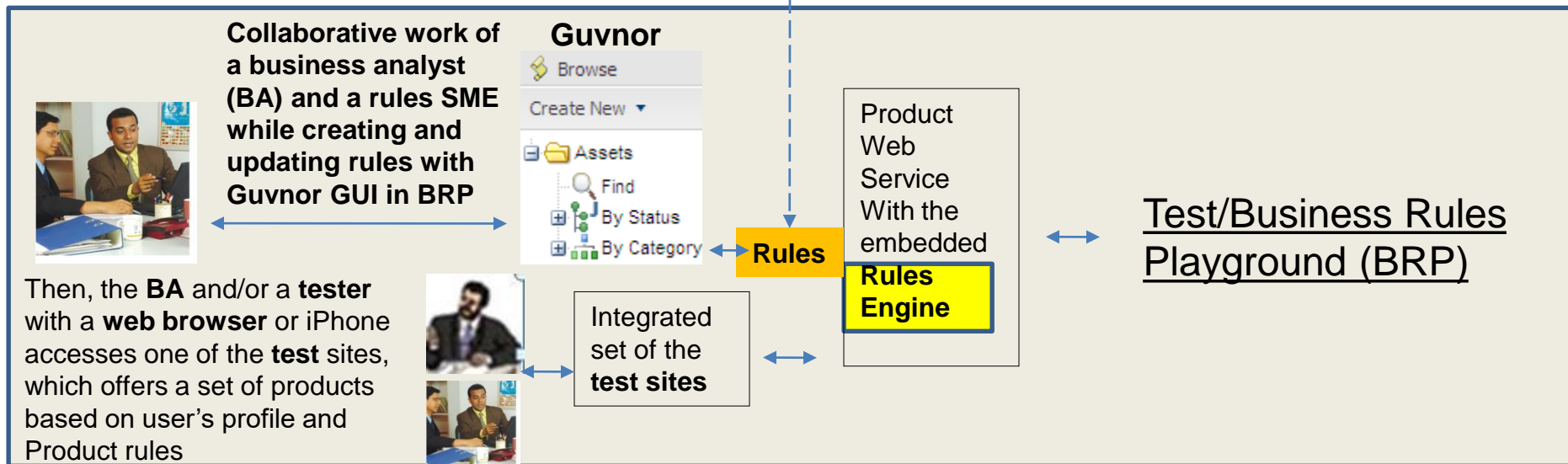
If no data model has been defined, you will see this yellow warning

RULE-BASED DEVELOPMENT

PRODUCT MARKETING EXAMPLE



After successful testing a new or updated Rules file will be moved to production



DATA STRUCTURE IS A MODEL.

INSTANCES OF THE STRUCTURE ARE FACTS.

- ▶ Some models come from an application (i.e. Person).
- ▶ They are loaded into java classes.
- ▶ Classes define a model imported as Java jar file.

- ▶ User may want to complement the application model with additional entities that are used mainly during the reasoning process (i.e. flexible rules).
- ▶ Users can define the model directly to the rules engine.
- ▶ You can create new types directly in DRL.

- ▶ Facts are asserted to run with specific data.
- ▶ JBoss + Spring frameworks connect rules with data.
- ▶ Data and a model must be available before firing rules!

EXAMPLE: PRODUCT RULES

- ▶ When ...
 - The person does have the product
 - The product is on the list
 - The ad type has not been supplied too recently
 - The ad size matches the web page
 - The web site is authorized to sell the product
 - (The person has not applied for the product)
- ▶ Then ...
 - Offer the product with the highest score

INPUT...DECLARE A MODEL

Processing ->

Input ->

Output ->

The screenshot shows the JBoss Guvnor web interface in a Microsoft Internet Explorer browser. The browser's address bar displays the URL `http://nunifijws030:8080/drools-guvnor/org.drools.guvnor`. The page header includes the Drools logo and a welcome message for 'admin' with a '[Sign Out]' link. The main interface is divided into several sections:

- Navigate:** A sidebar menu with options like 'Browse', 'Knowledge Bases', and a 'Create New' dropdown menu. The 'Create New' menu is open, showing options such as 'New Package', 'New Rule', 'Upload POJO Model jar', 'New Declarative Model' (which is highlighted), 'New Function', 'New DSL', 'New RuleFlow', 'New Enumeration', 'New Test Scenario', 'New File', and 'Rebuild all package binaries'.
- Find:** A search section with a 'Name search ...' input field and a 'Search' button. Below it, there are sections for 'Text search ...' and 'Attribute search ...'.
- QA, Package snapshots, Administration:** Additional navigation options at the bottom of the sidebar.

The browser's status bar at the bottom shows 'Local intranet' and a zoom level of '100%'.

SAVE DRL FILE

1. Select your package under Knowledge Bases
2. Click "Show package source" to view DRL
3. Click on "URL for package source" to save

The screenshot displays the JBoss Guvnor web interface in Microsoft Internet Explorer. The browser title is "JBoss Guvnor - Microsoft Internet Explorer provided by Sallie Mae." The address bar shows the URL: <http://nunifjws030:8080/drools-guvnor/org.drools.guvnor.Guvnor/Guvnor.html>. The page content is organized into several sections:

- Navigate:** A sidebar on the left with options like "Browse", "Knowledge Bases", and "Create New". Under "Packages", "Roberts" is selected.
- Find:** A search bar containing "Roberts" with buttons for "Copy", "Rename", and "Archive".
- Configuration:** A section for configuring the package, including "Configuration: Imported types Globals" and "Advanced view".
- Build and validate:** A section with buttons for "Build package" and "Create snapshot for deployment".
- Information and important URLs:** A section containing metadata such as "Last Modified: Monday, July 12, 2010 2:34:49 PM" and "Last contributor: admin". It also includes a "Show package source" button and two URLs: "URL for package source: <http://nunifjws030:8080/drools-guvnor/org.drools.guvnor.Guvnor/package/Roberts/LATEST.drl>" and "URL for package binary: <http://nunifjws030:8080/drools-guvnor/org.drools.guvnor.Guvnor/package/Roberts/LATEST>".

Two blue arrows point to the "Show package source" button and the "URL for package source" link, indicating the steps to save the DRL file.

OBJECTIVES –JAVA CODE EXAMPLE WITH THE RULES

- ▶ Two ways to create Rules
 - Modeling Rules
 - Writing Rules
- ▶ Working sample for “The Rules”
- ▶ Data Discussion

RULES: JAVA METHOD EXAMPLE

```
public void applySimpleRule(ProductRequest req, ArrayList<Product> marketingProducts){
    StatefulKnowledgeSession session;
try {
        KnowledgeBuilder kbuilder = KnowledgeBuilderFactory.newKnowledgeBuilder();
        kbuilder.add( ResourceFactory.newClassPathResource ("SampleRule.drl",
        getClass()),ResourceType.DRL);

        KnowledgeBuilderErrors buildErrors = kbuilder.getErrors();
if(buildErrors.size() > 1){
            log.error("Error encountered when building the rules... ");
}

        KnowledgeBase kbase = KnowledgeBaseFactory.newKnowledgeBase();
        kbase.addKnowledgePackages( kbuilder.getKnowledgePackages() );
        session = kbase.newStatefulKnowledgeSession();
        session.insert(req);
        session.insert(marketingProducts);
        session.insert(log);
        session.fireAllRules();
        session.dispose();
} catch ( Exception e ) {
        e.printStackTrace();
throw new RuntimeException( e );
}
}
```


SUMMARY

- ▶ Why Rule-based development?
- ▶ What are the benefits of Code/Rules separation?
- ▶ What are the steps to create a new rule and what are the challenges?
- ▶ How Code/Rules are connected?
- ▶ How Data/Rules are connected?

APPLYING THE CONVERSATIONAL, SEMANTIC APPROACH TO RULES

- ▶ While technology speaks XML and Web Services, business people prefer natural language (NL)
- ▶ A conversational semantic decision support can bridge these two worlds and provide mapping between NL and services
- ▶ For example, a Business Analyst (BA) writes a line of requirements: "application starts with login."
- ▶ The program would reply "Do you mean the Authentication Service?"



Conversational
Semantic
Decision
Support

Knowledge Warehouse

Common Sense Ontology
Service Ontology
Business Ontology
Industry Ontology



From Idea to Applications
and Business Optimization

Reference: <http://ITofTheFuture.com>

<http://ITUniversity.us>

HOMEWORK

1. Answer Summary Questions
2. Download and Install BRMS from <https://docs.jboss.org/drools/release/>

BUSINESS ARCHITECTURE SANDBOX FOR ENTERPRISE (BASE)

- ▶ BASE* offers to business an easy entrance and a playground to collaborate with IT.
- ▶ BASE helps placing the seeds of semantic technology in the current business ground and helps transitioning to Semantic Cloud Architecture.



Reference: <http://ITofTheFuture.com>

<http://ITUniversity.us>

CREATING A RULE WITH SEMANTIC CHECK

Current decision model for the selected business component is below. [!Run the Component Decision Model](#)

Test rules: [Match](#) | [MisMatch](#) | [Random](#)

RuleFamilyId	RuleFamilyName
8	Determine Person Identity
ConditionDataNames: SSN Person Name Find Best Match or Create Person Address Find Best Match or Create Person Account Status Find Best Match or Create	
ConclusionDataNames: Person Identity Validation Action Find Best Match or Create	
Edit Record Delete Record Conditions and conclusions History Export Disconnect the rule from the component	

[Add more records](#)

Semantic reality check for **Condition data names:**

Known DATA ATTRIBUTE: [SSN](#)

Definition:

No match was found for **PERSON NAME** in the Enterprise Business Model. You still can [CREATE PERSON NAME](#) in your Local Glossary and collaborate with an architect to indicate the *Retrieval and Validation Methods* for the Data Attribute. Meanwhile we recommend you consider suggestions below and [collaborate](#) to map this data attribute to the Enterprise Business Glossary. Another option is to [come back to change the name of the data attribute](#)

The best matches for **PERSON NAME** are:

Type: DATA ATTRIBUTE; Name: [LAST NAME](#)

The resulting screen displays this rule family and automatically produces the links for running and testing the model. The program provides the semantic check for Condition Data Names.

Some data attributes, like SSN, are already in the system, and some are not. The program gives recommendations on mapping the data names to similar data attributes, existing in the system, or creating new attributes on-the-fly.

<http://ITUniversity.us>

DECISION TABLE BASED RULES ENGINE (RE)

[List all rules](#) | [List all rule-based components](#) | Selected Rule Family is [Determine Person Identity](#) (id#8)

The rule family is used by the following components: [ENROLLMENT FOR WEB SERVICES WORKFLOW](#) | [STORE CUSTOMER PROFILE FROM THE WEB](#)

Conditions				Conclusions	
SSN Edit/Delete Data Name	Person Name Edit/Delete Data Name	Person Address Edit/Delete Data Name	Person Account Status Edit/Delete Data Name	Person Identity Validation Action Edit/Delete Data Name	
Existing Value true Edit Condition	Existing Value true Edit Condition	Existing Value false Edit Condition	Valid Value In Good Standing Edit Condition	Message: Existing account is valid for practical purposes Edit Conclusion	Delete Rule
Existing Value true Edit Condition	Existing Value true Edit Condition	Existing Value false Edit Condition	Valid Value Not Valid Edit Condition	Message: accept new customer profile instead of existing one. Edit Conclusion	Delete Rule
Add Condition	Add Condition	Add Condition	Add Condition	Add Conclusion	

The rules are present as the rows and columns in the **decision table**. Each row is a separate rule, which includes several **conditions** and a **conclusion**.

BASE uses semantic approach to connect the rules and data (a common RE problem). Read "data know how to handle data" here: <http://ITofTheFuture.com>