Руководство по настройке и разработке утилиты inav-backup

Утилита полностью переработана. Теперь она подходит для бэкапа абсолютно любой БД и любой таблицы, достаточно указать правильные настройки в configuration.xml.

Дополнительные параметры

**Дополнительные** параметры в теге <table> для настройки бэкапа в configuration.xml:  
(все параметры влияют только на этап восстановления, и не влияют на этап backup)

|  |  |  |
| --- | --- | --- |
| Атрибут | Возможное значение (значение по умолчанию) | Описание |
| clear | true/false (false) | Если true, то перед восстановлением выполнится truncate table (т.е. таблица очистится со сбросом счетчика identity) |
| update | Названия колонок (через “;”) в таблице, по которым будет производиться обновление записей | Если указан, то сгенерируется запрос вида:  Update <table> set <column1> = <value1>, … where <column\_update1> = <valueN> and ….  If @@rowcount = 0  Insert …. |
| updateOnly | true/false (false) | Если true, то в запросе не будет insert (будет только update) |
| identityInsert | true/false (false) | Если true, то в запросы (при восстановлении) будут попадать колонки с identity. При этом сначала выставится флаг identity\_insert. |
| skipRestore | true/false (false) | Дополнительный тригер, запрещающий программе восстанавливать эту таблицу (если для какой-то таблицы нет сохраненного backup-а, то эта таблица автоматически будет пропущена) |
| ignore | Названия колонок (через “;”) в таблице, которые необходимо игнорировать | Игнорирование колонок; их значение останется тем же, что и до восстановления. |

Поддержка и расширение

Сейчас утилитой поддерживаются 3 типа данных: int, image, и все остальное, которое преобразовывается как обычная строка.

За поддержку типов отвечает класс *ru.sbt.utils.backup.conversion.ConvertorFactory*. Чтобы добавить новый тип надо создать класс от интерфейса *ru.sbt.utils.backup.conversion.IConvertor* и добавить его в ConvertorFactory. Задача класса IConvertor –получать значение из ResultSet и преобразовывать его в строку; и наоборот, уметь получать исходное значение из строки и записывать его в PreparedStatement.

ConvertorFactory для каждого типа выдает нужный конвертер. Если такой не задан, он выдает DefaultConvertor, который получает строку из ResultSet или записывает значение как строку в PreparedStatement, не изменяя его.

Требуется расширять утилитку, путем добавления новых конверторов, при необходимости. Для большинства типов DefaultConvertor вполне хватает.

Также есть некоторые типы данных, которые нельзя использовать в сравнениях. Такими являются, например, image и varbinary. Они добавлены в *ru.sbt.utils.backup.db.QueryManager* в сет **incompatible\_in\_the\_equal\_to\_operator** . требуется добавлять туда типы, которые нельзя использовать для сравнения (по мере их обнаружения – я не могу тестировать все возможные типы).