Spring и базы данных

№ урока: 11 **Курс:** Основы Java EE

Средства обучения: InteliJ Idea, MySQL Workbench

Обзор, цель и назначение урока

Spring & JDBC. Сравнение с чистым JDBC. CRUD операции. Транзакции

Изучив материал данного занятия, учащийся сможет:

- Использовать упрощенный уровень JDBC с помощью Spring.
- Выполнять транзакции с помощью Spring & JDBC.

Содержание урока

1. Введение в Spring Data.
2. Преимущества Spring&JDBC по сравнению с чистым JDBC.
3. Создание структуры DAO.
4. Пример выборки.
5. Пример вставки.
6. Пример транзакций.
7. Типы транзакций. Свойство Propagation.
8. Уровни изоляции транзакций. Свойство Isolation.
9. Аспекты для тразакций.

Резюме

- **Доступ к данным**. Spring предоставляет свой слой доступа к базам данных и поддерживает все популярные СУБД: JDBC, iBatis / MyBatis, Hibernate, JDO, JPA, Oracle TopLink, Apache OJB, Apache Cayenne и т. д.
  Для всех этих фреймворков Spring предоставляет такие особенности:
  - **Управление ресурсами** — автоматическое получение и освобождение ресурсов базы данных
  - **Обработка исключений** — перевод исключений при доступе к данным в исключения Spring-а
  - **Транзакционность** — прозрачные транзакции в операциях с данными
  - **Распаковка ресурсов** — получение объектов базы данных из пула соединений
  - **Абстракция** для обработки BLOB и CLOB

  Фреймворк управления транзакциями в Spring привносит механизм абстракций для платформы Java. Основные возможности этих абстракций:
  - работа с локальными и глобальными транзакциями
  - работа с вложенными транзакциями
  - работа с точками сохранения в транзакциях
- **Транзакция** – это группа последовательно выполняемых операторов SQL, которые либо должны быть выполнены все, либо не должен быть выполнен ни один из них.
- **Откат транзакции** – это действие, обеспечивающее аннулирование всех изменений данных, которые были сделаны в теле текущей незавершенной транзакции.
- **Точки сохранения** – это именованные метки, которые разбивают транзакцию на этапы, позволяя возвращаться к выполнению любого из этапов, указав к какой метке нужно выполнить возврат.
- **Уровень изолированности транзакций** — условное значение, определяющее, в какой мере в результате выполнения логически параллельных транзакций в СУБД допускается получение несогласованных данных. Шкала уровней изолированности транзакций содержит ряд значений,

CyberBionic systematics

CyberBionic Systematics ® 2018
19 Eugene Sverstyuk Str.,5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

проранжированных от наинизшего до наивысшего; более высокий уровень изолированности соответствует лучшей согласованности данных, но его использование может снижать количество физически параллельно выполняемых транзакций. И наоборот, более низкий уровень изолированности позволяет выполнять больше параллельных транзакций, но снижает точность данных. Таким образом, выбирая используемый уровень изолированности транзакций, разработчик информационной системы в определённой мере обеспечивает выбор между скоростью работы и обеспечением гарантированной согласованности получаемых из системы данных.

- При параллельном выполнении транзакций возможны следующие проблемы:
  - потерянное обновление (англ. lost update) — при одновременном изменении одного блока данных разными транзакциями одно из изменений теряется;
  - «грязное» чтение (англ. dirty read) — чтение данных, добавленных или изменённых транзакцией, которая впоследствии не подтвердится (откатится);
  - неповторяющееся чтение (англ. non-repeatable read) — при повторном чтении в рамках одной транзакции ранее прочитанные данные оказываются изменёнными;
  - фантомное чтение (англ. phantom reads) — одна транзакция в ходе своего выполнения несколько раз выбирает множество строк по одним и тем же критериям. Другая транзакция в интервалах между этими выборками добавляет или удаляет строки, или изменяет столбцы некоторых строк, используемых в критериях выборки первой транзакции, и успешно заканчивается. В результате получится, что одни и те же выборки в первой транзакции дают разные множества строк.
- Под «**уровнем изоляции транзакций**» понимается степень обеспечиваемой внутренними механизмами СУБД (то есть не требующей специального программирования) защиты от всех или некоторых видов вышеперечисленных несогласованностей данных, возникающих при параллельном выполнении транзакций. Стандарт SQL-92 определяет шкалу из четырёх уровней изоляции: Read uncommitted, Read committed, Repeatable read, Serializable. Первый из них является самым слабым, последний — самым сильным, каждый последующий включает в себя все предыдущие.
- **Read uncommitted (чтение незафиксированных данных)**
  Низший (первый) уровень изоляции. Он гарантирует только отсутствие потерянных обновлений. Если несколько параллельных транзакций пытаются изменять одну и ту же строку таблицы, то в окончательном варианте строка будет иметь значение, определенное всем набором успешно выполненных транзакций. При этом возможно считывание не только логически несогласованных данных, но и данных, изменения которых ещё не зафиксированы.
  Типичный способ реализации данного уровня изоляции — блокировка данных на время выполнения команды изменения, что гарантирует, что команды изменения одних и тех же строк, запущенные параллельно, фактически выполнятся последовательно, и ни одно из изменений не потеряется. Транзакции, выполняющие только чтение, при данном уровне изоляции никогда не блокируются.
- **Read committed (чтение фиксированных данных)**
  Большинство промышленных СУБД, в частности, Microsoft SQL Server, PostgreSQL и Oracle, по умолчанию используют именно этот уровень. На этом уровне обеспечивается защита от чернового, «грязного» чтения, тем не менее, в процессе работы одной транзакции другая может быть успешно завершена и сделанные ею изменения зафиксированы. В итоге первая транзакция будет работать с другим набором данных.
  Реализация завершённого чтения может основываться на одном из двух подходов: блокировании или версионности.
  Блокирование читаемых и изменяемых данных.
  Заключается в том, что пишущая транзакция блокирует изменяемые данные для читающих транзакций, работающих на уровне read committed или более высоком, до своего завершения, препятствуя, таким образом, «грязному» чтению, а данные, блокируемые читающей транзакцией, освобождаются сразу после завершения операции SELECT (таким образом, ситуация «неповторяющегося чтения» может возникать на данном уровне изоляции).
  Сохранение нескольких версий параллельно изменяемых строк.
  При каждом изменении строки СУБД создаёт новую версию этой строки, с которой продолжает работать изменившая данные транзакция, в то время как любой другой «читающей» транзакции возвращается последняя зафиксированная версия. Преимущество такого подхода в

CyberBionic systematics
CyberBionic Systematics ® 2018
19 Eugene Sverstyuk Str.,5 floor
Kyiv, Ukraine
Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

том, что он обеспечивает большую скорость, так как предотвращает блокировки. Однако он требует, по сравнению с первым, существенно большего расхода оперативной памяти, которая тратится на хранение версий строк. Кроме того, при параллельном изменении данных несколькими транзакциями может создаться ситуация, когда несколько параллельных транзакций произведут несогласованные изменения одних и тех же данных (поскольку блокировки отсутствуют, ничто не помешает это сделать). Тогда та транзакция, которая зафиксируется первой, сохранит свои изменения в основной БД, а остальные параллельные транзакции окажется невозможно зафиксировать (так как это приведёт к потере обновления первой транзакции). Единственное, что может в такой ситуации СУБД — это откатить остальные транзакции и выдать сообщение об ошибке «Запись уже изменена».

Конкретный способ реализации выбирается разработчиками СУБД, а в ряде случаев может настраиваться. Так, по умолчанию MS SQL использует блокировки, но (в версии 2005 и выше) при установке параметра READ_COMMITTED_SNAPSHOT базы данных переходит на стратегию версионности, Oracle исходно работает только по версионной схеме. В Informix можно предотвратить конфликты между читающими и пишущими транзакциями, установив параметр конфигурации USELASTCOMMITTED (начиная с версии 11.1), при этом читающая транзакция будет получать последние подтвержденные данные.

- **Repeatable read (повторяемость чтения)**

  Уровень, при котором читающая транзакция «не видит» изменения данных, которые были ею ранее прочитаны. При этом никакая другая транзакция не может изменять данные, читаемые текущей транзакцией, пока та не окончена.

  Блокировки в разделяющем режиме применяются ко всем данным, считываемым любой инструкцией транзакции, и сохраняются до её завершения. Это запрещает другим транзакциям изменять строки, которые были считаны незавершённой транзакцией. Однако другие транзакции могут вставлять новые строки, соответствующие условиям поиска инструкций, содержащихся в текущей транзакции. При повторном запуске инструкции текущей транзакцией будут извлечены новые строки, что приведёт к фантомному чтению. Учитывая то, что разделяющие блокировки сохраняются до завершения транзакции, а не снимаются в конце каждой инструкции, степень параллелизма ниже, чем при уровне изоляции READ COMMITTED. Поэтому пользоваться данным и более высокими уровнями транзакций без необходимости обычно не рекомендуется.
- **Serializable (упорядочиваемость)**

  Самый высокий уровень изолированности; транзакции полностью изолируются друг от друга, каждая выполняется так, как будто параллельных транзакций не существует. Только на этом уровне параллельные транзакции не подвержены эффекту «фантомного чтения».
- **Поддержка изоляции транзакций в реальных СУБД**

  СУБД, обеспечивающие транзакционность, не всегда поддерживают все четыре уровня, а также могут вводить дополнительные. Возможны также различные нюансы в обеспечении изоляции.

  Так, Oracle в принципе не поддерживает нулевой уровень, так как его реализация транзакций исключает «грязные чтения», и формально не позволяет устанавливать уровень Repeatable read, то есть поддерживает только Read committed (по умолчанию) и Serializable. При этом на уровне отдельных команд он, фактически, гарантирует повторяемость чтения (если команда SELECT в первой транзакции выбирает из базы набор строк, и в это время параллельная вторая транзакция изменяет какие-то из этих строк, то результирующий набор, полученный первой транзакцией, будет содержать неизменённые строки, как будто второй транзакции не было). Также Oracle поддерживает так называемые READ-ONLY транзакции, которые соответствуют Serializable, но при этом не могут сами изменять данные.

  Microsoft SQL Server поддерживает все четыре стандартных уровня изоляции транзакций, а дополнительно — уровень SNAPSHOT, находящийся между Repeatable read и Serialized. Транзакция, работающая на данном уровне, видит только те изменения данных, которые были зафиксированы до её запуска, а также изменения, внесённые ею самой, то есть ведёт себя так, как будто получила при запуске моментальный снимок данных БД и работает с ним.

Закрепление материала

- Что такое Spring Data?

CyberBionic systematics

CyberBionic Systematics ® 2018
19 Eugene Sverstyuk Str.,5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

- В чем преимущество Spring & JDBC над чистым JDBC?
- Что такое транзакция?
- Зачем нужно свойство Propagation?
- Какие есть проблемы параллельного доступа транзакций?
- Какие есть уровни изоляции транзакций?

Дополнительное задание

Задание
Создать таблицу для хранения MP3(id, name, author_id) и Author(id, name), таблицы связанные между собой. Прочитать главы 12.1, 12.2. Создать интерфейс DAO. Добавить методы и реализовать добавление коллекции песен и удаление песни по id.

Самостоятельная деятельность учащегося

Задание 1
Выучите основные понятия, рассмотренные на уроке.
Прочитать документацию pdf главу 12.1

Задание 2
Реализовать метод, который возвращает сгруппированные значения: для каждого автора – его имя и количество песен (Group by)

Задание 3
Прочитать раздел 12.4
Создать метод для обновления записей из списка MP3

Задание 4
Реализовать поочередную вставку записей в таблицы Author и MP3 для методов: insert, insertList из доп. задания

Рекомендуемые ресурсы

Spring docs pdf
https://docs.spring.io/spring/docs/3.0.x/spring-framework-reference/pdf/spring-framework-reference.pdf

Пул соединений JNDI
https://commons.apache.org/proper/commons-dbcp/

Использование транзакций в обычном JDBC
http://www.mkyong.com/jdbc/jdbc-transaction-example/
http://tutorials.jenkov.com/jdbc/transaction.html
https://docs.oracle.com/javase/tutorial/jdbc/basics/transactions.html

Propagation
http://www.byteslounge.com/tutorials/spring-transaction-propagation-tutorial
https://www.ibm.com/developerworks/ru/library/j-ts1/

CyberBionic systematics
CyberBionic Systematics ® 2018
19 Eugene Sverstyuk Str.,5 floor
Kyiv, Ukraine
Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com