when you have to read exception messages. Note that the hbm2ddl exporter considers constraint names only for foreign keys that have been set on the noninverse side of a bidirectional association mapping.

Foreign key constraints also have features in SQL that your legacy schema may already utilize. Instead of immediately rejecting a modification of data that would violate a foreign key constraint, an SQL database can CASCADE the change to the referencing rows. For example, if a row that is considered a parent is deleted, all child rows with a foreign key constraint on the primary key of the parent row may be deleted as well. If you have or want to use these database-level cascading options, enable them in your foreign key mapping:

```
<class name="Item" table="ITEM">
    ...
    <set name="bids" cascade="save-update, delete">
        <key column="ITEM_ID" on-delete="cascade"/>
        <one-to-many class="Bid"/>
    </set>

</class>
```

Hibernate now creates and relies on a database-level ON CASCADE DELETE option of the foreign key constraint, instead of executing many individual DELETE statements when an Item instance is deleted and all bids have to be removed. Be aware that this feature bypasses Hibernate's usual optimistic locking strategy for versioned data!

Finally, unrelated to integrity rules translated from business logic, database performance optimization is also part of your typical DDL customization effort.

8.3.6 *Creating indexes*

Indexes are a key feature when optimizing the performance of a database application. The query optimizer in a database-management system can use indexes to avoid excessive scans of the data tables. Because they're relevant only in the physical implementation of a database, indexes aren't part of the SQL standard, and the DDL and available indexing options are specific for a particular product. However, the most common DDL for typical indexes can be embedded in a Hibernate mapping (that is, without the generic <database-object> element).

Many queries in CaveatEmptor will probably involve the endDate property of an auction Item. You can speed up these queries by creating an index for the column of this property:

```
<property   name="endDate"
            column="END_DATE"
```