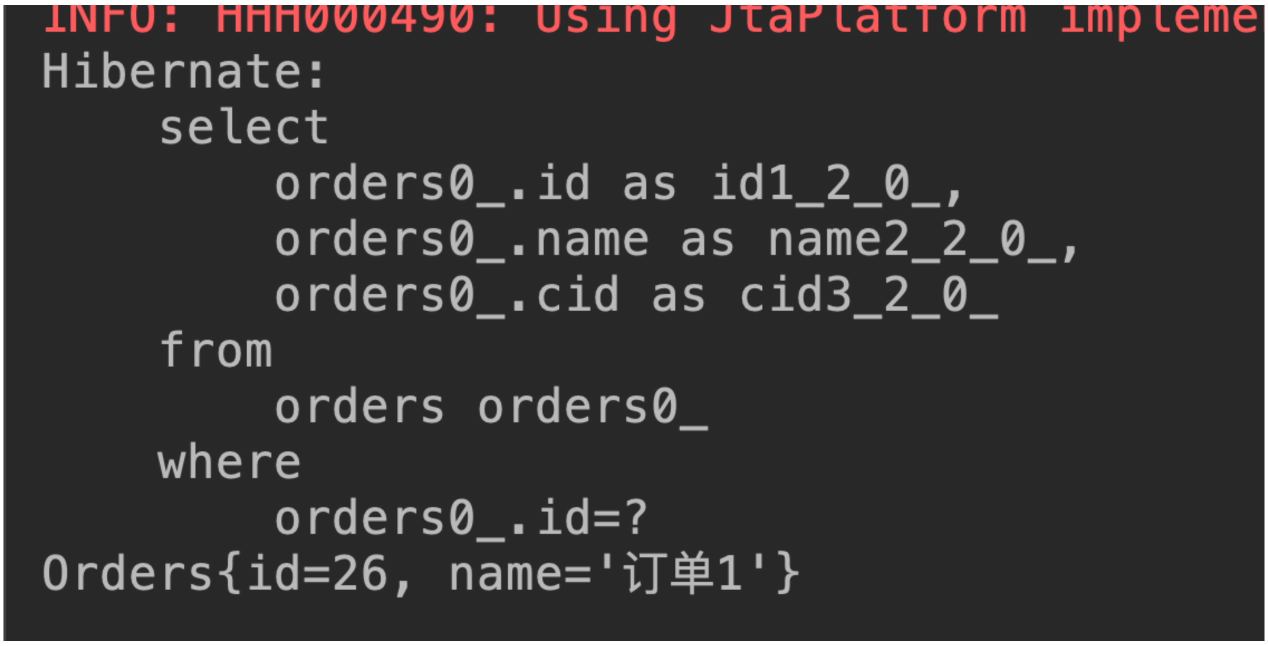
也可以通过 Orders 来设置 Customer 的延迟加载，orders.hbm.xml 中进行设置。

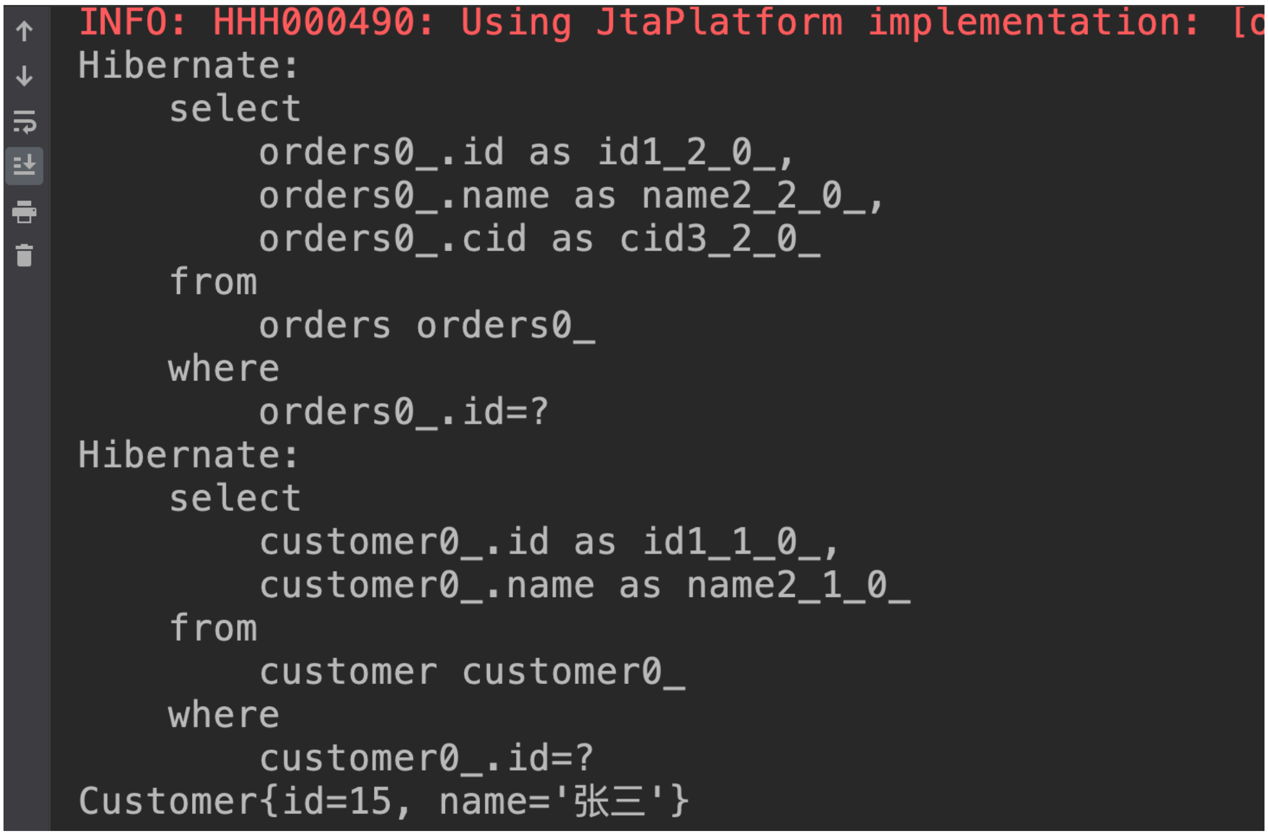
```
<many-to-one name="customer" class="com.southwind.entity.Customer"
column="cid" lazy="proxy"></many-to-one>
```

```
package com.southwind.test;

import com.southwind.entity.Orders;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class Test5 {
    public static void main(String[] args) {
        //创建 Configuration
        Configuration configuration = new
Configuration().configure("hibernate.xml");
        //获取 SessionFactory
        SessionFactory sessionFactory = configuration.buildSessionFactory();
        //获取 Session
        Session session = sessionFactory.openSession();

        Orders orders = session.get(Orders.class,26);
        System.out.println(orders);

        session.close();
    }
}
```



```
INFO: HHH000490: Using JtaPlatform impleme
Hibernate:
    select
        orders0_.id as id1_2_0_,
        orders0_.name as name2_2_0_,
        orders0_.cid as cid3_2_0_
    from
        orders orders0_
    where
        orders0_.id=?
Orders{id=26, name='订单1'}
```

```java
package com.southwind.test;

import com.southwind.entity.Orders;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class Test5 {
    public static void main(String[] args) {
        //创建 Configuration
        Configuration configuration = new
Configuration().configure("hibernate.xml");
        //获取 SessionFactory
        SessionFactory sessionFactory = configuration.buildSessionFactory();
        //获取 Session
        Session session = sessionFactory.openSession();

        Orders orders = session.get(Orders.class,26);
        System.out.println(orders.getCustomer());

        session.close();
    }
}
```



```
INFO: HHH000490: Using JtaPlatform implementation: [o
Hibernate: 
    select
        orders0_.id as id1_2_0_,
        orders0_.name as name2_2_0_,
        orders0_.cid as cid3_2_0_ 
    from
        orders orders0_ 
    where
        orders0_.id=?
Hibernate: 
    select
        customer0_.id as id1_1_0_,
        customer0_.name as name2_1_0_ 
    from
        customer customer0_ 
    where
        customer0_.id=?
Customer{id=15, name='张三'}
```

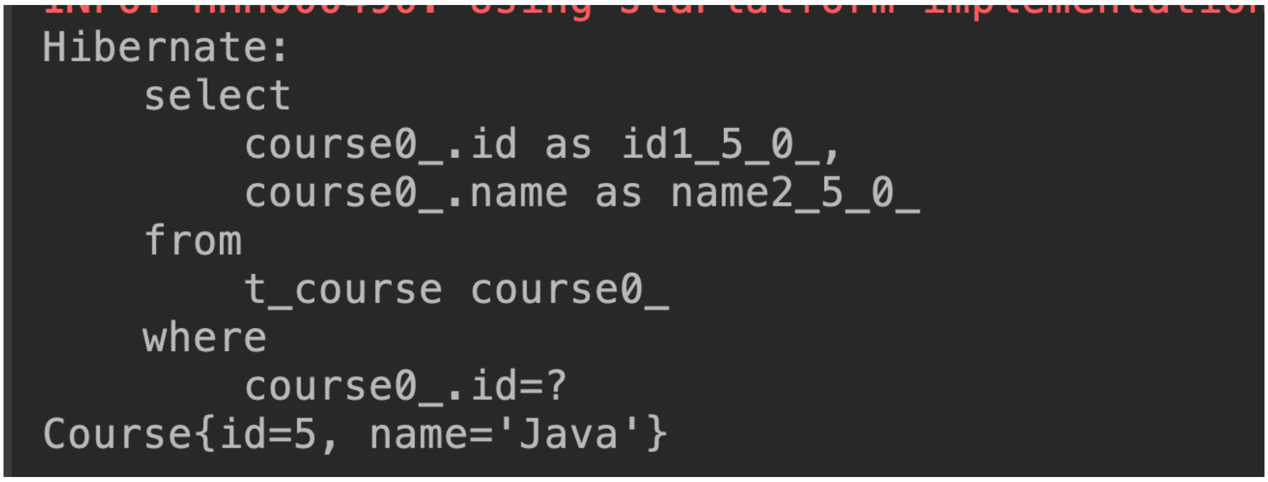
no-proxy：当调用方法需要访问 customer 的成员变量时，发送 SQL 语句查询 Customer，否则不查询。

proxy：无论调用方法是否需要访问 customer 的成员变量，都会发送 SQL 语句查询 Customer。

多对多

查询 Course，加载对应的 Account，默认延迟加载开启。

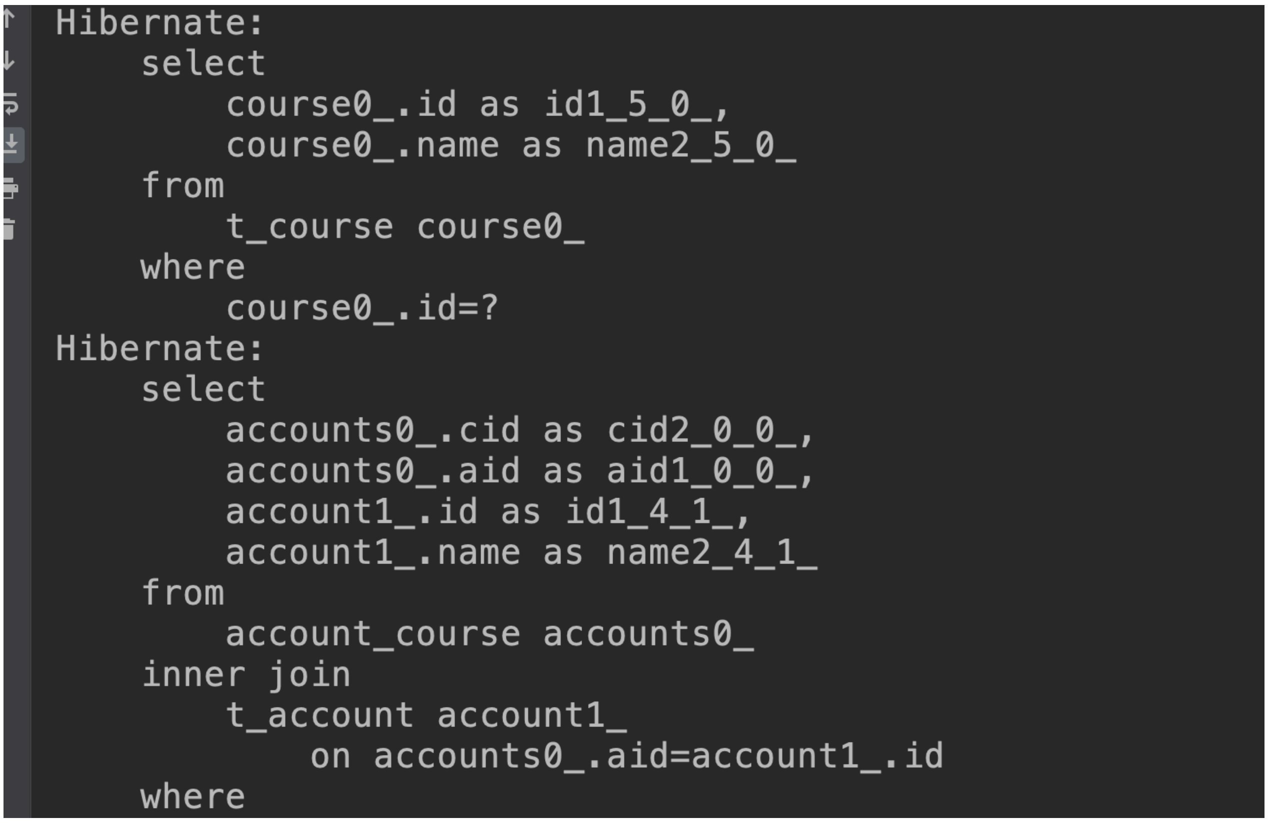
```java
package com.southwind.test;

import com.southwind.entity.Course;
import com.southwind.entity.Orders;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class Test6 {
    public static void main(String[] args) {
        //创建 Configuration
        Configuration configuration = new
Configuration().configure("hibernate.xml");
        //获取 SessionFactory
        SessionFactory sessionFactory = configuration.buildSessionFactory();
        //获取 Session
        Session session = sessionFactory.openSession();

        Course course = session.get(Course.class,5);
        System.out.println(course);

        session.close();
    }
}
```



```
Hibernate:
    select
        course0_.id as id1_5_0_,
        course0_.name as name2_5_0_
    from
        t_course course0_
    where
        course0_.id=?
Course{id=5, name='Java'}
```

```java
package com.southwind.test;

import com.southwind.entity.Course;
import com.southwind.entity.Orders;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;
```

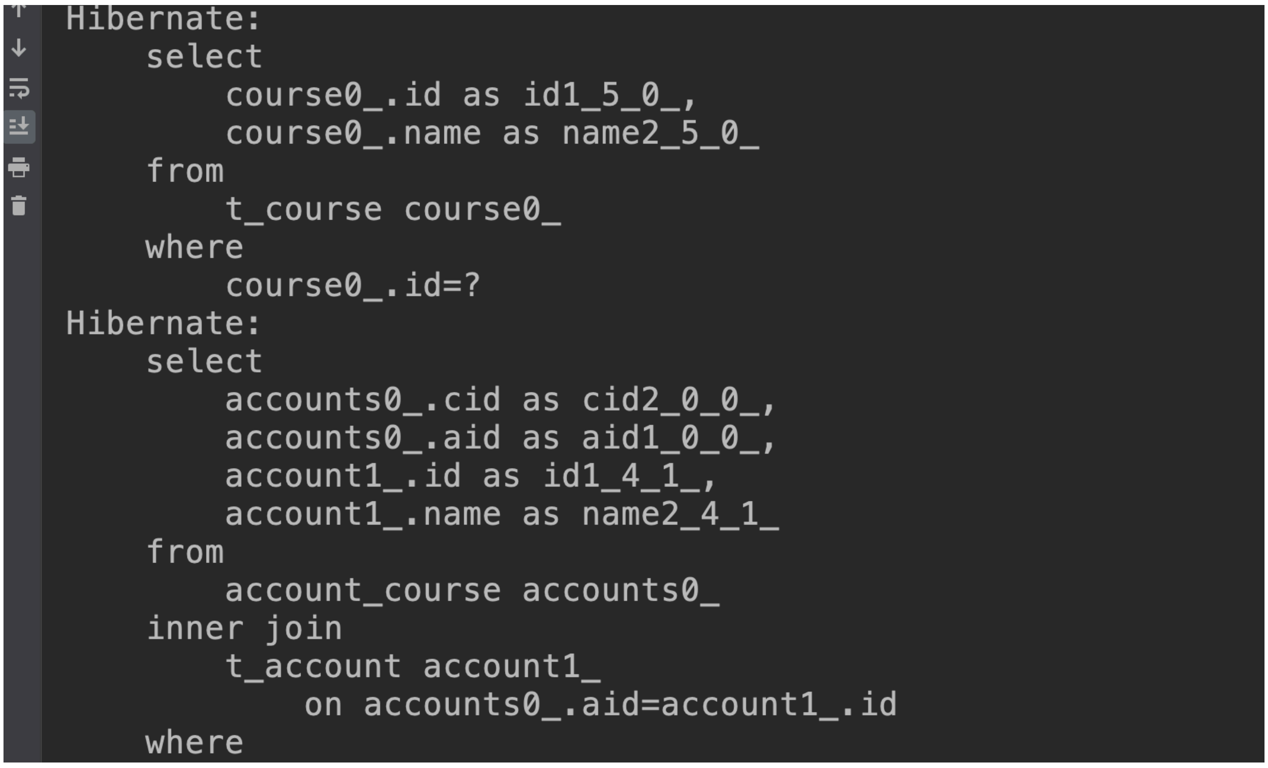
```
public class Test6 {
    public static void main(String[] args) {
        //创建 Configuration
        Configuration configuration = new
Configuration().configure("hibernate.xml");
        //获取 SessionFactory
        SessionFactory sessionFactory = configuration.buildSessionFactory();
        //获取 Session
        Session session = sessionFactory.openSession();

        Course course = session.get(Course.class,5);
        System.out.println(course.getAccounts());

        session.close();
    }
}
```



```
Hibernate:
    select
        course0_.id as id1_5_0_,
        course0_.name as name2_5_0_
    from
        t_course course0_
    where
        course0_.id=?
Hibernate:
    select
        accounts0_.cid as cid2_0_0_,
        accounts0_.aid as aid1_0_0_,
        account1_.id as id1_4_1_,
        account1_.name as name2_4_1_
    from
        account_course accounts0_
    inner join
        t_account account1_
            on accounts0_.aid=account1_.id
    where
```

```
<set name="accounts" table="account_course" lazy="false">
    <key column="cid"></key>
    <many-to-many class="com.southwind.entity.Account" column="aid"></many-to-
many>
</set>
```

```
package com.southwind.test;

import com.southwind.entity.Course;
import com.southwind.entity.Orders;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class Test6 {
    public static void main(String[] args) {
        //创建 Configuration
        Configuration configuration = new
Configuration().configure("hibernate.xml");
        //获取 SessionFactory
        SessionFactory sessionFactory = configuration.buildSessionFactory();
        //获取 Session
        Session session = sessionFactory.openSession();

        Course course = session.get(Course.class,5);
        System.out.println(course);

        session.close();
    }
}
```

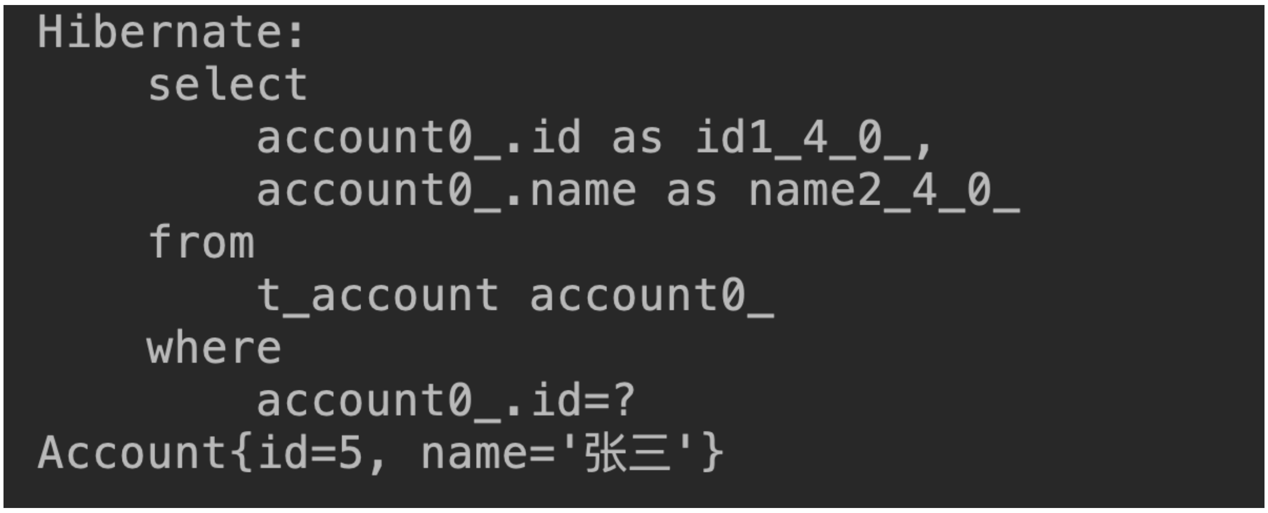


```
Hibernate:
    select
        course0_.id as id1_5_0_,
        course0_.name as name2_5_0_
    from
        t_course course0_
    where
        course0_.id=?
Hibernate:
    select
        accounts0_.cid as cid2_0_0_,
        accounts0_.aid as aid1_0_0_,
        account1_.id as id1_4_1_,
        account1_.name as name2_4_1_
    from
        account_course accounts0_
    inner join
        t_account account1_
            on accounts0_.aid=account1_.id
    where
```

```
<set name="courses" table="account_course" lazy="true">
    <key column="aid"></key>
    <many-to-many class="com.southwind.entity.Course" column="cid"></many-to-
many>
</set>
```

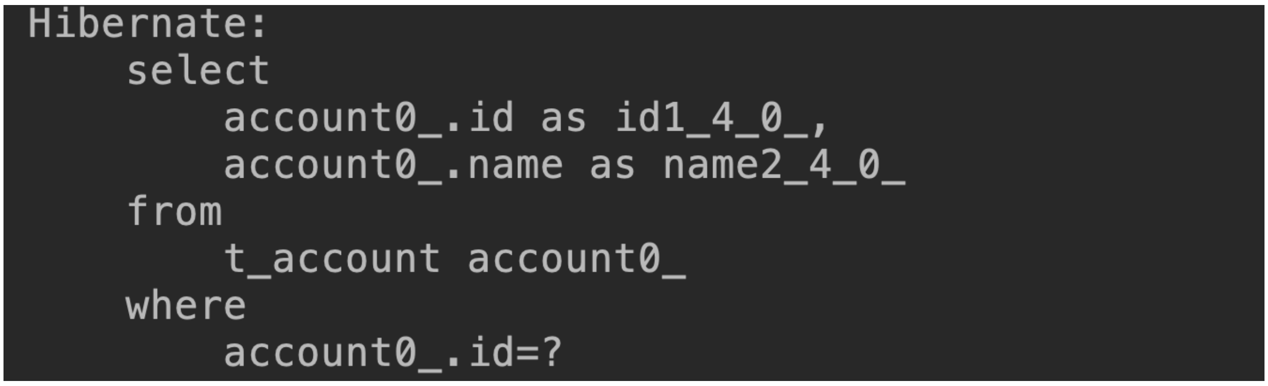
```
package com.southwind.test;
```

```java
import com.southwind.entity.Account;
import com.southwind.entity.Course;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class Test7 {
    public static void main(String[] args) {
        //创建 Configuration
        Configuration configuration = new
Configuration().configure("hibernate.xml");
        //获取 SessionFactory
        SessionFactory sessionFactory = configuration.buildSessionFactory();
        //获取 Session
        Session session = sessionFactory.openSession();

        Account account = session.get(Account.class,5);
        System.out.println(account);

        session.close();
    }
}
```

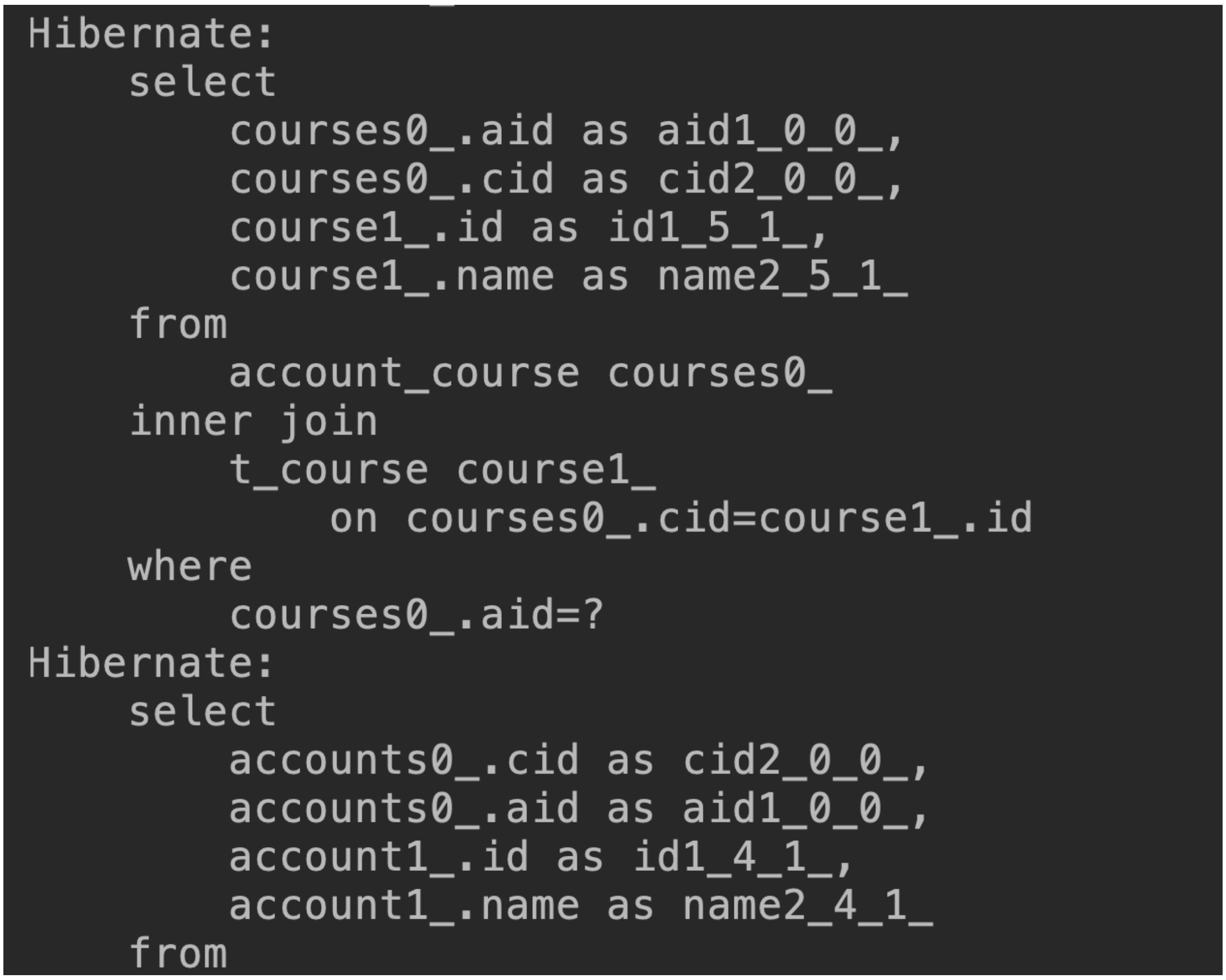


```
Hibernate:
    select
        account0_.id as id1_4_0_,
        account0_.name as name2_4_0_
    from
        t_account account0_
    where
        account0_.id=?
Account{id=5, name='张三'}
```

```java
package com.southwind.test;

import com.southwind.entity.Account;
import com.southwind.entity.Course;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class Test7 {
    public static void main(String[] args) {
        //创建 Configuration
        Configuration configuration = new
Configuration().configure("hibernate.xml");
        //获取 SessionFactory
        SessionFactory sessionFactory = configuration.buildSessionFactory();
        //获取 Session
        Session session = sessionFactory.openSession();

        Account account = session.get(Account.class,5);
        System.out.println(account.getCourses());

        session.close();
    }
}
```



```
Hibernate: 
    select
        account0_.id as id1_4_0_,
        account0_.name as name2_4_0_ 
    from
        t_account account0_ 
    where
        account0_.id=?
```

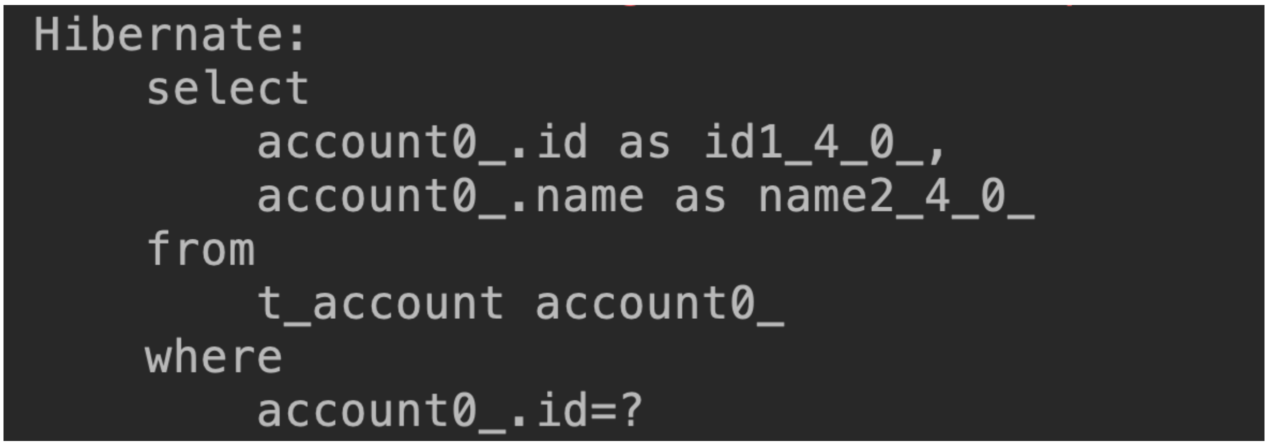


```
Hibernate: 
    select
        courses0_.aid as aid1_0_0_,
        courses0_.cid as cid2_0_0_,
        course1_.id as id1_5_1_,
        course1_.name as name2_5_1_ 
    from
        account_course courses0_ 
    inner join
        t_course course1_ 
            on courses0_.cid=course1_.id 
    where
        courses0_.aid=?
Hibernate: 
    select
        accounts0_.cid as cid2_0_0_,
        accounts0_.aid as aid1_0_0_,
        account1_.id as id1_4_1_,
        account1_.name as name2_4_1_ 
    from
```

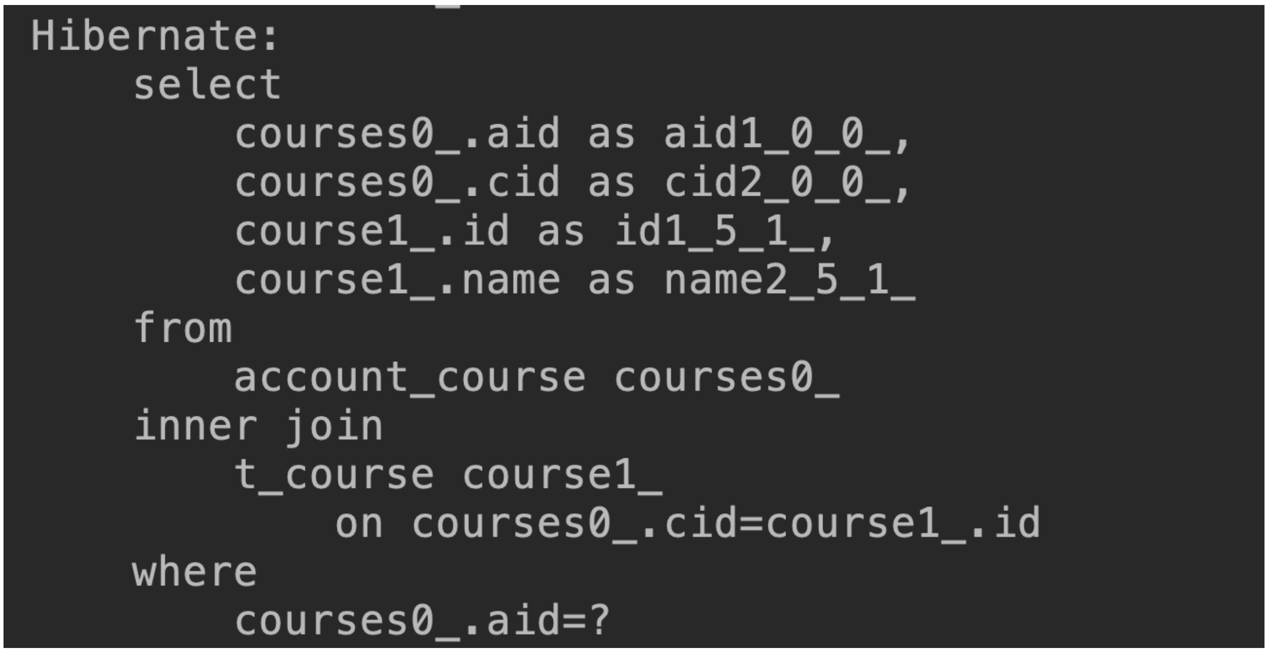
```
        account_course accounts0_
    inner join
        t_account account1_
            on accounts0_.aid=account1_.id
    where
        accounts0_.cid=?
[Course{id=5, name='Java'}]
```

```
<set name="courses" table="account_course" lazy="false">
    <key column="aid"></key>
    <many-to-many class="com.southwind.entity.Course" column="cid"></many-to-
many>
</set>
```

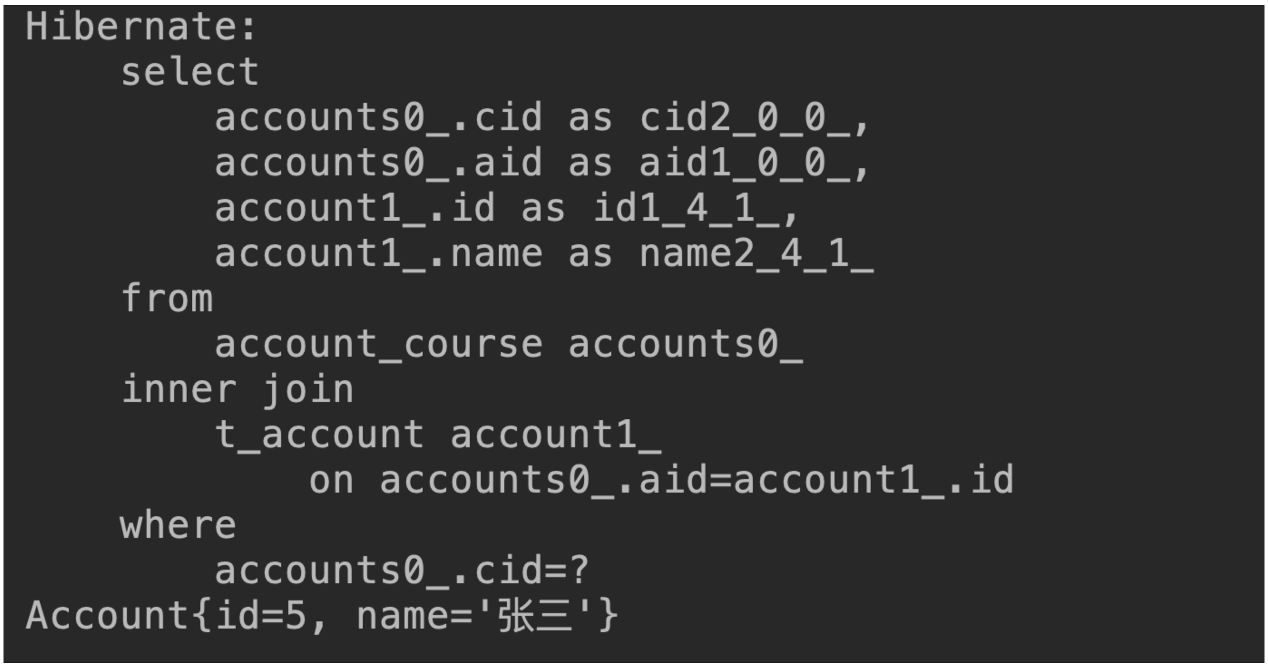
```
package com.southwind.test;

import com.southwind.entity.Account;
import com.southwind.entity.Course;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class Test7 {
    public static void main(String[] args) {
        //创建 Configuration
        Configuration configuration = new
Configuration().configure("hibernate.xml");
        //获取 SessionFactory
        SessionFactory sessionFactory = configuration.buildSessionFactory();
        //获取 Session
        Session session = sessionFactory.openSession();

        Account account = session.get(Account.class,5);
        System.out.println(account);

        session.close();
    }
}
```



```
Hibernate:
    select
        account0_.id as id1_4_0_,
        account0_.name as name2_4_0_
    from
        t_account account0_
    where
        account0_.id=?
```



```
Hibernate:
    select
        courses0_.aid as aid1_0_0_,
        courses0_.cid as cid2_0_0_,
        course1_.id as id1_5_1_,
        course1_.name as name2_5_1_
    from
        account_course courses0_
    inner join
        t_course course1_
            on courses0_.cid=course1_.id
    where
        courses0_.aid=?
```



```
Hibernate:
    select
        accounts0_.cid as cid2_0_0_,
        accounts0_.aid as aid1_0_0_,
        account1_.id as id1_4_1_,
        account1_.name as name2_4_1_
    from
        account_course accounts0_
    inner join
        t_account account1_
            on accounts0_.aid=account1_.id
    where
        accounts0_.cid=?
Account{id=5, name='张三'}
```

Hibernate 配置文件

- hibernate.xml
- hbm.xml

Hibernate.xml

hibernate.xml 配置 Hibernate 的全局环境。

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
        "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
        "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">
<hibernate-configuration>

    <session-factory>
        <!-- 数据源配置 -->
        <property name="connection.username">root</property>
        <property name="connection.password">root</property>
        <property
name="connection.driver_class">com.mysql.cj.jdbc.Driver</property>
        <property name="connection.url">jdbc:mysql://localhost:3306/test?
useUnicode=true&amp;characterEncoding=UTF-8</property>

        <!-- C3P0 -->
        <property name="hibernate.c3p0.acquire_increment">10</property>
        <property name="hibernate.c3p0.idle_test_period">10000</property>
        <property name="hibernate.c3p0.timeout">5000</property>
        <property name="hibernate.c3p0.max_size">30</property>
        <property name="hibernate.c3p0.min_size">5</property>
        <property name="hibernate.c3p0.max_statements">10</property>

        <!-- 数据库方言 -->
        <property name="dialect">org.hibernate.dialect.MySQLDialect</property>

        <!-- 打印SQL -->
        <property name="show_sql">true</property>

        <!-- 格式化SQL -->
        <property name="format_sql">true</property>

        <!-- 是否自动生成数据库 -->
        <property name="hibernate.hbm2ddl.auto"></property>

        <!-- 注册实体关系映射文件 -->
        <mapping resource="com/southwind/entity/People.hbm.xml"></mapping>
        <mapping resource="com/southwind/entity/Customer.hbm.xml"></mapping>
        <mapping resource="com/southwind/entity/Orders.hbm.xml"></mapping>
        <mapping resource="com/southwind/entity/Account.hbm.xml"></mapping>
        <mapping resource="com/southwind/entity/Course.hbm.xml"></mapping>

    </session-factory>

</hibernate-configuration>
```

1、数据库的基本信息。

```xml
<property name="connection.username">root</property>
<property name="connection.password">root</property>
<property name="connection.driver_class">com.mysql.cj.jdbc.Driver</property>
<property name="connection.url">jdbc:mysql://localhost:3306/test?
useUnicode=true&amp;characterEncoding=UTF-8</property>
```

2、集成 C3P0，设置数据库连接池信息。

```xml
<property name="hibernate.c3p0.acquire_increment">10</property>
<property name="hibernate.c3p0.idle_test_period">10000</property>
<property name="hibernate.c3p0.timeout">5000</property>
<property name="hibernate.c3p0.max_size">30</property>
<property name="hibernate.c3p0.min_size">5</property>
<property name="hibernate.c3p0.max_statements">10</property>
```

3、Hibernate 基本信息。

```xml
<!-- 数据库方言 -->
<property name="dialect">org.hibernate.dialect.MySQLDialect</property>

<!-- 打印SQL -->
<property name="show_sql">true</property>

<!-- 格式化SQL -->
<property name="format_sql">true</property>

<!-- 是否自动生成数据库 -->
<property name="hibernate.hbm2ddl.auto"></property>
```

- update：动态创建表，如果表存在，则直接使用，如果表不存在，则创建。
- create：无论表是否存在，都会重新创建。
- create-drop：初始化创建表，程序结束时删除表。
- validate：校验实体关系映射文件和数据表是否对应，不能对应直接报错。

4、注册实体关系映射文件。

```xml
<!-- 注册实体关系映射文件 -->
<mapping resource="com/southwind/entity/People.hbm.xml"></mapping>
<mapping resource="com/southwind/entity/Customer.hbm.xml"></mapping>
<mapping resource="com/southwind/entity/Orders.hbm.xml"></mapping>
<mapping resource="com/southwind/entity/Account.hbm.xml"></mapping>
<mapping resource="com/southwind/entity/Course.hbm.xml"></mapping>
```

实体关系映射文件

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD
3.0//EN"
        "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping>

    <class name="com.southwind.entity.Course" table="t_course">

        <id name="id" type="java.lang.Integer">
            <column name="id"></column>
            <generator class="identity"></generator>
        </id>

        <property name="name" type="java.lang.String">
            <column name="name"></column>
        </property>

        <set name="accounts" table="account_course" lazy="false">
            <key column="cid"></key>
            <many-to-many class="com.southwind.entity.Account" column="aid">
</many-to-many>
        </set>

    </class>

</hibernate-mapping>
```

hibernate-mapping 属性

- package：给 class 节点对应的实体类统一设置包名，此处设置包名，class 的 name 属性就可以省略包名。

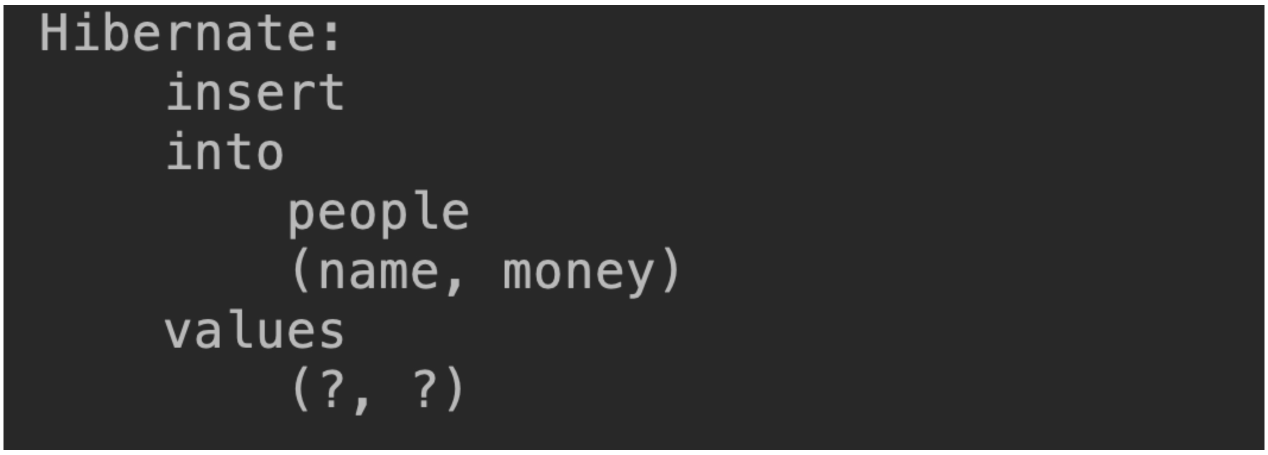
```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD
3.0//EN"
        "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="com.southwind.entity">

    <class name="Course" table="t_course">

        <id name="id" type="java.lang.Integer">
            <column name="id"></column>
            <generator class="identity"></generator>
        </id>

        <property name="name" type="java.lang.String">
            <column name="name"></column>
        </property>
        <set name="accounts" table="account_course" lazy="false">
            <key column="cid"></key>
            <many-to-many class="Account" column="aid"></many-to-many>
        </set>

    </class>

</hibernate-mapping>
```

- schema：数据库 schema 的名称
- catalog：数据库 catalog 的名称
- default-cascade：默认的级联关系，默认为 none
- default-access：Hibernate 用来访问属性的策略
- default-lazy：指定了未明确注明 lazy 属性的 Java 属性和集合类，Hibernate 会采用什么样的加载风格，默认为 true
- auto-import：指定我们是否可以在查询语句中使用非全限定类名，默认为 true，如果项目中有两个同名的持久化类，最好在这两个类的对应映射文件中国配置为 false

class 属性

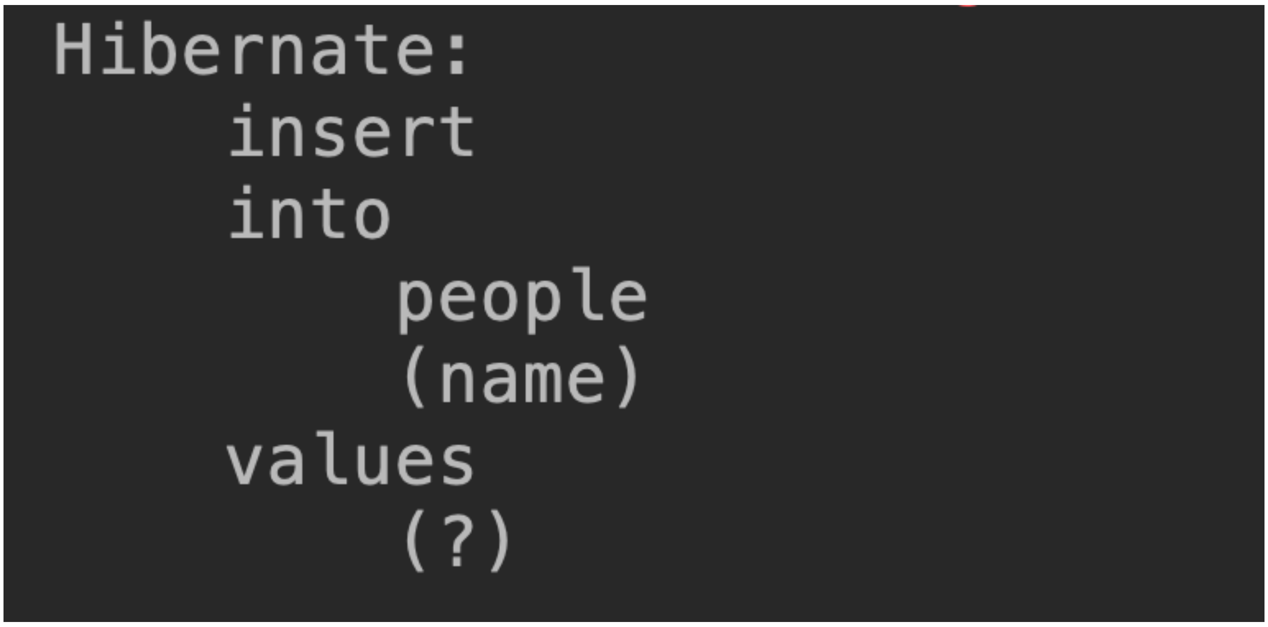
- name：实体类名
- table：数据表名
- schema：数据库 schema 的名称，会覆盖 hibernate-mapping 的 schema
- catalog：数据库 catalog 的名称，会覆盖 hibernate-mapping 的 catalog
- proxy：指定一个接口，在延迟加载时作为代理使用
- dynamic-update：动态更新
- dynamic-insert：动态添加

```java
package com.southwind.test;

import com.southwind.entity.Account;
import com.southwind.entity.People;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class Test8 {
    public static void main(String[] args) {
        //创建 Configuration
        Configuration configuration = new
Configuration().configure("hibernate.xml");
        //获取 SessionFactory
        SessionFactory sessionFactory = configuration.buildSessionFactory();
        //获取 Session
        Session session = sessionFactory.openSession();

        People people = new People();
        people.setName("张三");

        session.save(people);

        session.close();
    }
}
```



```
Hibernate:
    insert
    into
        people
        (name, money)
    values
        (?, ?)
```

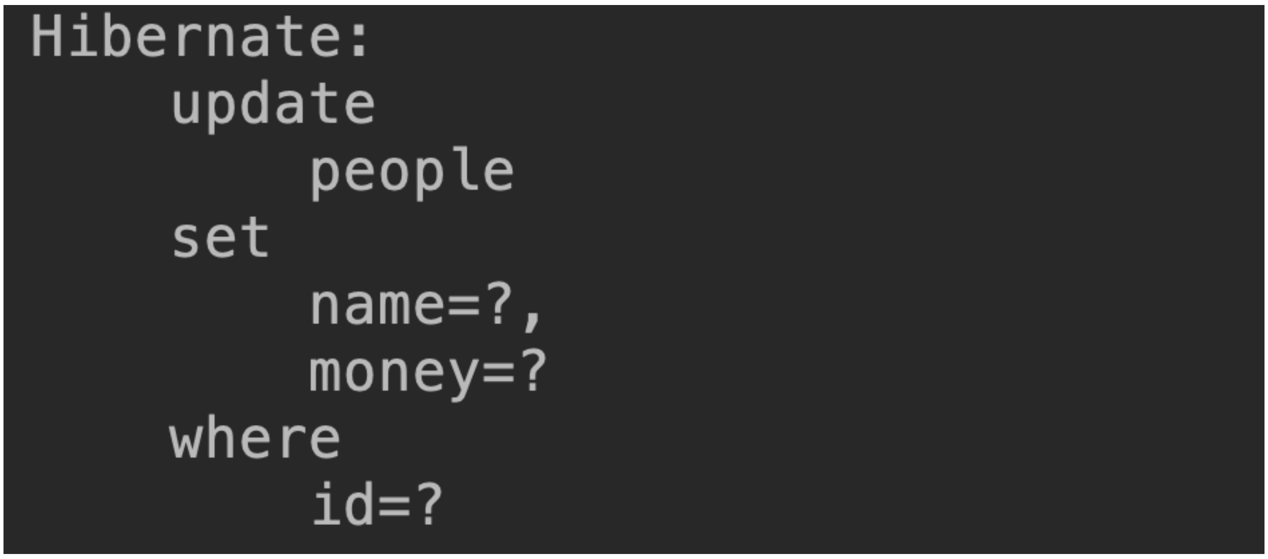
```
<class name="com.southwind.entity.People" table="people" dynamic-
insert="true">
```



```
Hibernate:
    insert
    into
        people
        (name)
    values
        (?)
```

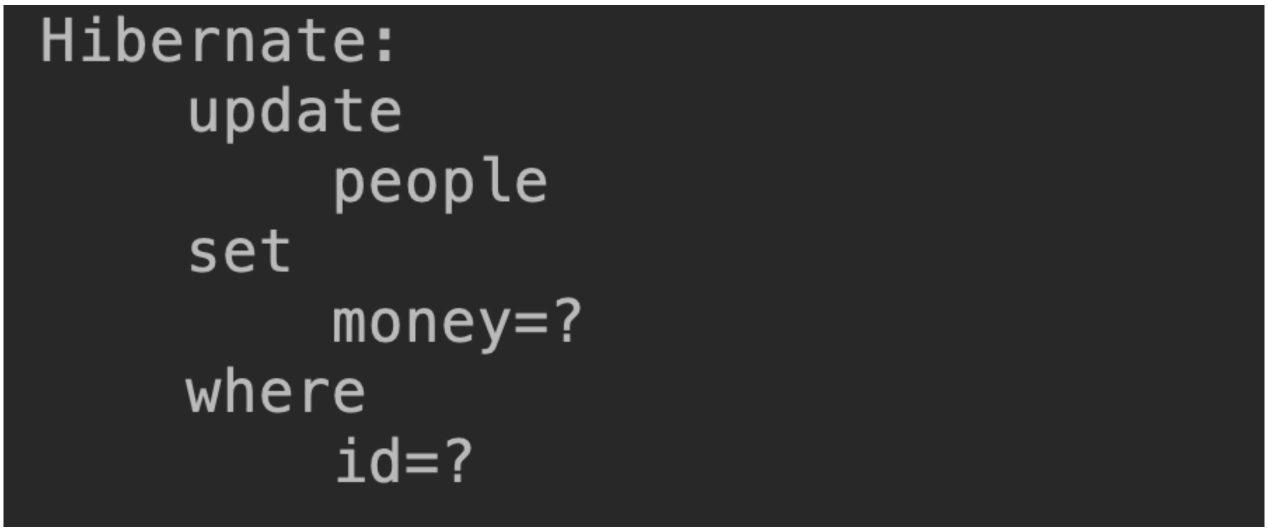
```
package com.southwind.test;

import com.southwind.entity.Account;
import com.southwind.entity.People;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class Test8 {
```

```java
    public static void main(String[] args) {
        //创建 Configuration
        Configuration configuration = new
Configuration().configure("hibernate.xml");
        //获取 SessionFactory
        SessionFactory sessionFactory = configuration.buildSessionFactory();
        //获取 Session
        Session session = sessionFactory.openSession();

        People people = session.get(People.class,6);
        people.setMoney(2000.0);

        session.update(people);

        session.beginTransaction().commit();

        session.close();
    }
}
```



```
Hibernate:
    update
        people
    set
        name=?,
        money=?
    where
        id=?
```

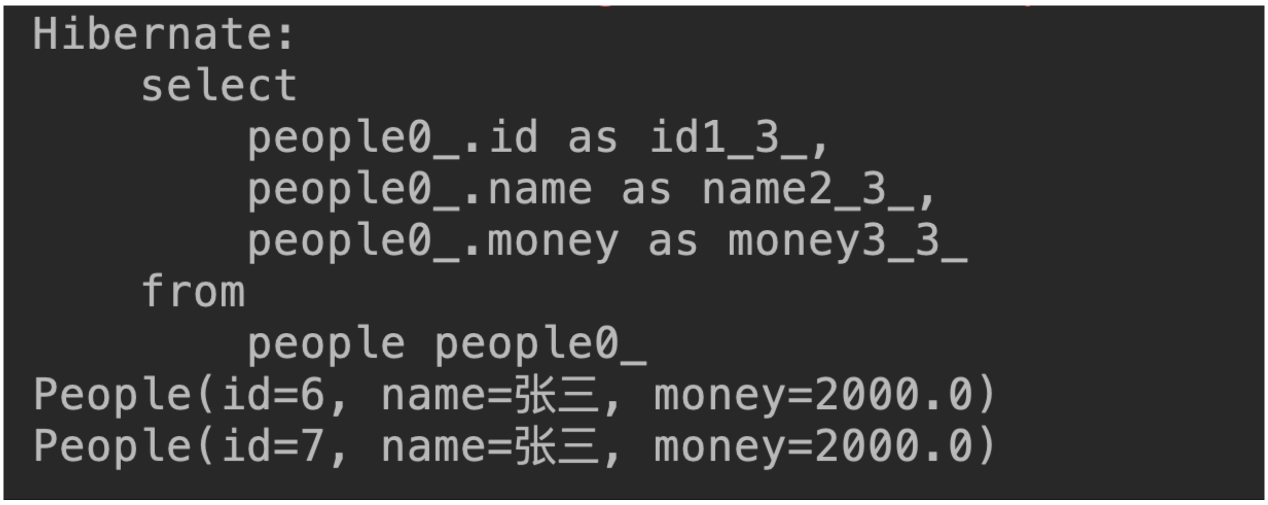
```xml
<class name="com.southwind.entity.People" table="people" dynamic-insert="true"
dynamic-update="true">
```



```
Hibernate:
    update
        people
    set
        money=?
    where
        id=?
```

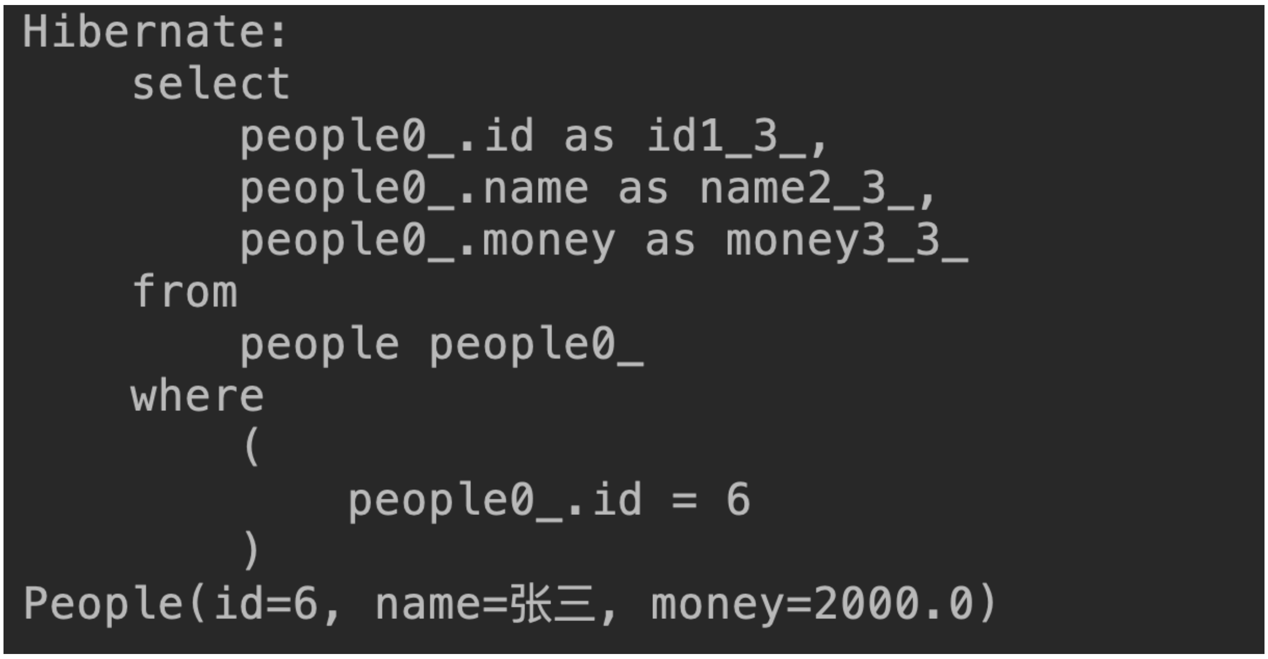
- where：查询时给 SQL 添加 where 条件

```java
package com.southwind.test;

import com.southwind.entity.Account;
import com.southwind.entity.People;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;
import org.hibernate.query.Query;

import java.util.List;

public class Test8 {
    public static void main(String[] args) {
        //创建 Configuration
        Configuration configuration = new
Configuration().configure("hibernate.xml");
        //获取 SessionFactory
        SessionFactory sessionFactory = configuration.buildSessionFactory();
        //获取 Session
        Session session = sessionFactory.openSession();

        String hql = "from People";
        Query query = session.createQuery(hql);
        List<People> list = query.list();
        for (People people:list) {
            System.out.println(people);
        }

        session.beginTransaction().commit();

        session.close();
    }
}
```



```
Hibernate:
    select
        people0_.id as id1_3_,
        people0_.name as name2_3_,
        people0_.money as money3_3_
    from
        people people0_
People(id=6, name=张三, money=2000.0)
People(id=7, name=张三, money=2000.0)
```

```xml
<class name="com.southwind.entity.People" table="people" dynamic-insert="true"
dynamic-update="true" where="id = 6">
```



```
Hibernate:
    select
        people0_.id as id1_3_,
        people0_.name as name2_3_,
        people0_.money as money3_3_
    from
        people people0_
    where
        (
            people0_.id = 6
        )
People(id=6, name=张三, money=2000.0)
```

id 属性

- name：实体类属性名
- type：实体类属性数据类型

此处可以设置两种类型的数据：Java 数据类型或者 Hibernate 映射类型。

实体类的属性数据类型必须与数据表对应的字段数据类型一致：

int 对应 int，String 对应 varchar

如何进行映射？

Java 数据类型映射到 Hibernate 映射类型，再由 Hibernate 映射类型映射到 SQL 数据类型

Java ---》Hibernate ---〉SQL

- column：数据表的主键字段名

- generator：主键生成策略

1、hilo 算法

2、increment：Hibernate 自增

3、identity：数据库自增

4、native：本地策略，根据底层数据库自动选择主键的生成策略

5、uuid.hex 算法

6、select 算法

property 属性

- name：实体类的属性名
- column：数据表字段名
- type：数据类型
- update：该字段是否可以修改，默认为 true
- insert：该字段是否可以添加，默认为 true
- lazy：延迟加载策略

实体关系映射文件属性

1、inverse

Customer 和 Orders 是一对多关系，一个 Customer 对应多个 Orders，实体类中用一个 set 集合来表示对应的 Orders。

```java
package com.southwind.entity;

import lombok.Data;
import lombok.Getter;
import lombok.Setter;

import java.util.Set;

@Getter
@Setter
public class Customer {
    private Integer id;
    private String name;
    private Set<Orders> orders;

    @Override
    public String toString() {
        return "Customer{" +
                "id=" + id +
                ", name='" + name + '\'' +
                '}';
    }
}
```

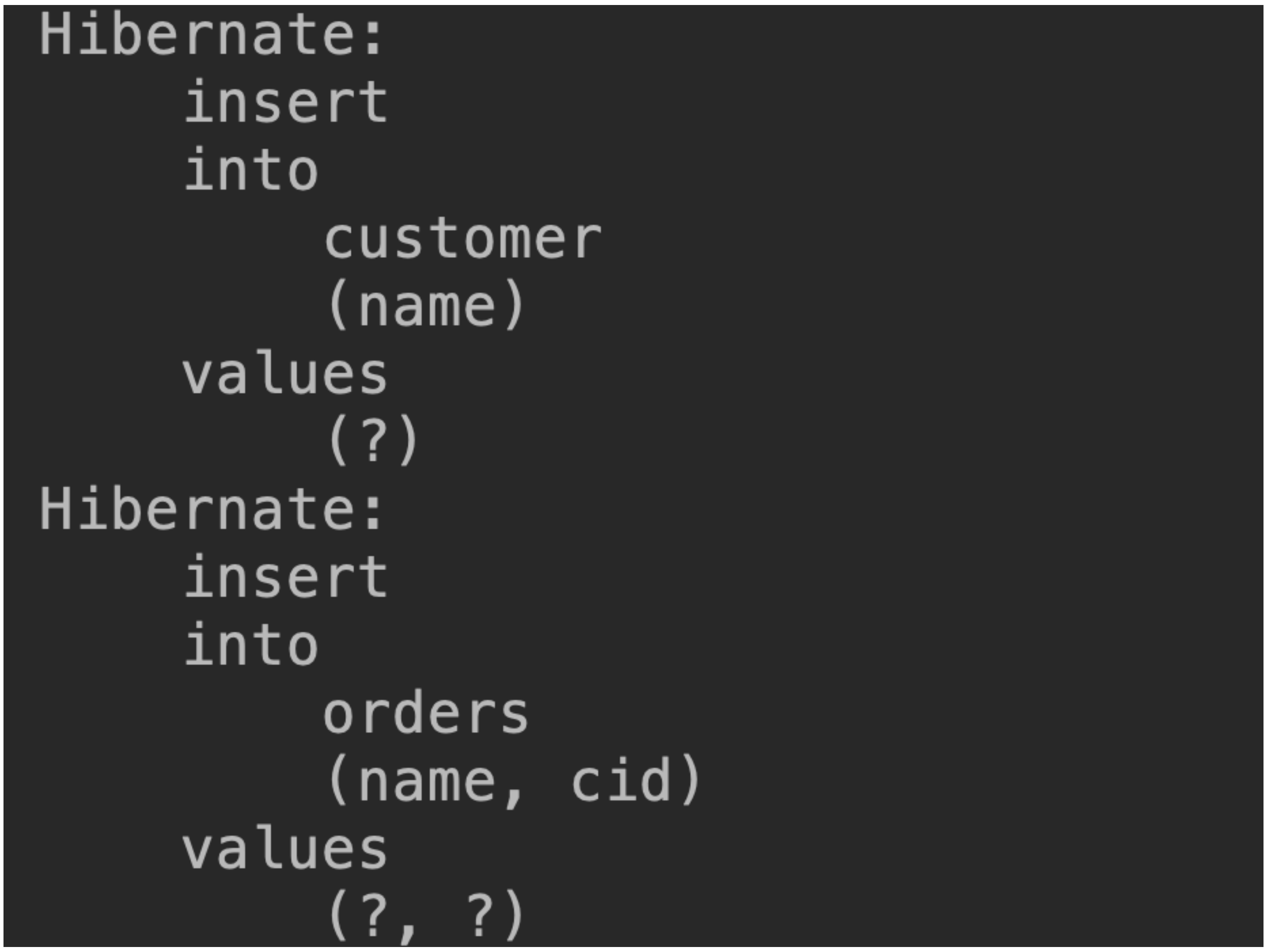
Customer.hbm.xml 中使用 set 标签来配置映射关系。

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD
3.0//EN"
        "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping>

    <class name="com.southwind.entity.Customer" table="customer">

        <id name="id" type="java.lang.Integer">
            <column name="id"></column>
            <generator class="identity"></generator>
        </id>

        <property name="name" type="java.lang.String">
            <column name="name"></column>
        </property>

        <set name="orders" table="orders" lazy="extra">
            <key column="cid"></key>
            <one-to-many class="com.southwind.entity.Orders"></one-to-many>
        </set>

    </class>

</hibernate-mapping>
```

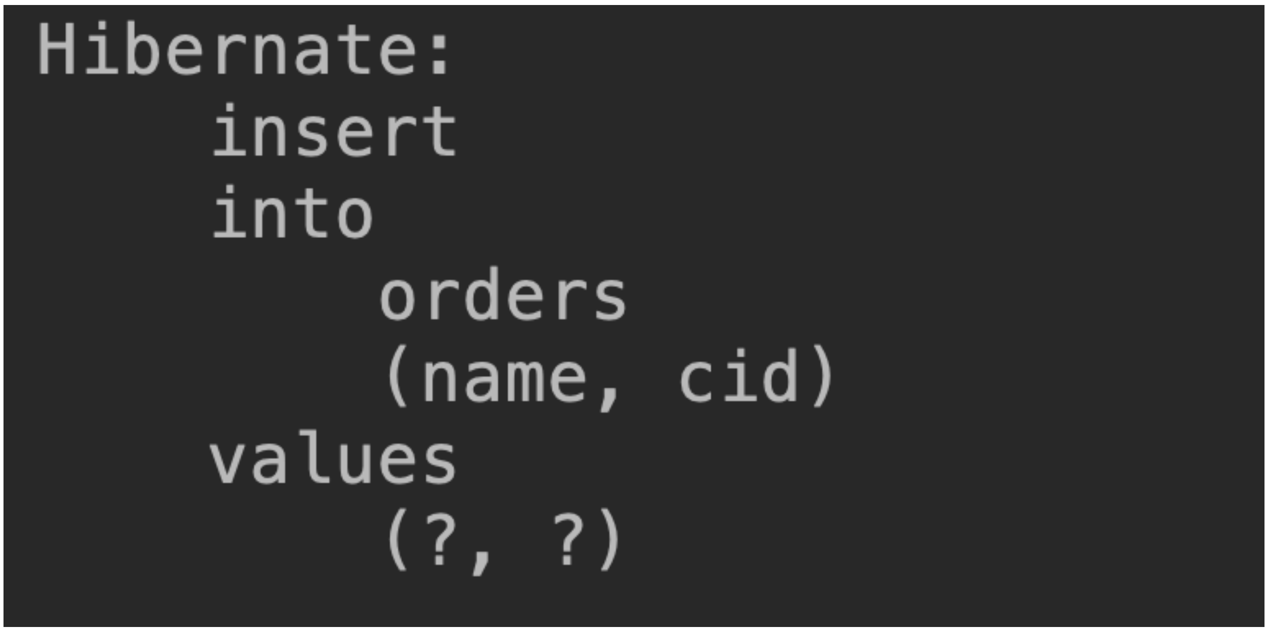
```
package com.southwind.entity;

import lombok.Data;
import lombok.Getter;
import lombok.Setter;

@Getter
@Setter
public class Orders {
    private Integer id;
    private String name;
    private Customer customer;

    @Override
    public String toString() {
        return "Orders{" +
                "id=" + id +
                ", name='" + name + '\'' +
                '}';
    }
}
```

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
        "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping>

    <class name="com.southwind.entity.Orders" table="orders">

        <id name="id" type="java.lang.Integer">
            <column name="id"></column>
            <generator class="identity"></generator>
        </id>

        <property name="name" type="java.lang.String">
            <column name="name"></column>
        </property>

        <many-to-one name="customer" class="com.southwind.entity.Customer" column="cid" lazy="proxy"></many-to-one>

    </class>

</hibernate-mapping>
```

```
package com.southwind.test;

import com.southwind.entity.Customer;
import com.southwind.entity.Orders;
import com.southwind.entity.People;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;
import org.hibernate.query.Query;

import java.util.List;

public class Test9 {
    public static void main(String[] args) {
        //创建 Configuration
        Configuration configuration = new Configuration().configure("hibernate.xml");
        //获取 SessionFactory
        SessionFactory sessionFactory = configuration.buildSessionFactory();
        //获取 Session
        Session session = sessionFactory.openSession();

        Customer customer = new Customer();
        customer.setName("张三");

        Orders orders1 = new Orders();
        orders1.setName("订单1");
        orders1.setCustomer(customer);

        Orders orders2 = new Orders();
        orders2.setName("订单2");
        orders2.setCustomer(customer);

        session.save(customer);
        session.save(orders1);
        session.save(orders2);

        session.beginTransaction().commit();

        session.close();
    }
}
```

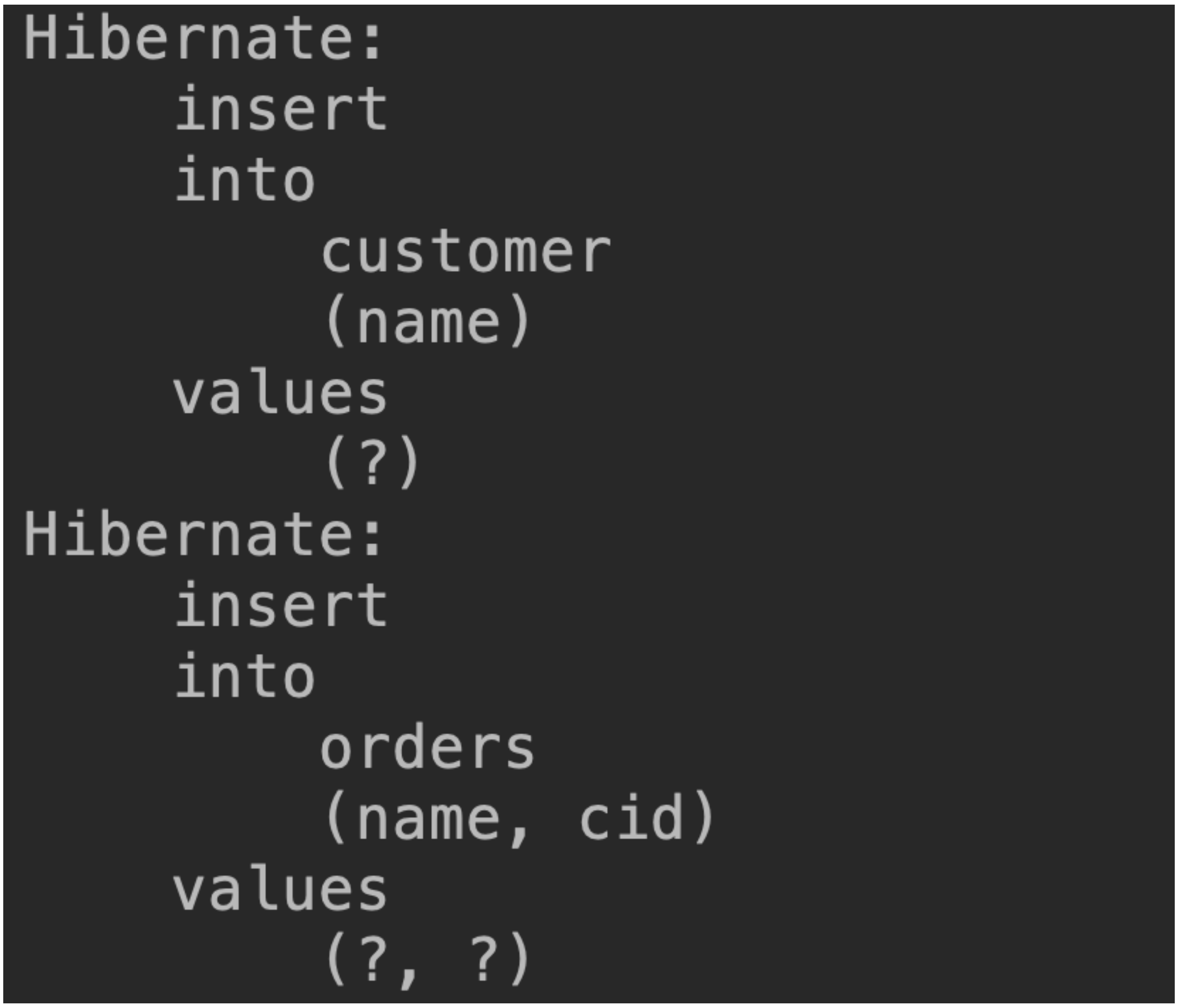


```
Hibernate:
    insert
    into
        customer
        (name)
    values
        (?)
Hibernate:
    insert
    into
        orders
        (name, cid)
    values
        (?, ?)
```

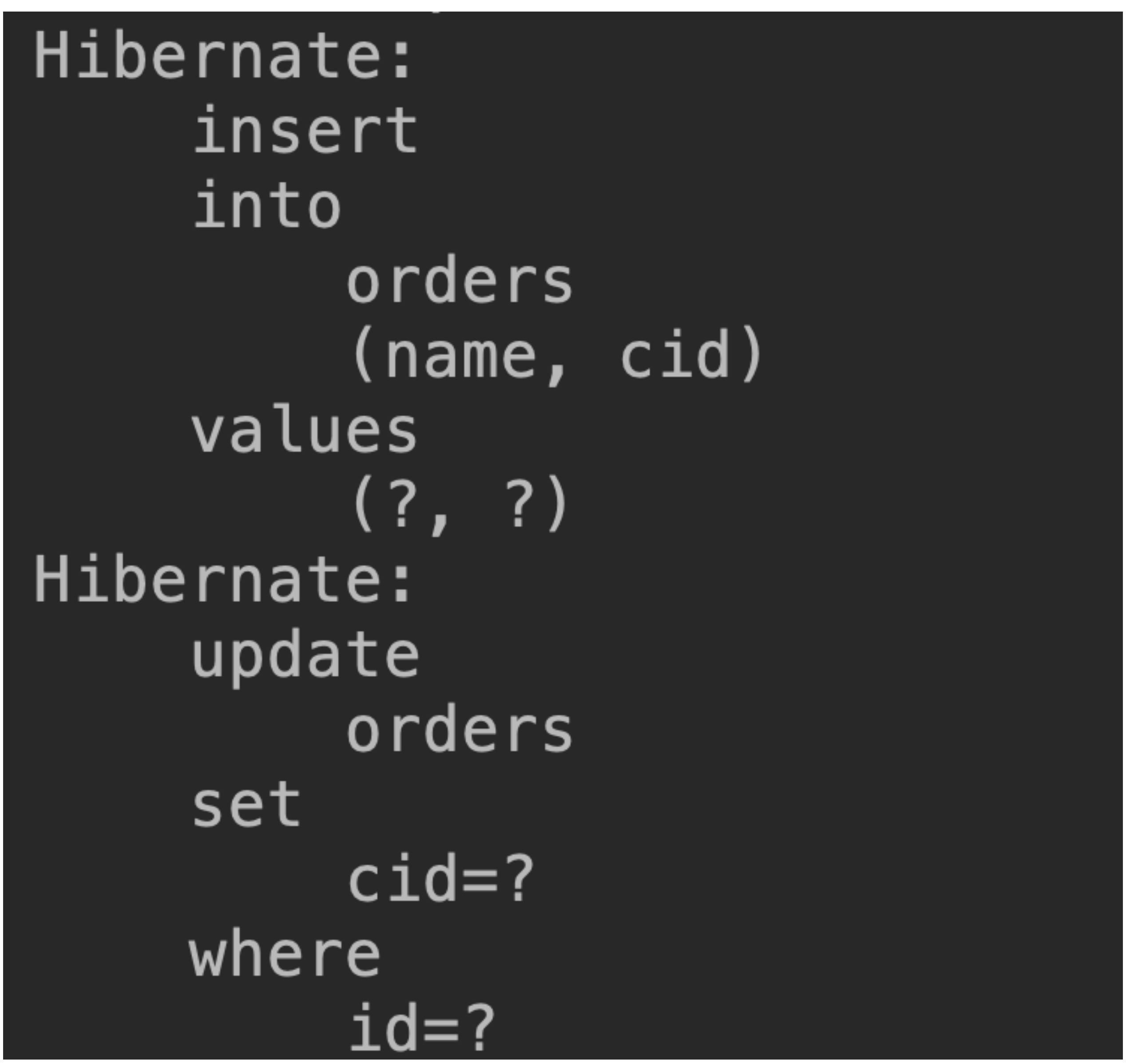


```
Hibernate:
    insert
    into
        orders
        (name, cid)
    values
        (?, ?)
```

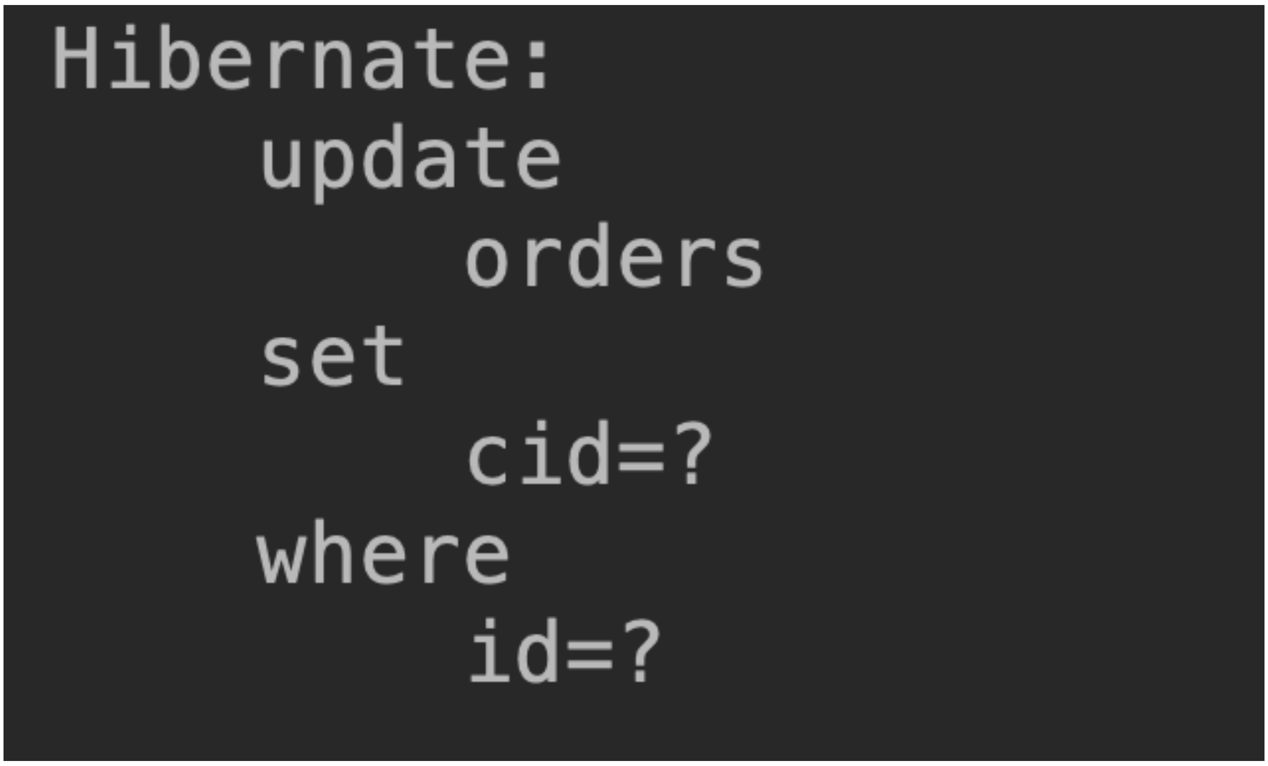
```java
package com.southwind.test;

import com.southwind.entity.Customer;
import com.southwind.entity.Orders;
import com.southwind.entity.People;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;
import org.hibernate.query.Query;

import java.util.HashSet;
import java.util.List;
import java.util.Set;

public class Test9 {
    public static void main(String[] args) {
        //创建 Configuration
        Configuration configuration = new
Configuration().configure("hibernate.xml");
        //获取 SessionFactory
        SessionFactory sessionFactory = configuration.buildSessionFactory();
        //获取 Session
        Session session = sessionFactory.openSession();

        Customer customer = new Customer();
        customer.setName("张三");

        Orders orders1 = new Orders();
        orders1.setName("订单1");
        orders1.setCustomer(customer);

        Orders orders2 = new Orders();
        orders2.setName("订单2");
        orders2.setCustomer(customer);

        Set<Orders> orders = new HashSet<>();
        orders.add(orders1);
        orders.add(orders2);
        customer.setOrders(orders);

        session.save(customer);
        session.save(orders1);
        session.save(orders2);

        session.beginTransaction().commit();

        session.close();
    }
}
```



```
Hibernate:
    insert
    into
        customer
        (name)
    values
        (?)
Hibernate:
    insert
    into
        orders
        (name, cid)
    values
        (?, ?)
```



```
Hibernate: 
    insert 
    into
        orders
        (name, cid) 
    values
        (?, ?)
Hibernate: 
    update
        orders 
    set
        cid=? 
    where
        id=?
```



```
Hibernate: 
    update
        orders 
    set
        cid=? 
    where
        id=?
```

因为 Customer 和 Orders 都在维护一对多关系，所以会重复设置主外键约束关系。

如何避免这种情况?

1、在 Java 代码中去掉一方维护关系代码。

2、通过配置来解决。

```
<set name="orders" table="orders" lazy="extra" inverse="true">
    <key column="cid"></key>
    <one-to-many class="com.southwind.entity.Orders"></one-to-many>
</set>
```

inverse 属性是用来设置是否将维护权交给对方，默认是 false，不交出维护权，双方都在维护，将它设置为 true，表示 Customer 放弃维护。

cascade：用来设置级联操作

```
package com.southwind.test;

import com.southwind.entity.Customer;
import com.southwind.entity.Orders;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

import java.util.HashSet;
import java.util.Iterator;
import java.util.Set;

public class Test10 {
    public static void main(String[] args) {
        //创建 Configuration
        Configuration configuration = new
Configuration().configure("hibernate.xml");
        //获取 SessionFactory
        SessionFactory sessionFactory = configuration.buildSessionFactory();
        //获取 Session
        Session session = sessionFactory.openSession();

        Customer customer = session.get(Customer.class,18);

        Iterator<Orders> iterator = customer.getOrders().iterator();
        while(iterator.hasNext()){
            session.delete(iterator.next());
        }

        session.delete(customer);

        session.beginTransaction().commit();

        session.close();
    }
}
```

实体关系映射文件中设置 cascade 值完成级联删除。

```
<set name="orders" table="orders" lazy="extra" inverse="true"
cascade="delete">
    <key column="cid"></key>
    <one-to-many class="com.southwind.entity.Orders"></one-to-many>
</set>
```

```
package com.southwind.test;

import com.southwind.entity.Customer;
import com.southwind.entity.Orders;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

import java.util.HashSet;
import java.util.Iterator;
import java.util.Set;

public class Test10 {
    public static void main(String[] args) {
        //创建 Configuration
        Configuration configuration = new
Configuration().configure("hibernate.xml");
        //获取 SessionFactory
        SessionFactory sessionFactory = configuration.buildSessionFactory();
        //获取 Session
        Session session = sessionFactory.openSession();

        Customer customer = session.get(Customer.class,19);
        session.delete(customer);

        session.beginTransaction().commit();

        session.close();
    }
}
```