

	<u>Java Language</u>	Java Language						
		java	javac	javadoc	jar	javap	jdeps	Scripting
<u>Tools &amp; Tool APIs</u>		Security	Monitoring	JConsole	VisualVM	JMC	JFR	
		JPDA	JVM TI	IDL	RMI	Java DB	Deployment	
		Internationalization		Web Services		Troubleshooting		
<u>Deployment</u>		Java Web Start			Applet / Java Plug-in			
		JavaFX						
<u>User Interface Toolkits</u>		Swing		Java 2D	AWT	Accessibility		
		Drag and Drop		Input Methods	Image I/O	Print Service	Sound	
		IDL	JDBC	JNDI	RMI	RMI-IIOP	Scripting	
<u>JRE</u>	<u>Integration Libraries</u>	Beans	Security		Serialization		Extension Mechanism	
	<u>Other Base Libraries</u>	JMX	XML JAXP		Networking		Override Mechanism	
		JNI	Date and Time		Input/Output		Internationalization	
		lang and util						
	<u>lang and util Base Libraries</u>	Math	Collections		Ref Objects		Regular Expressions	
	Logging	Management		Instrumentation		Concurrency Utilities		
	Reflection	Versioning		Preferences API		JAR	Zip	
	<u>Java Virtual Machine</u>	Java HotSpot Client and Server VM						

JDK

JRE

Java SE API

Compact Profiles

ResultSet

boolean next();

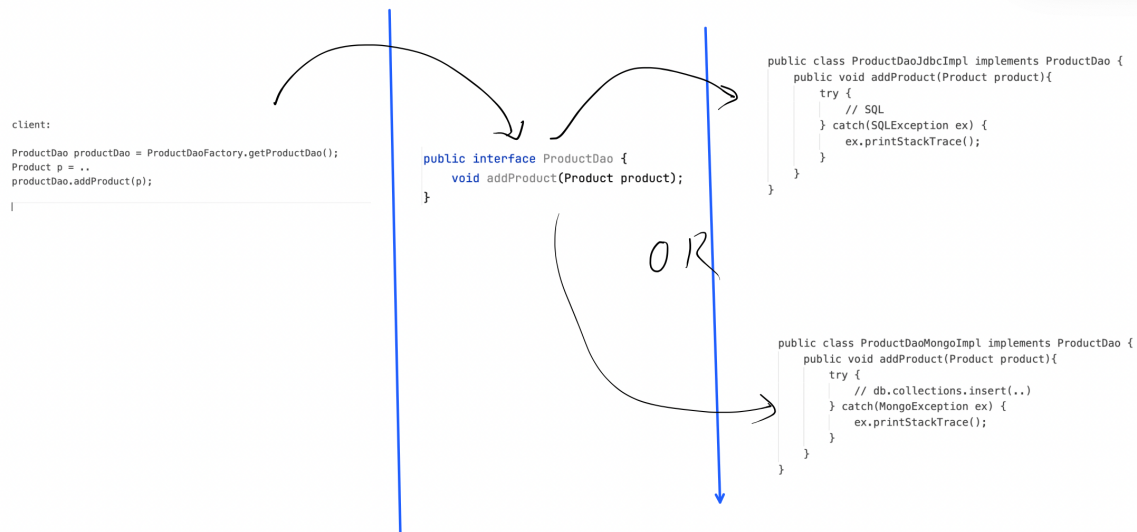
rs.getInt("id")

rs.getString("name");

id	age	name
8	21	John Doe
11	23	James
12	23	James
13	31	James

### Scenario 1:

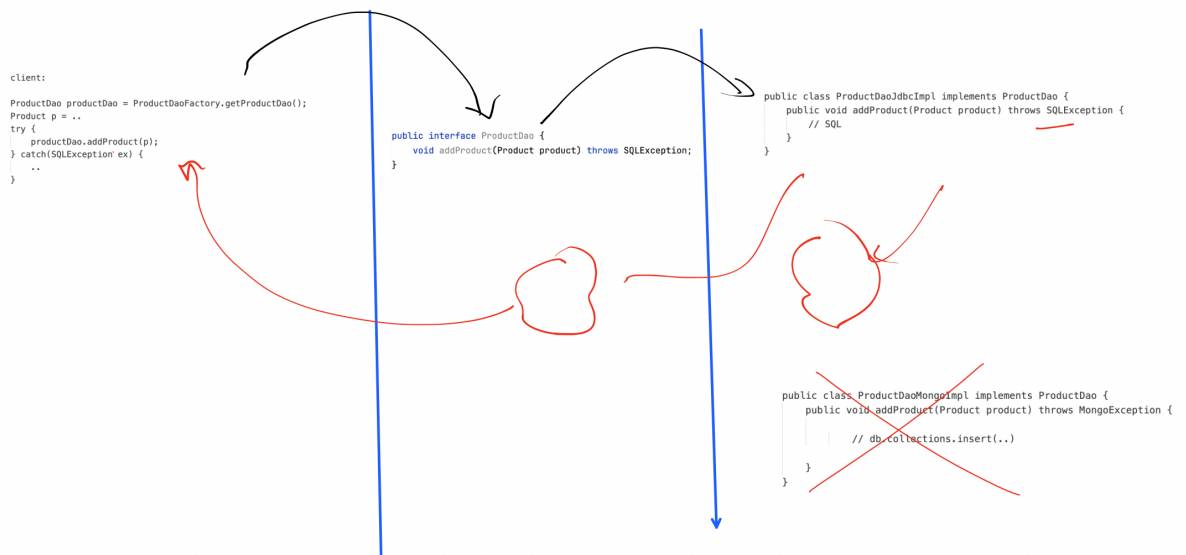
Exceptions are not propagated to client; client has no clue if exception occurred or not



### Scenario 2:

Exceptions are propagated to client.

- Tight coupling
- No abstraction; why should we expose the implementation type to client
- Exception messages are not understandable to client [ORA9001; unique constraint exception..]



### Scenario 3:

Using CustomExceptions

```

ProductDao productDao = ProductDaoFactory.getProductDao();
Product p = ..
try {
    productDao.addProduct(p);
} catch (PersistenceException ex) {
    ..
}

```

```

public interface ProductDao {
    void addProduct(Product product) throws PersistenceException;
}

```

```

public class ProductDaoJdbcImpl implements ProductDao {
    public void addProduct(Product product) throws PersistenceException {
        try {
            // SQL
        } catch (SQLException ex) {
            log
            throw new PersistenceException(message);
        }
    }
}

```

```

public class ProductDaoMongoImpl implements ProductDao {
    public void addProduct(Product product) throws PersistenceException {
        try {
            // db.collections.insert(..)
        } catch (MongoException ex) {
            log
            throw new PersistenceException(message);
        }
    }
}

```

```
mysql> select * from customers;
```

email	first_name
raj@cisco.com	Rajesh
rani@cisco.com	Rani

```
mysql> select * from products;
```

id	name	price	qty
1	iPhone 15	89000	100
2	Samsung Fold	145000	100
3	Tata Play	5400.55	100
4	Wacom	9400	100

update  
n ←

*Orders*

Ord	Order_date	Total	Customer_fk
914	12-12-2024 2:40:00	8993	rani@cisco.com
977	12-12-2024 3:10	9813	raja@cisco.com
8223	--	8211	rani@cisco.com

*Items*

Item_id	Order_fk	Product_fk	quantity	amount
5	914	2	1	125000
6	914	1	2	150000
7	977	1	1	90000

INSERT

n → INSERT