

MODÜL - 2
**İLERİ JAVA VE
VERİTABANI**

**PAKET - 6
VERİTABANI VE
SQL**



JDBC

T E C H P R O E D

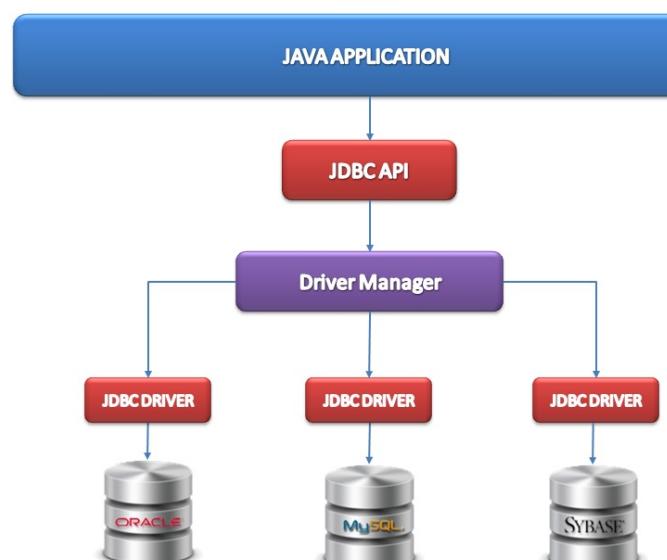
JDBC NEDİR?

- JDBC: Sun firmasının ticari markasıdır.
- Java Database Connectivity (Java Veritabanı Bağlantısı)
- JDBC, Java dilinde yazılmış programların, ilişkisel veritabanları ile iletişim kurabilmesi için kullanılan bir **Java API** sidir.
- JDBC, Java dilinde yazılmış sınıf ve arayüzler içermektedir
- Veritabanı uygulamalarının Sadece Java ile gerçekleştirilmesini mümkün kılar.
- JDBC standartlarını **Sun Microsystems** belirlenmektedir. Ancak, farklı firmaların bu standardı kullanarak kendi JDBC sürücülerini geliştirmesine izin vermektedir.

T E C H P R O E D

JDBC NEDİR?

- Java oldukça standartlaşmıştır, Ancak SQL'in farklı bir çok versiyonu bulunmaktadır.
- JDBC, Java ortamında SQL veritabanına erişim için bir araçtır.



JDBC API'si, ilgili veritabanının JDBC Sürücülerini yardımcıyla

- Veritabanı ile bağlantı kurar.
- SQL ifadelerini gönderir.
- Sonuçları işler.

JDBC'DE TANIMLI SINIFLAR (CLASSES)

- Driver
 - **getConnection**
- Bağlantı
 - **createStatement**
- Sql ifadesi çalışma
 - **execute, executeQuery, executeBatch, executeUpdate**
- Sonuçları alma
 - **next, getString, getInt, getDate, getMetaData**
- ResultSetMetadata
 - **getColumnCount, getColumnNome, getColumnType**

T E C H P R O E D

JDBC ADIMLARI

1. Kullanılacak Veritabanı için doğru sürücüyü ekle,
2. Veritabanı ile iletişimini başlat,
3. SQL ifadeleri oluştur ve çalıştır (Select, insert/update/delete)
4. Gelen sonuçları işle, kaydet vb.,
5. Veritabanı bağlantısını bitir.

T E C H P R O E D

1. DRIVER (SÜRÜCÜ) EKLEME

- Driver'ı eklemek için aşağıdaki metot kullanılır.
 - **Class.forName("driver class")**
- Driver, seçilen veritabanına göre değişmektedir.
 - **oracle.jdbc.driver.OracleDriver**
 - mysql.jdbc.Driver
 - org.postgresql.Driver
- **ÖRNEK:**
`Class.forName("oracle.jdbc.driver.OracleDriver");`

T E C H P R O E D

2. VERİTABANI BAĞLANTISI OLUŞTURMA

- Veritabanı ile bağlantı oluşturabilmek için **getConnection()** metodu kullanılır.
- Bu metot, bir **Connection** nesnesi döndürür.
- **DriverManager.getConnection(url, user, pwd)**
 - **ÖRNEK1:** "jdbc:oracle:thin:@localhost:1521/orcl", "hr", "oracle"
 - **ÖRNEK2:** "jdbc:oracle:thin:@localhost:1521/ORCLCDB.localdomain", "murat", "1234"
- **ÖRNEK:**
 - **Connection con =**
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521/orcl",
"hr", "oracle");

T E C H P R O E D

3. SQL İFADESİ OLUŞTURMA VE ÇALIŞTIRMA

A) Bir SQL ifade oluşturmak için **createStatement()** metodu kullanılır.

- **Statement st = con.createStatement();**
- **st,** SQL komutlarını yazabilmemize imkan tanıyan bir **Statement** nesnesidir.

B) SQL ifadelerini çalıştırmak için ise aşağıdaki Metotlar kullanılabilir.

- **executeQuery()** => SELECT ifadeleri için
- **executeUpdate()** => INSERT, UPDATE, DELETE ifadeleri
- **SELECT** ifadesinin sonuç kümesi (satırlar/sutunlar) bir **ResultSet** nesnesi olarak döner.

ÖRNEK:

- **ResultSet rs = st.executeQuery("SELECT * FROM personel");**

T E C H P R O E D

4. SONUÇLARI İŞLE

- **ResultSet** olarak alınan SQL sonuçlarının her bir satırına erişmek için bir döngü kullanılabilir.
- Döngü içerisinde dolaşmak için rs.next() metodu kullanılmalıdır.

```
ResultSet rs = st.executeQuery("Select personel_isim FROM personel");
while(rs.next()) {
    System.out.println("Personel Adı:" + rs.getString("personel_isim"));
}
```

- Bir sütundaki değerleri almak için
 - String personelId = rs.getString("personel_id");
 - Double maas = rs.getDouble("maas");

T E C H P R O E D

5. BAĞLANTIYI BİTİR

- ResultSet nesnesi kapat.
 - **rs.close();**
- Statement nesnesini kapat.
 - **st.close();**
- Bağlantıyı kapat.
 - **con.close();**

ORNEK

```
import java.sql.*;      // JDBC metotlari icin kutuphane
public class JdbcQuery01 {
    public static void main(String[] args) throws ClassNotFoundException, SQLException {
        //1) Veritabani icin uygun Driver'i ekle
        Class.forName("oracle.jdbc.driver.OracleDriver");

        //2) Veritabani baglantisi olustur
        Connection con = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521/ORCLCDB.localdomain", "meryem", "1234");

        //3) SQL komutlari icin bir Statement nesnesi olustur.
        Statement st = con.createStatement();

        //4)Sorgu ifadesini calistir. ( Personel tablosundaki id=101 olan personelin ismini sorgula)
        ResultSet rs1 = st.executeQuery("select isim from personel where id=101");

        //5)Sonucları işle
        while(rs1.next()) {
            System.out.println("Personel Adı: " + rs1.getString("isim"));
        }
        //6) Olusturulan nesneleri bellekten kaldır.
        st.close();
        con.close();
        rs1.close();
    }
}
```