JDBC: Java Database Connectivity

Java Database Connectivity (JDBC) is a Java-based API for connecting to and interacting with relational databases. JDBC provides a standard interface that enables Java applications to execute SQL queries and manage database connections. This presentation explores the key components of JDBC.

JDBC Components

Get Ready to Drive Your Data!

Introducing the JDBC Drivers and Managers:

1 **Drivers** ✓

2 **Driver Manager** ✓

3 **Connection** ✓

4 **Statement** ✓

5 **ResultSet** ✓

6 **SQL Exception** ✓

Client → Language

JDBC-ODBC Bridge

Databa

JDBC Drivers

Java App. ← Database MySQL //

= { MySQL IBM PostGreSQL }

1 Type 1: JDBC-ODBC Bridge Driver

The JDBC-ODBC Bridge provides a JDBC interface to ODBC data sources. This driver is based on the Sun Microsystems JDBC-ODBC Bridge.

2 Type 2: Native-API Driver

The Native-API Driver uses a vendor-supplied library to communicate with the database. This driver requires a library specific to the database being accessed.

Java Application

client → Middleware

Database

3 Type 3: Network Protocol Driver

The Network Protocol Driver communicates with a middleware server that provides a translation layer between the JDBC API and the database-specific protocol.

4 Type 4: Thin Driver (Direct-to-Database)

The Thin Driver communicates directly with the database using Java sockets. It does not require any additional software to be installed and is the most commonly used driver.

→ Java

Client

) directly

Database

Class

Interface

DriverManager

- The DriverManager class manages a list of database drivers.
- It establishes connections to the database using the appropriate driver and handles the process of loading the driver class.
- It provides a standardized method for handling multiple database connections and selecting the appropriate driver.

Connection

→ Interface

1 Creation of Statements

The Connection interface is used to create a Statement object, which is used to execute SQL queries against the database.

2 Transaction Management

The Connection interface allows transactions to be managed with methods such as commit() and rollback().

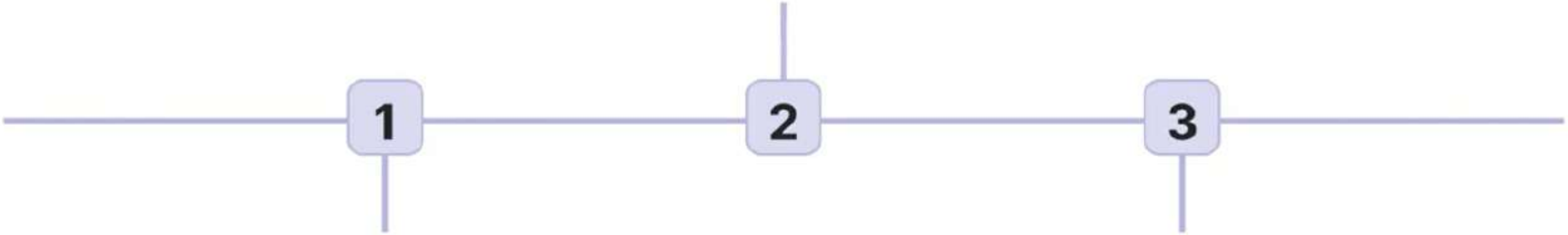
3 Retrieval of Metadata

The Connection interface provides methods to retrieve metadata from the database, including information about the database structure and the various objects that are defined in it.

Statement

Select * from Student ;

Java Code



PreparedStatement

PreparedStatement is used for executing precompiled SQL queries with parameters, which can be more efficient and secure than Statement objects.

Statement

The Statement interface is used for executing simple SQL queries without parameters.

CallableStatement

CallableStatement is used for executing database stored procedures. It provides a more efficient way to access them than with SQL statements.

ResultSet

✓ Retrieving Data

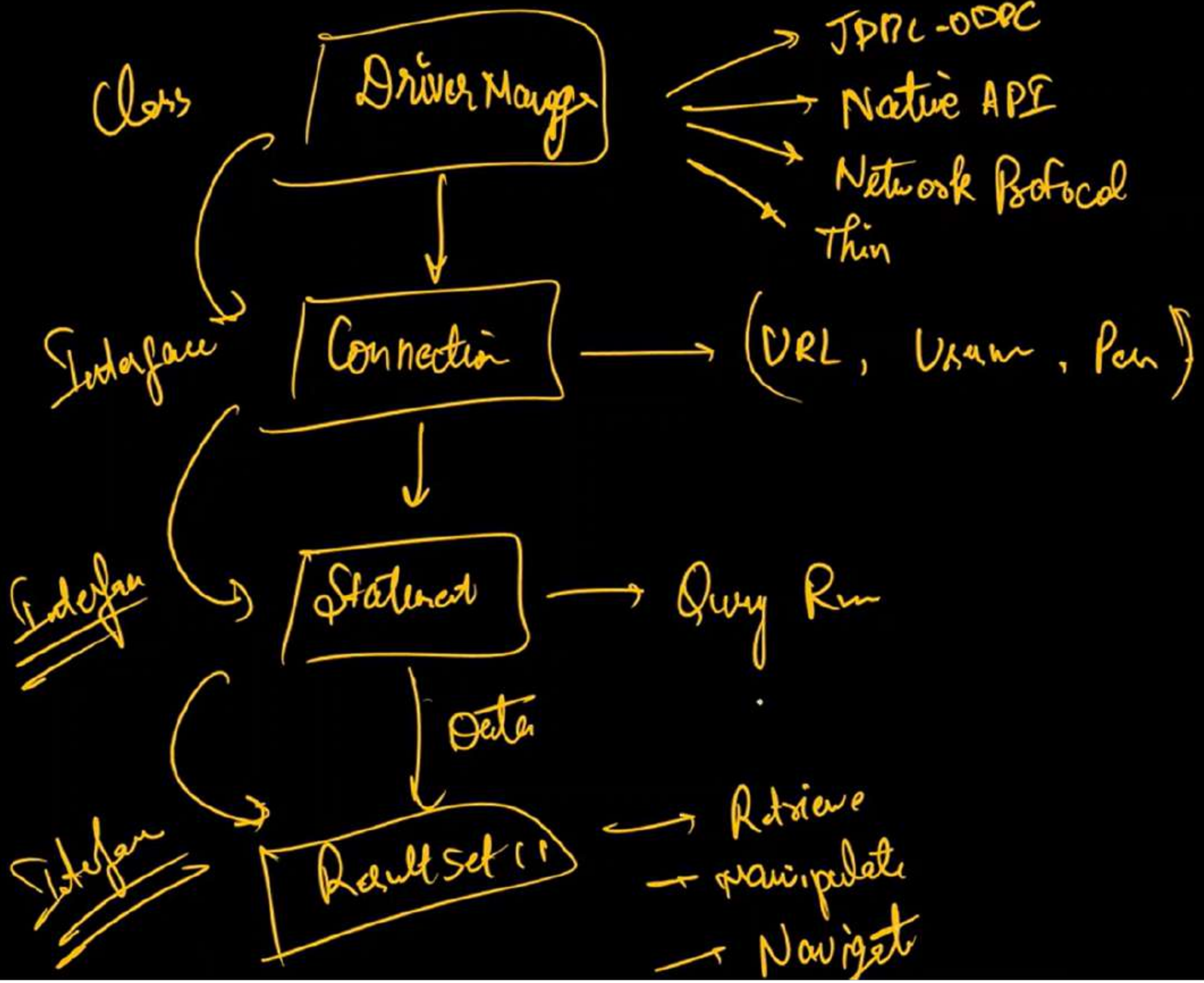
The ResultSet interface is used to iterate through the rows of the result set and retrieve the data from the columns.

✓ Scrolling Through Results

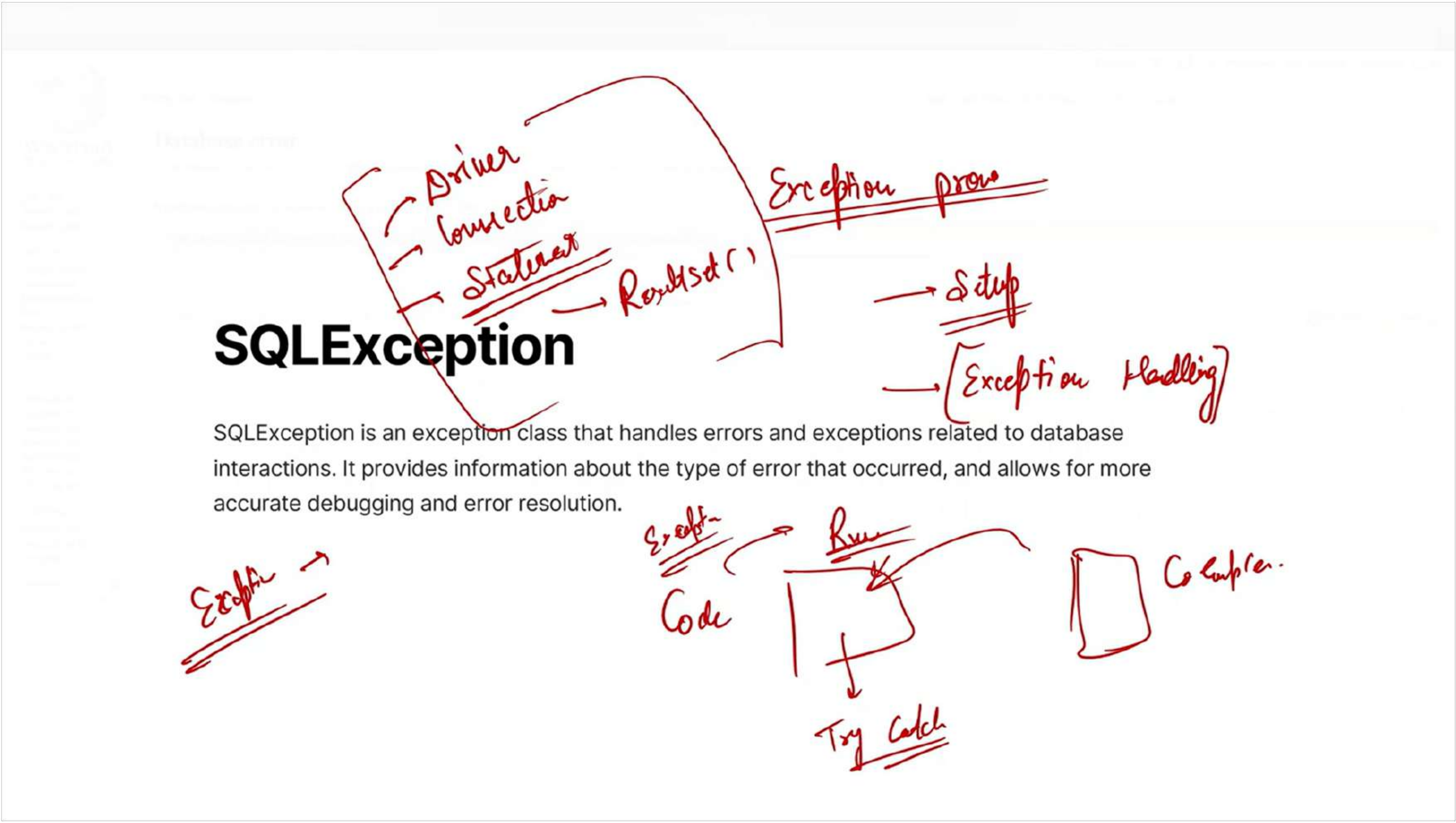
The ResultSet can be scrolled forwards and backwards to navigate through the results of the SQL query.

Modification of Data

The ResultSet allows modifications to the data within the database to be made, including insertion, deletion, and updates.



Class
Driver Manager
JDBC-ODBC
Native API
Network Protocol
Thin
Interface
Connection
(URL, Username, Pass)
Interface
Statement
Query Run
Data
Interface
Resultset()
Retrieve
manipulate
Navigate



SQLException

SQLException is an exception class that handles errors and exceptions related to database interactions. It provides information about the type of error that occurred, and allows for more accurate debugging and error resolution.