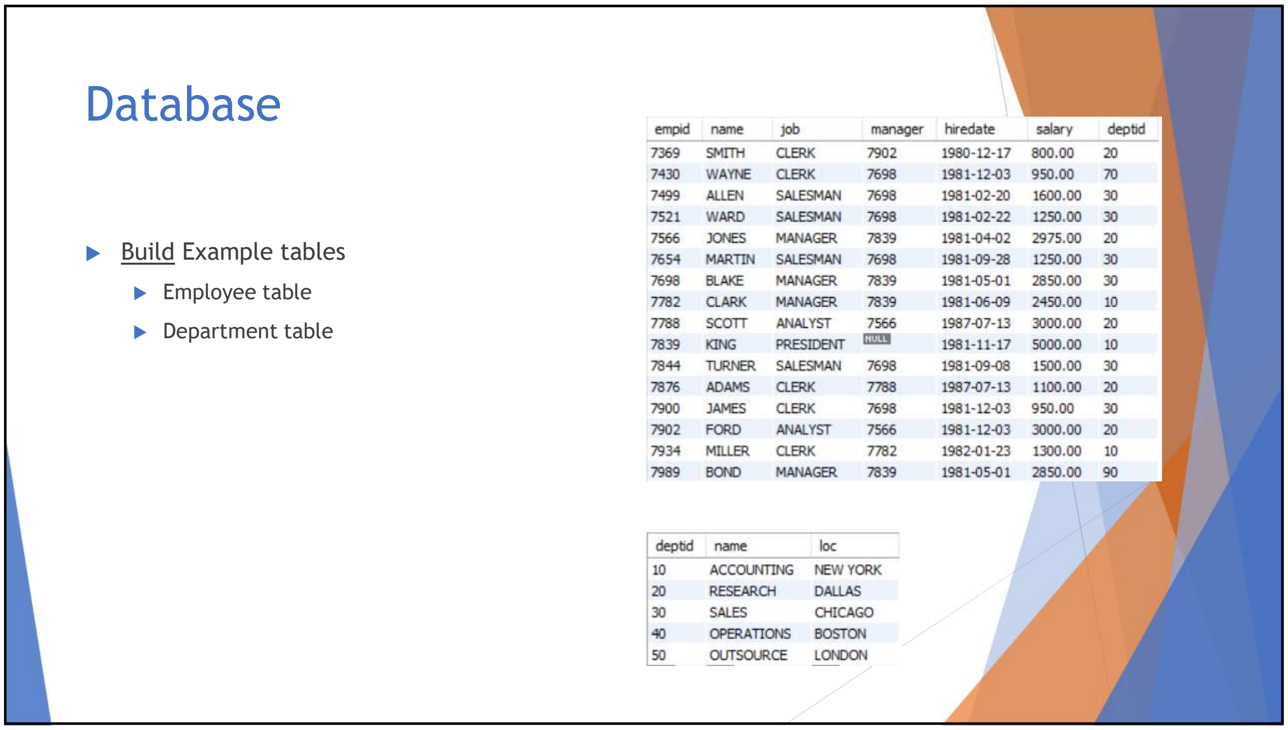
DATABASE

Andy

1



Database

- Build Example tables
  - Employee table
  - Department table

empid	name	job	manager	hiredate	salary	deptid
7369	SMITH	CLERK	7902	1980-12-17	800.00	20
7430	WAYNE	CLERK	7698	1981-12-03	950.00	70
7499	ALLEN	SALESMAN	7698	1981-02-20	1600.00	30
7521	WARD	SALESMAN	7698	1981-02-22	1250.00	30
7566	JONES	MANAGER	7839	1981-04-02	2975.00	20
7654	MARTIN	SALESMAN	7698	1981-09-28	1250.00	30
7698	BLAKE	MANAGER	7839	1981-05-01	2850.00	30
7782	CLARK	MANAGER	7839	1981-06-09	2450.00	10
7788	SCOTT	ANALYST	7566	1987-07-13	3000.00	20
7839	KING	PRESIDENT	NULL	1981-11-17	5000.00	10
7844	TURNER	SALESMAN	7698	1981-09-08	1500.00	30
7876	ADAMS	CLERK	7788	1987-07-13	1100.00	20
7900	JAMES	CLERK	7698	1981-12-03	950.00	30
7902	FORD	ANALYST	7566	1981-12-03	3000.00	20
7934	MILLER	CLERK	7782	1982-01-23	1300.00	10
7989	BOND	MANAGER	7839	1981-05-01	2850.00	90

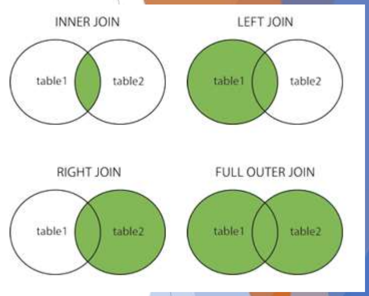
deptid	name	loc
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON
50	OUTSOURCE	LONDON

2

Database

- SQL: CREATION, VIEW, INSERT, UPDATE, DELETE, IN, SUBQUERY, VIEW
- Different Types of Joins
- Primary Key vs Unique Key vs Foreign Key
- Truncate vs Delete vs Drop
- View vs Materialized View vs Table
- Non-clustered Index vs Clustered Index
- Union, Union All

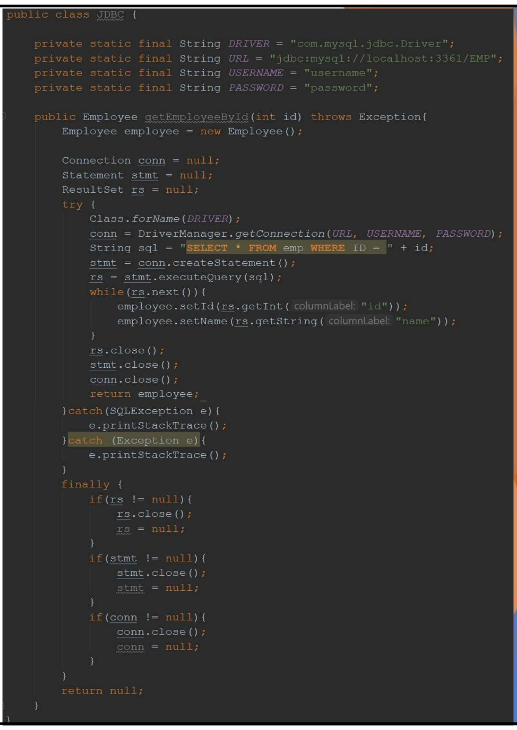
- SQL: Distinct, **Group by**, isnull/nvl/ifnull, **Join** on, **Having, Order by**
- SQL: Trigger, **Stored Procedure**, Function, Cursor(3)



3

JDBC

- DataSource:
  - Driver, URL, Username, Password
- Try/Catch/Finally
  - Order of "Catch"
  - Necessity of "Finally"

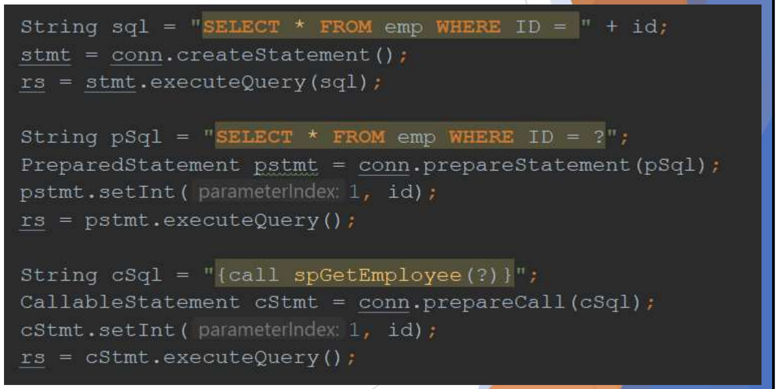


```java
public class JDBC {

    private static final String DRIVER = "com.mysql.jdbc.Driver";
    private static final String URL = "jdbc:mysql://localhost:3361/EMP";
    private static final String USERNAME = "username";
    private static final String PASSWORD = "password";

    public Employee getEmployeeById(int id) throws Exception{
        Employee employee = new Employee();

        Connection conn = null;
        Statement stmt = null;
        ResultSet rs = null;
        try {
            Class.forName(DRIVER);
            conn = DriverManager.getConnection(URL, USERNAME, PASSWORD);
            String sql = "SELECT * FROM emp WHERE ID = " + id;
            stmt = conn.createStatement();
            rs = stmt.executeQuery(sql);
            while(rs.next()){
                employee.setId(rs.getInt( columnLabel: "id"));
                employee.setName(rs.getString( columnLabel: "name"));
            }
            rs.close();
            stmt.close();
            conn.close();
            return employee;
        }catch(SQLException e){
            e.printStackTrace();
        }catch (Exception e){
            e.printStackTrace();
        }
        finally {
            if(rs != null){
                rs.close();
                rs = null;
            }
            if(stmt != null){
                stmt.close();
                stmt = null;
            }
            if(conn != null){
                conn.close();
                conn = null;
            }
        }
        return null;
    }
}
```

4

Statement, PreparedStatement Callable statement

- PreparedStatement is pre-compiled; it takes parameters
- CallableStatement is for calling stored procedures
- SQL INJECTION



```
String sql = "SELECT * FROM emp WHERE ID = " + id;
stmt = conn.createStatement();
rs = stmt.executeQuery(sql);

String pSql = "SELECT * FROM emp WHERE ID = ?";
PreparedStatement pstmt = conn.prepareStatement(pSql);
pstmt.setInt( parameterIndex: 1, id);
rs = pstmt.executeQuery();

String cSql = "{call spGetEmployee(?)}";
CallableStatement cStmt = conn.prepareCall(cSql);
cStmt.setInt( parameterIndex: 1, id);
rs = cStmt.executeQuery();
```

5

RDBMS vs NoSQL

Relational Database	NoSQL Database
Structure Data (table based)	Unstructured Data (document, column based)
Supports Transactions	Does not support Transactions
Store Medium to Large Data	Store Huge Amount of Data
Relative Fixed Query Language (SQL)	Different Query languages
Performance Low	Performance High
Design Principle: ACID	Design Principle: CAP
Example: MySQL, Oracle, SQL Server	Example: MongoDB, Cassandra

Question: Which one to choose?

6

MySQL vs Cassandra vs MongoDB

Syntax	MySQL	Cassandra	MongoDB

Table	USE dbschema;
	CREATE TABLE emp()
		USE dbschema;
		CREATE TABLE emp()
			USE dbschema;
			db.createCollection(“emp”)

READ	SELECT * from emp;
	SELECT * from emp WHERE id=5;
		SELECT * from emp;
		SELECT * from emp WHERE id=5;
		(id must be primary key)
			db.emp.find()
			db.emp.find({id:5})

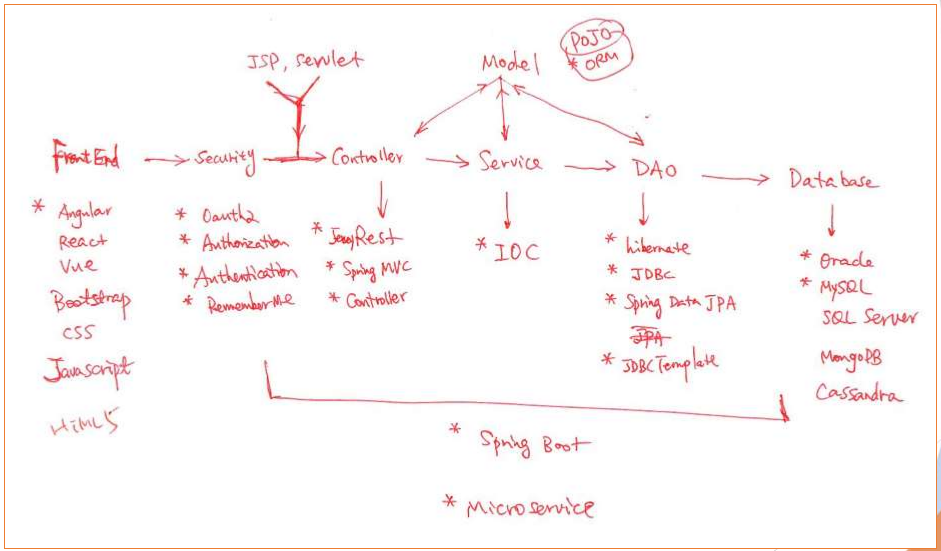
INSERT	INSERT INTO emp (id, name)
	VALUES (5, “SMITH”)
		INSERT INTO emp (id, name)
		VALUES (5, “SMITH”)
			db.emp.insert({ id:5, “name”:”SMITH”})

UPDATE	UPDATE emp
	SET name = “SMITH”
	WHERE id = 5;
		UPDATE emp
		SET name = “SMITH”
		WHERE id = 5;
			db.emp.update(
			{id: 5},
			{$set: {
			name: “SMITH”
			}
			}
			}

DELETE	DELETE FROM emp WHERE id=5	DELETE name FROM emp WHERE id=5 DELETE FROM emp WHERE id=5	db.emp.remove({id: 5})
JOIN	SELECT * FROM emp e JOIN department d ON e.deptid = d.deptid	NA	$lookup - NOT RECOMMENDED NA

7

Architecture Overview



8