


Introduction to SQL

- Stands for Structured Query Language
 - Standard interface to relational databases
 - SQL is used to create database structure, or to fill it with data, or to remove data or alter data
 - User Friendly Non Procedural Very High Level Query Language
- 
- The bottom right corner of the slide features a decorative graphic consisting of several sets of concentric circles, resembling ripples in water, rendered in a lighter blue shade than the background.

SQL Statements

□ **SELECT** **Data retrieval**

□ **INSERT**
□ **UPDATE** **Data manipulation language (DML)**
□ **DELETE**

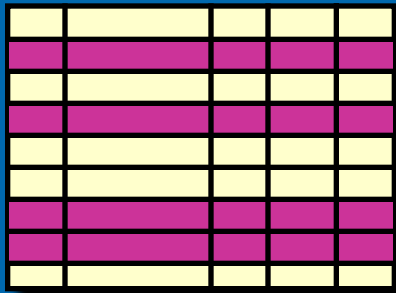
□ **CREATE**
□ **ALTER**
□ **DROP**
□ **RENAME** **Data definition language (DDL)**
□ **TRUNCATE**

□ **COMMIT**
□ **ROLLBACK**
□ **SAVEPOINT** **Transaction control**

□ **GRANT**
□ **REVOKE**
Data control language (DCL)

Capabilities of SQL SELECT Statements

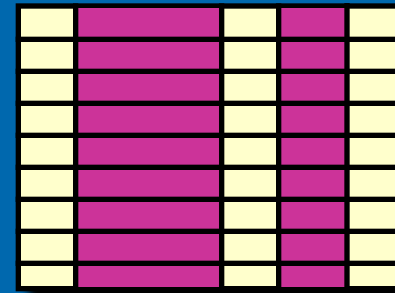
Selection



A 10x5 grid representing a table. The second, third, fourth, sixth, seventh, eighth, and ninth rows are highlighted in pink, indicating they are selected. The first, fifth, and tenth rows are yellow.

Table 1

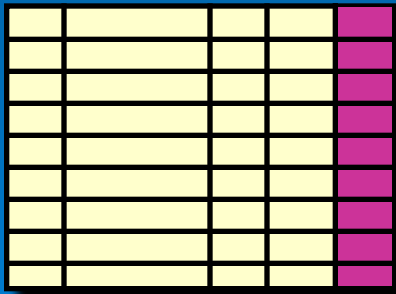
Projection



A 10x5 grid representing a table. The second, third, fourth, sixth, seventh, eighth, and ninth columns are highlighted in pink, indicating they are projected. The first, fifth, and tenth columns are yellow.

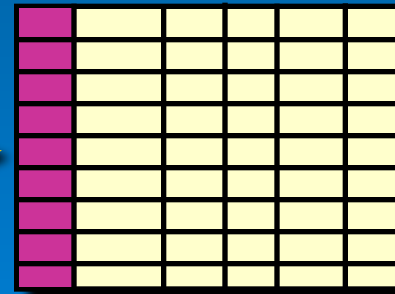
Table 1

Join



A 10x5 grid representing a table. The fifth column is highlighted in pink, indicating it is the join key. The other columns are yellow.

Table 1



A 10x5 grid representing a table. The first column is highlighted in pink, indicating it is the join key. The other columns are yellow.

Table 2

Capabilities of SQL SELECT Statements

- A SELECT statement retrieves information from the database. Using a SELECT statement, you can do the following:
 - Selection
 - Projection
 - Join

Writing SQL Statements

- SQL statements are not case sensitive.
- SQL statements can be on one or more lines.
- Keywords cannot be abbreviated or split across lines.
- Tabs and indents are used to enhance readability.

Guidelines are from Oracle Corporation for Oracle DBMS but also valid for other databases (if not all then also most of them are true)

Tables Used in the Lecture

EMP

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7839	KING	PRESIDENT		17-NOV-81	5000		10
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10
7566	JONES	MANAGER	7839	02-APR-81	2975		20
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30
7900	JAMES	CLERK	7698	03-DEC-81	950		30
			7698	22-FEB-81	1250	500	30
			7566	03-DEC-81	1300		20
			7902	17-DEC-80			
			7566	09-DEC-82			
			7788	12-JAN-83			
			7782	23-JAN-82			

DEPT

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

SALGRADE

GRADE	LOSAL	HISAL
1	700	1200
2	1201	1400
3	1401	2000
4	2001	3000
5	3001	9999

Basic SELECT Statement

```
SELECT <column list>  
FROM table;
```

- SELECT identifies *what* columns(similar to projection operation).
- FROM identifies *which* table.

Selecting All Columns

```
SQL> SELECT *  
2 FROM dept;
```

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

Selecting Specific Columns

```
SQL> SELECT deptno, loc  
2 FROM dept;
```

DEPTNO	LOC
10	NEW YORK
20	DALLAS
30	CHICAGO
40	BOSTON

Projection

Eliminating Duplicate Rows

Eliminate duplicate rows by using the **DISTINCT** keyword in the **SELECT** clause.

```
SQL> SELECT DISTINCT deptno  
2 FROM emp;
```

DEPTNO
10
20
30

Limiting Rows Selected

- Restrict the rows returned by using the WHERE clause.

```
SELECT      column1, column2, ...  
FROM        table  
[WHERE      condition(s)];
```

- The WHERE clause follows the FROM clause.

Using the WHERE Clause

```
SQL> SELECT ename, job, deptno  
2 FROM emp  
3 WHERE job='CLERK';
```

ENAME	JOB	DEPTNO
-----	-----	-----
JAMES	CLERK	30
SMITH	CLERK	20
ADAMS	CLERK	20
MILLER	CLERK	10

Character Strings and Dates

- Character strings and date values are enclosed in single quotation marks.
- Character values are case sensitive and date values are format sensitive.

```
SQL> SELECT  ename, job, deptno  
2  FROM      emp  
3  WHERE     ename = 'JAMES' ;
```

Comparison Operators

Operator	Meaning
=	Equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
<>	Not equal to

Using the Comparison Operators

```
SQL> SELECT ename, sal, comm  
2 FROM emp  
3 WHERE sal<=comm;
```

ENAME	SAL	COMM
MARTIN	1250	1400

Other Comparison Operators

Operator	Meaning
BETWEEN ...AND...	Between two values (inclusive)
IN(list)	Match any of a list of values
LIKE	Match a character pattern
IS NULL	Is a null value

Using the BETWEEN Operator

- Use the BETWEEN operator to display rows based on a range of values.

```
SQL> SELECT  ename, sal
2  FROM      emp
3  WHERE     sal BETWEEN 1000 AND 1500;
```

ENAME	SAL
MARTIN	1250
TURNER	1500
WARD	1250
ADAMS	1100
MILLER	1300

Lower
limit

Higher
limit

Using the IN Operator

- Use the IN operator to test for values in a list.

```
SQL> SELECT empno, ename, sal, mgr
2 FROM emp
3 WHERE mgr IN (7902, 7566, 7788);
```

EMPNO	ENAME	SAL	MGR
7902	FORD	3000	7566
7369	SMITH	800	7902
7788	SCOTT	3000	7566
7876	ADAMS	1100	7788

Defining a Column Alias

- Renames a column heading
- Is useful with calculations
- Immediately follows column name;
AS keyword between column name and
alias

Using Column Aliases

```
SQL> SELECT ename AS name, sal AS salary  
FROM emp;
```

NAME	SALARY
-----	-----
...	

```
SQL> SELECT ename Name,  
            sal*12 AS AnnualSalary  
FROM emp;
```

Name	AnnualSalary
-----	-----

...

Select-From-Where Statements

- The principal form of a query is:

SELECT desired attributes

FROM one or more tables

WHERE condition about tuples of
the tables