Introduction

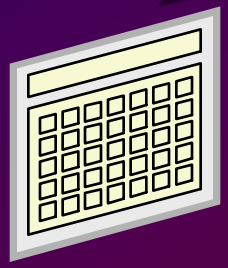
Data Storage on Different Media

SALGRADE

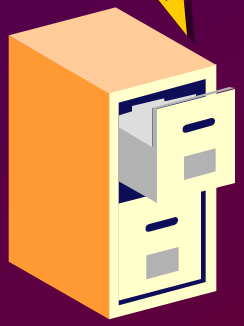
GRADE	LOSAL	HISAL
---	---	---
	700	1200
	1201	1400
	1401	2000
	2001	3000
	3001	9999

DEPT

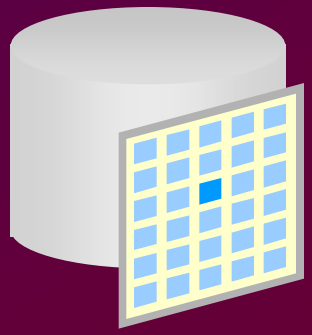
DEPTNO	DNAME	LOC
---	---	---
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON



Electronic spreadsheet



Filing cabinet



Database

Relational Database Concept

- Dr. E. F. Codd proposed the relational model for database systems in 1970.
- It is the basis for the relational database management system (RDBMS).
- The relational model consists of the following:
  - Collection of objects or relations
  - Set of operators to act on the relations
  - Data integrity for accuracy and consistency

Relational Database Definition

A relational database is a collection of relations or two-dimensional tables.

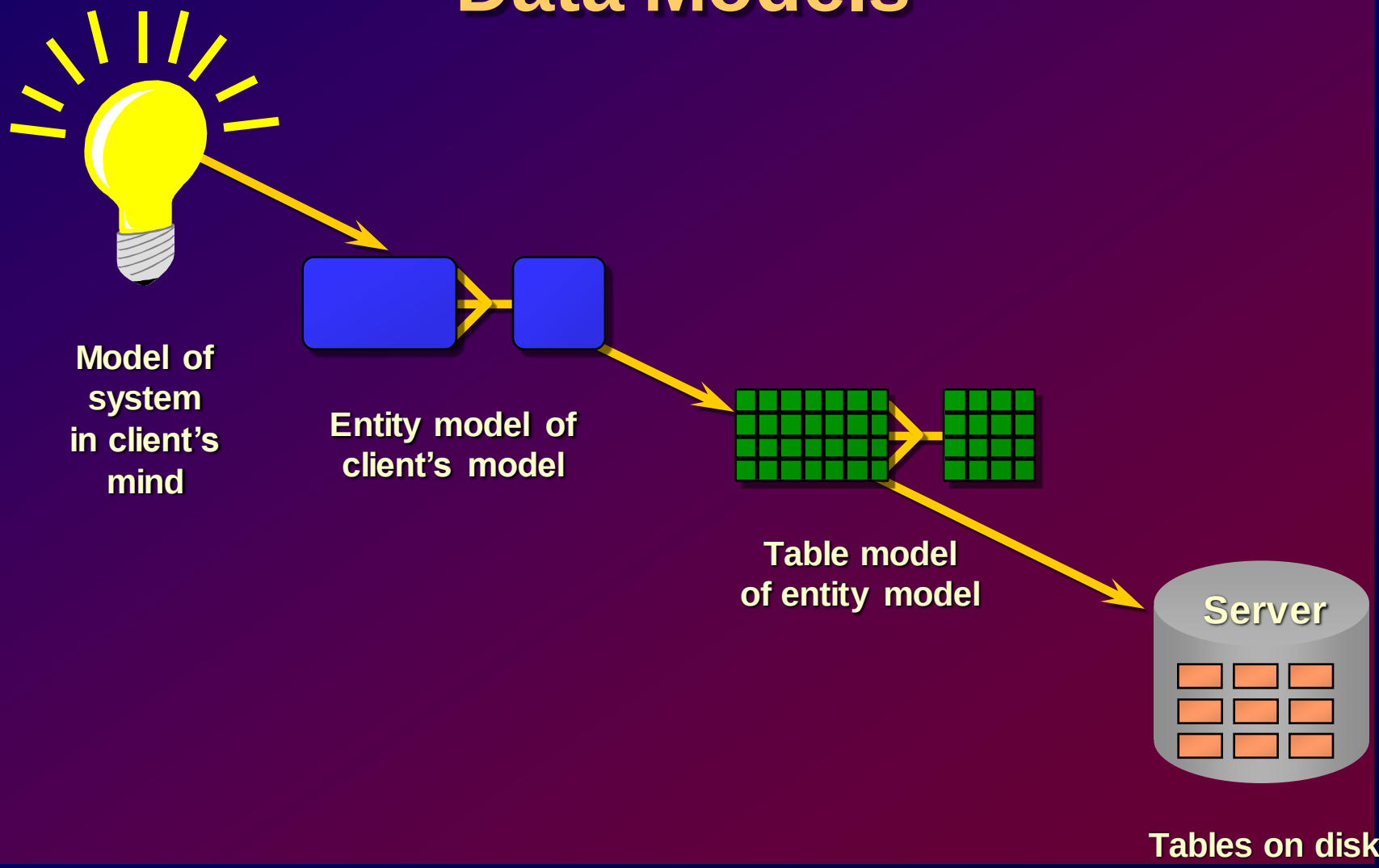
Table Name: **EMP**

EMPNO	ENAME	JOB	DEPTNO
7839	KING	PRESIDENT	10
7698	BLAKE	MANAGER	30
7782	CLARK	MANAGER	10
7566	JONES	MANAGER	20

Table Name: **DEPT**

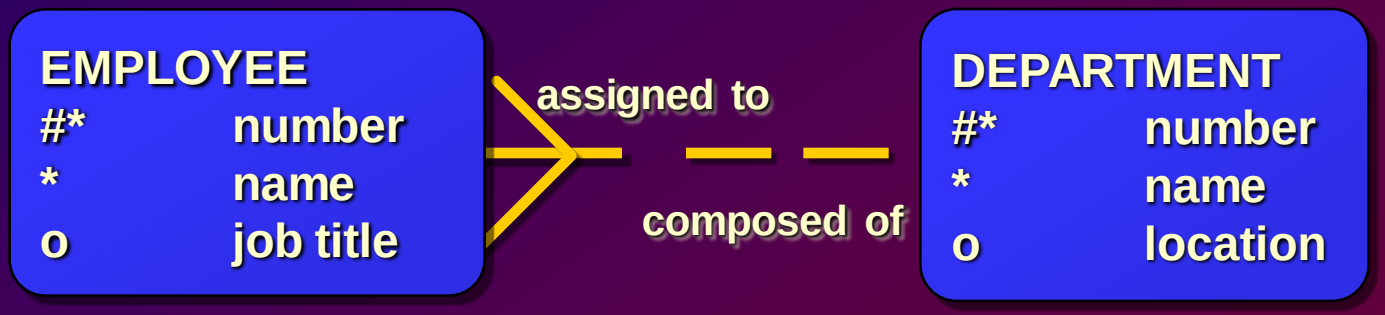
DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

Data Models



Entity Relationship Model

- Create an entity relationship diagram from business specifications or narratives

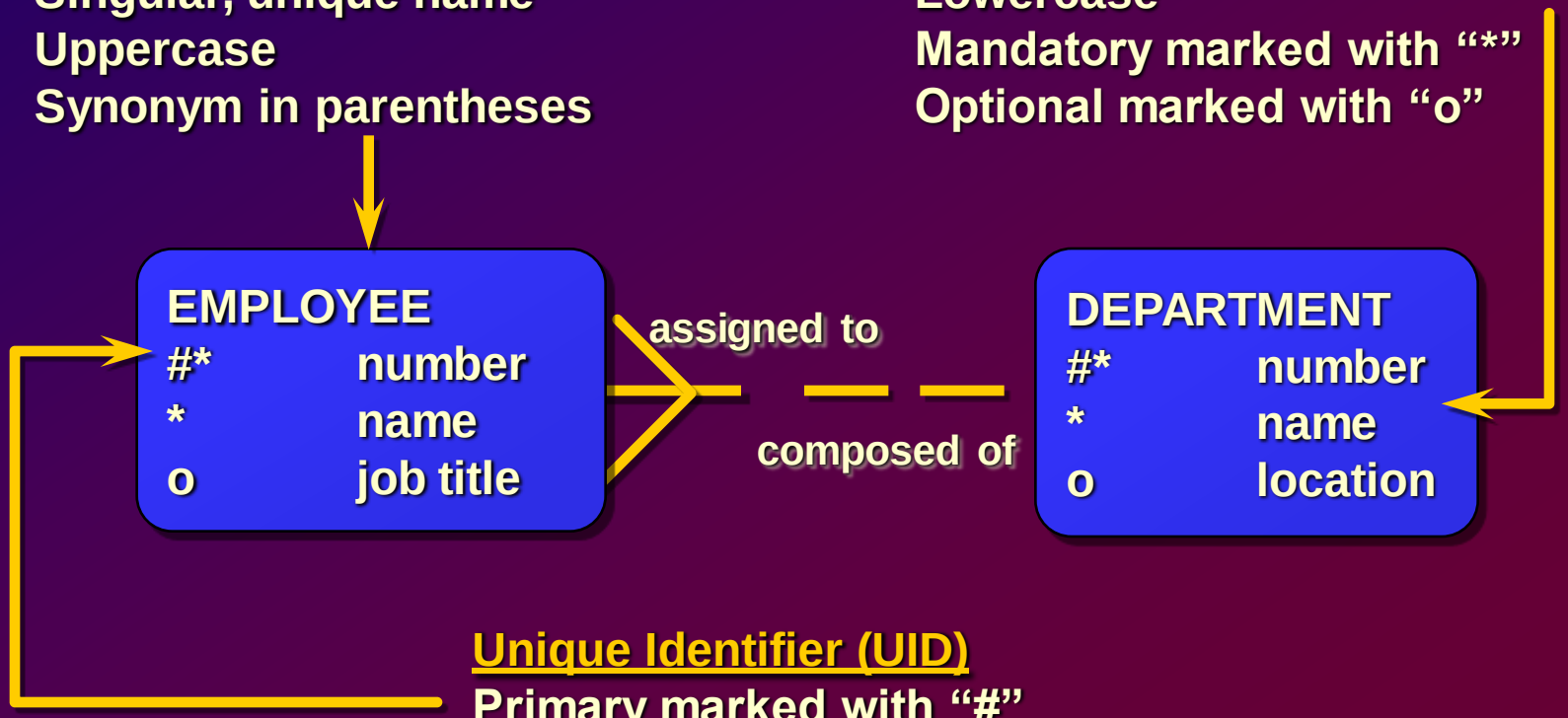


- Scenario
  - "... Assign one or more employees to a department ..."
  - "... Some departments do not yet have assigned employees ..."

Entity Relationship Modeling Conventions

Entity
Soft box
Singular, unique name
Uppercase
Synonym in parentheses

Attribute
Singular name
Lowercase
Mandatory marked with "*"
Optional marked with "o"



Unique Identifier (UID)
Primary marked with "#"
Secondary marked with "(#)"

Relational Database Terminology

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
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7902	FORD	ANALYST	7566	03-DEC-81	3000		20
7369	SMITH	CLERK	7902	17-DEC-80	800		20
7788	SCOTT	ANALYST	7566	09-DEC-82	3000		20
7876	ADAMS	CLERK	7788	12-JAN-83	1100		20
7934	MILLER	CLERK	7782	23-JAN-82	1300		10

Relating Multiple Tables

- Each row of data in a table is uniquely identified by a primary key (PK).
- You can logically relate data from multiple tables using foreign keys (FK).

Table Name: **EMP**

EMPNO	ENAME	JOB	DEPTNO
7839	KING	PRESIDENT	10
7698	BLAKE	MANAGER	30
7782	CLARK	MANAGER	10
7566	JONES	MANAGER	20

Table Name: **DEPT**

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

Primary key **Foreign key** **Primary key**

Relational Database Properties

A relational database

- Can be accessed and modified by executing structured query language (SQL) statements
- Contains a collection of tables with no physical pointers
- Uses a set of operators

Communicating with a RDBMS Using SQL

SQL statement is entered

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

Statement is sent to database

Data is displayed

```
LOC
-------------
NEW YORK
DALLAS
CHICAGO
BOSTON
```

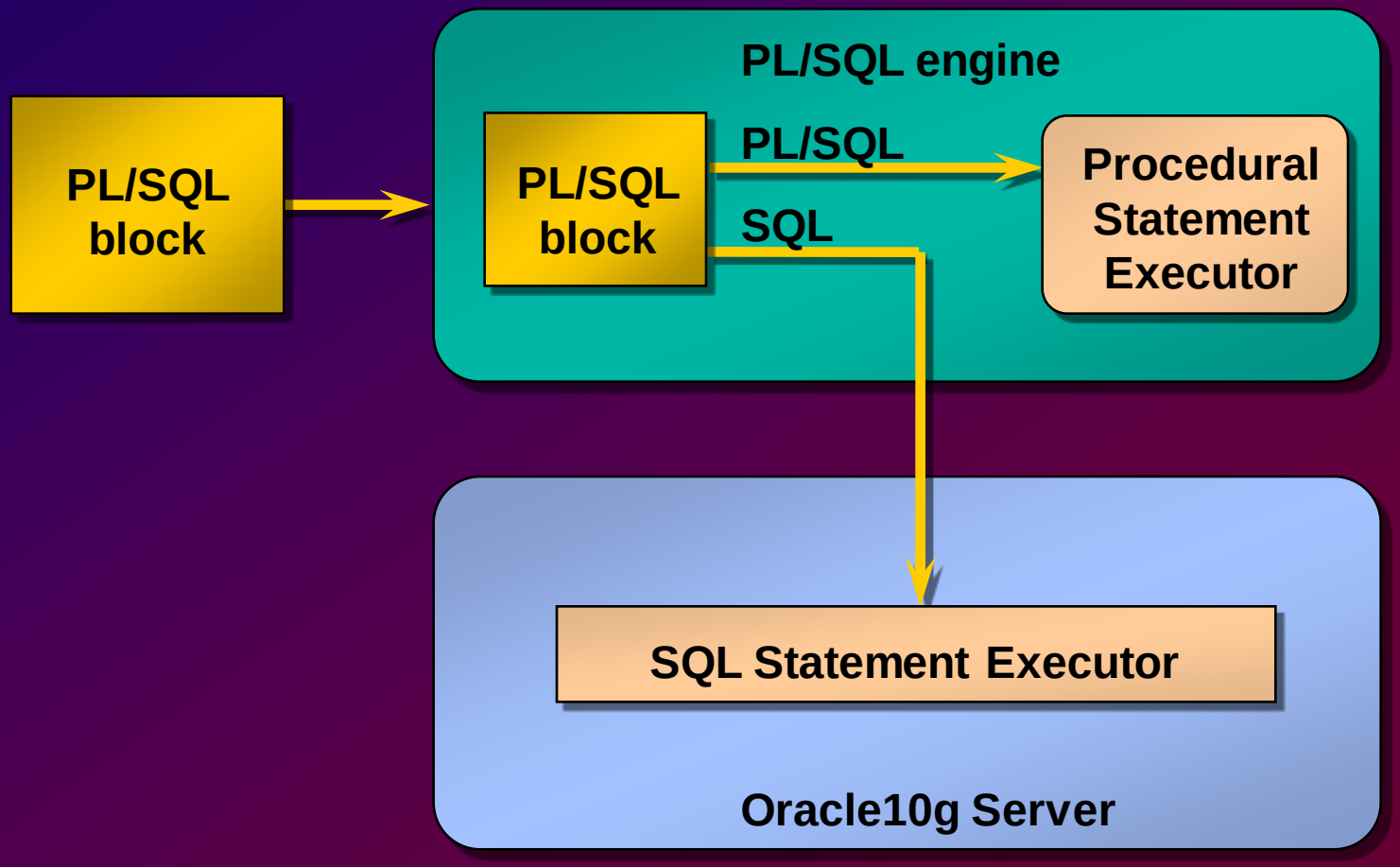
SQL Statements

SELECT	Data retrieval
INSERT UPDATE DELETE	Data manipulation language (DML)
CREATE ALTER DROP RENAME TRUNCATE	Data definition language (DDL)
COMMIT ROLLBACK SAVEPOINT	Transaction control
GRANT REVOKE	Data control language (DCL)

What Is PL/SQL?

- PL/SQL is an extension to SQL with design features of programming languages.
- Data manipulation and query statements of SQL are included within procedural units of code.

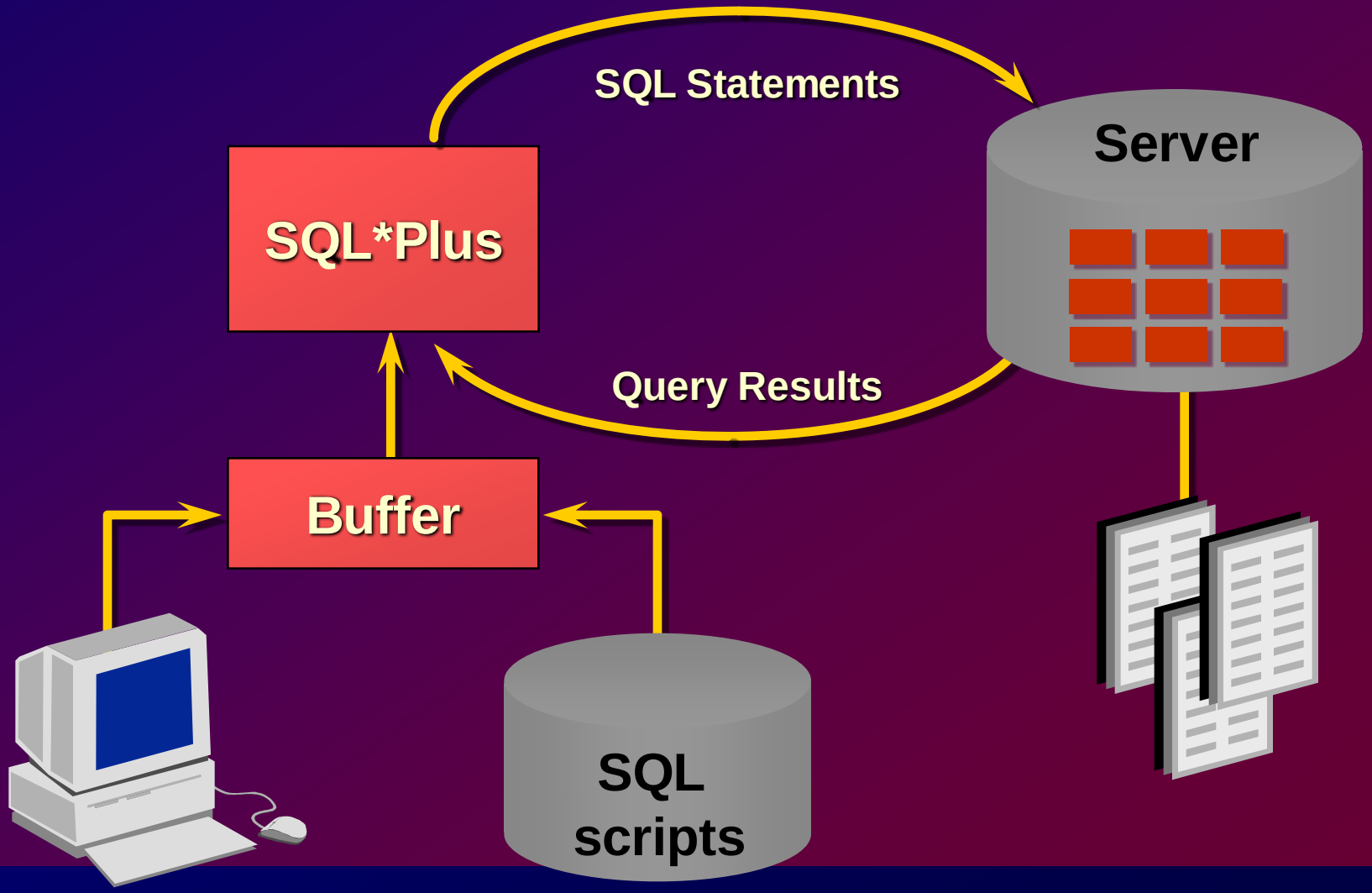
PL/SQL Environment



Benefits of PL/SQL

- It is portable.
- You can declare identifiers.
- You can program with procedural language control structures.
- It can handle errors.

SQL and SQL*Plus Interaction



SQL Statements Versus SQL*Plus Commands

SQL

- A language
- ANSI standard
- Keyword cannot be abbreviated
- Statements manipulate data and table definitions in the database

SQL*Plus

- An environment
- Oracle proprietary
- Keywords can be abbreviated
- Commands do not allow manipulation of values in the database

SQL statements → SQL buffer

SQL*Plus commands → ~~SQL*Plus buffer~~

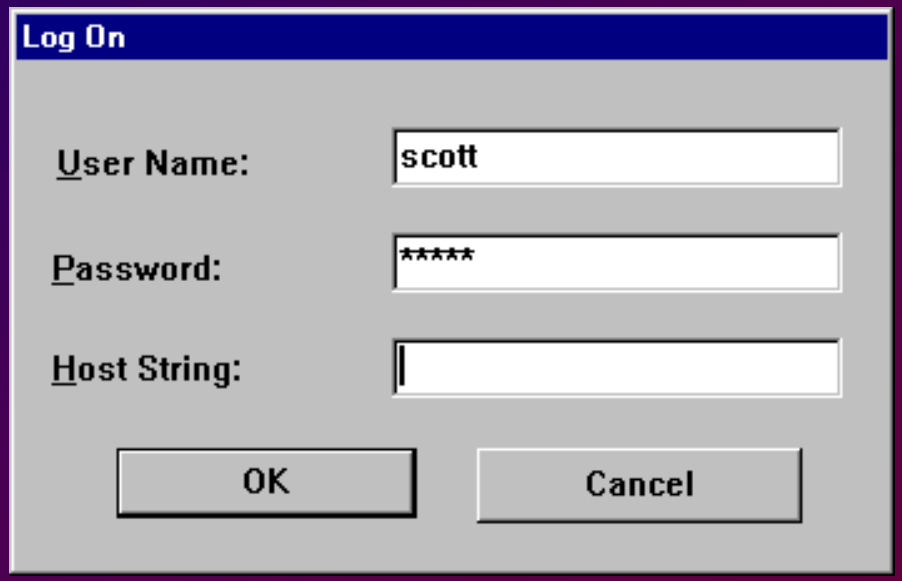
ORACLE®

Overview of SQL*Plus

- Log in to SQL*Plus.
- Describe the table structure.
- Edit your SQL statement.
- Execute SQL from SQL*Plus.
- Save SQL statements to files and append SQL statements to files.
- Execute saved files.
- Load commands from file to buffer to edit.

Logging In to SQL*Plus

- From Windows environment:



- From command line:

sqlplus [*username*[/*password* [@*database*]]]

Tables Used in the Course

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	1500		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	[illegible]		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

Displaying Table Structure

Use the SQL*Plus DESCRIBE command to display the structure of a table.

```
DESC[RIBE] tablename
```

Displaying Table Structure

```
SQL> DESCRIBE dept
```

```
Name                    Null?    Type
----------------------- -------- ----
DEPTNO                  NOT NULL NUMBER(2)
DNAME                            VARCHAR2(14)
LOC                              VARCHAR2(13)
```

Writing Basic SQL Statements

Capabilities of SQL SELECT Statements

Selection

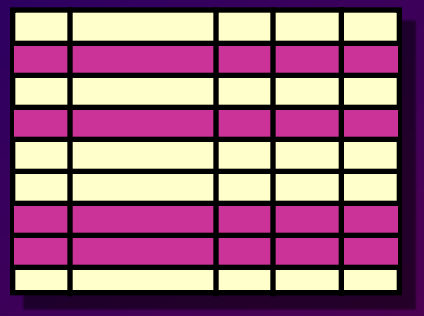


Table 1

Projection

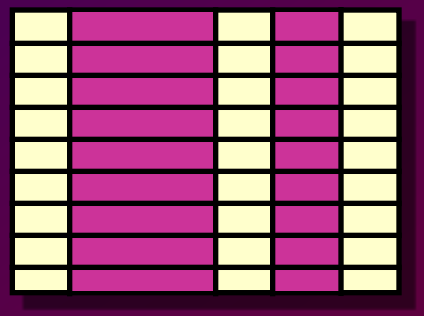


Table 1

Join

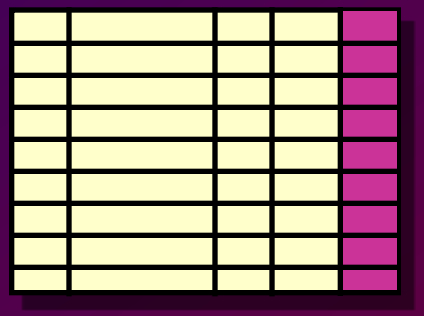


Table 1

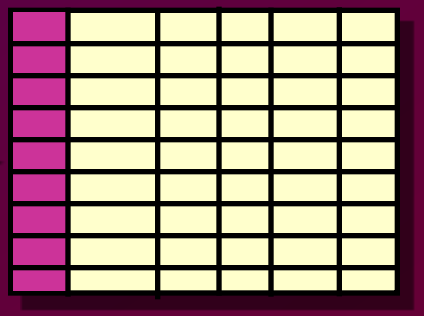


Table 2

Basic SELECT Statement

```
SELECT    [DISTINCT] {*, column [alias],...}
FROM      table;
```

- SELECT identifies *what* columns
- FROM identifies *which* table

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.
- Clauses are usually placed on separate lines.
- Tabs and indents are used to enhance readability.

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
```

Column Label Defaults

- Default justification
  - Left: Date and character data
  - Right: Numeric data
- Default display: Uppercase

Arithmetic Expressions

Create expressions on NUMBER and DATE data types by using arithmetic operators.

Operator	Description
+	Add
-	Subtract
*	Multiply
/	Divide

Using Arithmetic Operators

```
SQL> SELECT ename, sal, sal+300
  2  FROM   emp;
```

```
ENAME             SAL   SAL+300
---------- --------- ---------
KING             5000      5300
BLAKE            2850      3150
CLARK            2450      2750
JONES            2975      3275
MARTIN           1250      1550
ALLEN            1600      1900
...
14 rows selected.
```

Operator Precedence

* / + –

- Multiplication and division take priority over addition and subtraction.
- Operators of the same priority are evaluated from left to right.
- Parentheses are used to force prioritized evaluation and to clarify statements.

Operator Precedence

```
SQL> SELECT ename, sal, 12*sal+100
  2  FROM   emp;
```

```
ENAME             SAL 12*SAL+100
---------- --------- ----------
KING             5000      60100
BLAKE            2850      34300
CLARK            2450      29500
JONES            2975      35800
MARTIN           1250      15100
ALLEN            1600      19300
...
14 rows selected.
```

Using Parentheses

```
SQL> SELECT ename, sal, 12*(sal+100)
  2  FROM   emp;
```

```
ENAME             SAL 12*(SAL+100)
---------- --------- ------------
KING             5000        61200
BLAKE            2850        35400
CLARK            2450        30600
JONES            2975        36900
MARTIN           1250        16200
...
14 rows selected.
```

Defining a Null Value

- A null is a value that is unavailable, unassigned, unknown, or inapplicable.
- A null is not the same as zero or a blank space.

```
SQL> SELECT   ename, job, comm
  2  FROM     emp;
```

```
ENAME      JOB         COMM
---------- --------- ---------
KING       PRESIDENT
BLAKE      MANAGER
...
TURNER     SALESMAN         0
...
14 rows selected.
```

Null Values in Arithmetic Expressions

Arithmetic expressions containing a null value evaluate to null.

```
SQL> select ename NAME, 12*sal+comm
  2  from   emp
  3  WHERE  ename='KING';
```

```
NAME       12*SAL+COMM
---------- -----------
KING
```

Defining a Column Alias

- Renames a column heading
- Is useful with calculations
- Immediately follows column name; optional AS keyword between column name and alias
- Requires double quotation marks if it contains spaces or special characters or is case sensitive

Using Column Aliases

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

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

```
SQL> SELECT ename "Name",
  2         sal*12 "Annual Salary"
  3  FROM   emp;
```

```
Name                 Annual Salary
-------------- --------------
...
```

Concatenation Operator

- Concatenates columns or character strings to other columns
- Is represented by two vertical bars (||)
- Creates a resultant column that is a character expression

Using the Concatenation Operator

```
SQL> SELECT   ename||job AS "Employees"
  2  FROM     emp;
```

```
Employees
-------------------
KINGPRESIDENT
BLAKEMANAGER
CLARKMANAGER
JONESMANAGER
MARTINSALESMAN
ALLENSALESMAN
...
14 rows selected.
```

Literal Character Strings

- A literal is a character, expression, or number included in the SELECT list.
- Date and character literal values must be enclosed within single quotation marks.
- Each character string is output once for each row returned.

Using Literal Character Strings

```
SQL> SELECT ename ||' '||'is a'||' '||job
  2                AS "Employee Details"
  3  FROM   emp;
```

```
Employee Details
-------------------------
KING is a PRESIDENT
BLAKE is a MANAGER
CLARK is a MANAGER
JONES is a MANAGER
MARTIN is a SALESMAN
...
14 rows selected.
```

Duplicate Rows

The default display of queries is all rows, including duplicate rows.

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

```
   DEPTNO
---------
       10
       30
       10
       20
...
14 rows selected.
```


ORACLE®

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
```


ORACLE®

SQL*Plus Editing Commands

- A[PPEND] *text*
- C[HANGE] */ old / new*
- C[HANGE] */ text /*
- CL[EAR] BUFF[ER]
- DEL
- DEL *n*
- DEL *m n*

SQL*Plus Editing Commands

- I[NPUT]
- I[NPUT] *text*
- L[IST]
- L[IST] *n*
- L[IST] *m n*
- R[UN]
- *n*
- *n text*
- 0 *text*

SQL*Plus File Commands

- SAVE *filename*
- GET *filename*
- START *filename*
- @ *filename*
- EDIT *filename*
- SPOOL *filename*
- EXIT

Practice Overview

- Selecting all data from different tables.
- Describing the structure of tables.
- Performing arithmetic calculations and specifying column names.
- Using SQL*Plus editor.

Restricting and Sorting Data

Limiting Rows Using a Selection

EMP

EMPNO	ENAME	JOB	...	DEPTNO
7839	KING	PRESIDENT		10
7698	BLAKE	MANAGER		30
7782	CLARK	MANAGER		10
7566	JONES	MANAGER		20
...				

"...retrieve all employees in department 10"

EMP

EMPNO	ENAME	JOB	...	DEPTNO
7839	KING	PRESIDENT		10
7782	CLARK	MANAGER		10
7934	MILLER	CLERK		10

ORACLE®

Limiting Rows Selected

- Restrict the rows returned by using the WHERE clause.

```
SELECT        [DISTINCT] {*, column [alias], ...}
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
- Default date format is 'DD-MON-YY'

```
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;
```

Lower limit

Higher limit

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

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
```


ORACLE®

Using the LIKE Operator

- Use the LIKE operator to perform wildcard searches of valid search string values.
- Search conditions can contain either literal characters or numbers.
  - (%) denotes zero or many characters
  - (_) denotes one character

```
SQL> SELECT   ename
  2  FROM     emp
  3  WHERE    ename LIKE 'S%';
```

Using the LIKE Operator

- You can combine pattern matching characters.

```
SQL> SELECT   ename
  2  FROM     emp
  3  WHERE    ename LIKE '_A%';
```

```
ENAME
----------
JAMES
WARD
```

- You can use the ESCAPE identifier to search for “%” or “_”.

Using the IS NULL Operator

Test for null values with the IS NULL operator

```
SQL> SELECT   ename, mgr
  2  FROM     emp
  3  WHERE    mgr IS NULL;
```

```
ENAME          MGR
---------- ---------
KING
```

Logical Operators

Operator	Meaning
AND	Returns TRUE if *both* component conditions are TRUE
OR	Returns TRUE if *either* component condition is TRUE
NOT	Returns TRUE if the following condition is FALSE

Using the AND Operator

AND requires both conditions to be TRUE.

```
SQL> SELECT empno, ename, job, sal
  2  FROM   emp
  3  WHERE  sal>=1100
  4  AND    job='CLERK';
```

```
     EMPNO ENAME      JOB         SAL
---------- ---------- --------- ---------
      7876 ADAMS      CLERK        1100
      7934 MILLER     CLERK        1300
```

Using the OR Operator

OR requires either condition to be TRUE.

```
SQL> SELECT empno, ename, job, sal
  2  FROM   emp
  3  WHERE  sal>=1100
  4  OR     job='CLERK';
```

```
     EMPNO ENAME      JOB         SAL
--------- ---------- --------- ---------

      7839 KING       PRESIDENT      5000
      7698 BLAKE      MANAGER        2850
      7782 CLARK      MANAGER        2450
      7566 JONES      MANAGER        2975
      7654 MARTIN     SALESMAN       1250
...
14 rows selected.
```

Using the NOT Operator

```
SQL> SELECT ename, job
  2  FROM   emp
  3  WHERE  job NOT IN ('CLERK','MANAGER','ANALYST');
```

```
ENAME      JOB
---------- ---------
KING       PRESIDENT
MARTIN     SALESMAN
ALLEN      SALESMAN
TURNER     SALESMAN
WARD       SALESMAN
```

Rules of Precedence

Order Evaluated	Operator
1	All comparison operators
2	NOT
3	AND
4	OR

Override rules of precedence by using parentheses.

Rules of Precedence

```
SQL> SELECT ename, job, sal
  2  FROM   emp
  3  WHERE  job='SALESMAN'
  4  OR     job='PRESIDENT'
  5  AND    sal>1500;
```

```
ENAME      JOB              SAL
---------- --------- ---------
KING       PRESIDENT      5000
MARTIN     SALESMAN       1250
ALLEN      SALESMAN       1600
TURNER     SALESMAN       1500
WARD       SALESMAN       1250
```

Rules of Precedence

Use parentheses to force priority.

```
SQL> SELECT   ename, job, sal
  2  FROM     emp
  3  WHERE    (job='SALESMAN'
  4  OR       job='PRESIDENT')
  5  AND      sal>1500;
```

```
ENAME      JOB              SAL
---------- --------- ---------
KING       PRESIDENT      5000
ALLEN      SALESMAN       1600
```

ORDER BY Clause

- Sort rows with the ORDER BY clause
  - ASC: ascending order, default
  - DESC: descending order
- The ORDER BY clause comes last in the SELECT statement.

```
SQL> SELECT    ename, job, deptno, hiredate
  2  FROM      emp
  3  ORDER BY hiredate;
```

```
ENAME      JOB           DEPTNO HIREDATE
---------- --------- --------- ---------
SMITH      CLERK             20 17-DEC-80
ALLEN      SALESMAN          30 20-FEB-81
...
14 rows selected.
```

Sorting in Descending Order

```
SQL> SELECT    ename, job, deptno, hiredate
  2  FROM      emp
  3  ORDER BY hiredate DESC;
```

```
ENAME      JOB           DEPTNO HIREDATE
---------- --------- ---------- ---------
ADAMS      CLERK             20 12-JAN-83
SCOTT      ANALYST           20 09-DEC-82
MILLER     CLERK             10 23-JAN-82
JAMES      CLERK             30 03-DEC-81
FORD       ANALYST           20 03-DEC-81
KING       PRESIDENT         10 17-NOV-81
MARTIN     SALESMAN          30 28-SEP-81
...
14 rows selected.
```

Sorting by Column Alias

```
SQL> SELECT   empno, ename, sal*12 annsal
  2  FROM     emp
  3  ORDER BY annsal;
```

```
     EMPNO ENAME          ANNSAL
--------- ---------- ---------
      7369 SMITH           9600
      7900 JAMES          11400
      7876 ADAMS          13200
      7654 MARTIN         15000
      7521 WARD           15000
      7934 MILLER         15600
      7844 TURNER         18000
...
14 rows selected.
```

Sorting by Multiple Columns

- The order of ORDER BY list is the order of sort.

```
SQL> SELECT   ename, deptno, sal
  2  FROM     emp
  3  ORDER BY deptno, sal DESC;
```

```
ENAME          DEPTNO        SAL
---------- ---------- ----------
KING               10       5000
CLARK              10       2450
MILLER             10       1300
FORD               20       3000
...
14 rows selected.
```

- You can sort by a column that is not in the SELECT list.

Summary

```
SELECT        [DISTINCT] {*, column [alias], ...}
FROM          table
[WHERE        condition(s)]
[ORDER BY     {column, expr, alias} [ASC|DESC]];
```

Practice Overview

- Selecting data and changing the order of rows displayed
- Restricting rows by using the WHERE clause
- Using the double-quotation-marks in column aliases