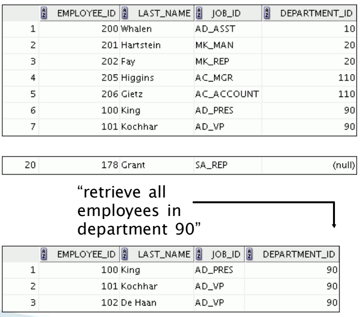
Restricting and Sorting Data

Objectives

- After completing this , you should be able to do the following:
  - Limit the rows that are retrieved by a query
  - Sort the rows that are retrieved by a query

Limiting Rows Using a Selection

EMPLOYEES



	EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
1	200	Whalen	AD_ASST	10
2	201	Hartstein	MK_MAN	20
3	202	Fay	MK_REP	20
4	205	Higgins	AC_MGR	110
5	206	Gietz	AC_ACCOUNT	110
6	100	King	AD_PRES	90
7	101	Kochhar	AD_VP	90
20	178	Grant	SA_REP	(null)

"retrieve all employees in department 90"

	EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
1	100	King	AD_PRES	90
2	101	Kochhar	AD_VP	90
3	102	De Haan	AD_VP	90

Limiting the Rows that Are Selected

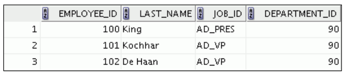
- Restrict the rows that are returned by using the WHERE clause:

```
SELECT *|{[DISTINCT] column|expression [alias],...}
FROM   table
[WHERE  condition(s)];
```

- The WHERE clause follows the FROM clause.
- The WHERE clause can compare values in columns, literal values, arithmetic expressions, or functions. It consists of three elements:
  - Column name
  - Comparison condition
  - Column name, constant, or list of values

Using the WHERE Clause

```
SELECT employee_id, last_name, job_id, department_id
FROM   employees
WHERE  department_id = 90 ;
```



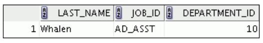
	EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
1	100	King	AD_PRES	90
2	101	Kochhar	AD_VP	90
3	102	De Haan	AD_VP	90

- w1

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.
- The default date format is DD-MON-RR.

```
SELECT last_name, job_id, department_id
FROM   employees
WHERE  last_name = 'Whalen' ;
```



	LAST_NAME	JOB_ID	DEPARTMENT_ID
1	Whalen	AD_ASST	10

- W2

Character Strings and Dates

- SELECT last_name, job_id, department_id
  FROM employees
  WHERE last_name = 'WHALEN';

- No rows are returned because the `EMPLOYEES` table stores all the last names in mixed case

Note:

- The Oracle Database stores dates in an internal numeric format, representing the century, year, month, day, hours, minutes, and seconds. The default date display is DD-MON-RR.

Comparison Conditions

Operator	Meaning
=	Equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
<>	Not equal to
`BETWEEN ...AND...`	Between two values (inclusive)
`IN(set)`	Match any of a list of values
`LIKE`	Match a character pattern
`IS NULL`	Is a null value

- Comparison conditions are used in conditions that compare one expression to another value or expression.
- They are used in the `WHERE` clause in the following format:

Syntax

```
WHERE expr operator value
```

Comparison Conditions

Example

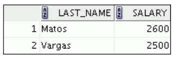
```
... WHERE hire_date = '01-JAN-95'
... WHERE salary >= 6000
... WHERE last_name = 'Smith'
```

- Is an alias can be used in the WHERE clause?
- An alias cannot be used in the `WHERE` clause.

- **Note**: The symbols `!=` and `^=` can also represent the *not equal to* condition.

Using Comparison Conditions

```
SELECT last_name, salary
FROM   employees
WHERE  salary <= 3000 ;
```



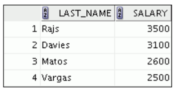
	LAST_NAME	SALARY
1	Matos	2600
2	Vargas	2500

- W3

Using the `BETWEEN` Condition

- Use the `BETWEEN` condition to display rows based on a range of values:W4

```
SELECT last_name, salary
FROM   employees
WHERE  salary BETWEEN 2500 AND 3500 ;
```



	LAST_NAME	SALARY
1	Rajs	3500
2	Davies	3100
3	Matos	2600
4	Vargas	2500

Lower limit

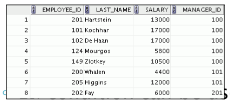
Upper limit

- Is `BETWEEN` condition work on character values?
- Yes:W5
- SELECT last_name
  FROM employees
  WHERE last_name BETWEEN 'King' AND 'Smith';

Using the `IN` Condition

- Use the `IN` membership condition to test for values in a list:W6

```
SELECT employee_id, last_name, salary, manager_id
FROM   employees
WHERE  manager_id IN (100, 101, 201) ;
```



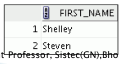
	EMPLOYEE_ID	LAST_NAME	SALARY	MANAGER_ID
1	201	Hartstein	13000	100
2	101	Kochhar	17000	100
3	102	De Haan	17000	100
4	124	Mourgos	5800	100
5	149	Zlotkey	10500	100
6	200	Whalen	4400	101
7	205	Higgins	12000	101
8	202	Fay	6000	201

- sed with any data type W7
  - SELECT employee_id, manager_id, department_id
  - FROM employees
  - WHERE last_name IN ('Hartstein', 'Vargas');
- If characters or dates are used in the list, they must be enclosed in single quotation marks (`''`).

Using the LIKE Condition

- Use the `LIKE` condition to perform wildcard searches of valid search string values.
  - **wildcard searches:** The character pattern-matching operation is referred to as a *wildcard* search
- Search conditions can contain either literal characters or numbers:
  - `%` denotes zero or many characters.
  - _ denotes one character.
- L1

```
SELECT    first_name
FROM      employees
WHERE     first_name LIKE 'S%' ;
```



	FIRST_NAME
1	Shelley
2	Steven

- The `LIKE` condition can be used as a shortcut for some

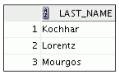
Example

- displays the last names and hire dates of all employees who joined between January 1995 and December 1995: `BETWEEN` comparisons
- L2
  - SELECT last_name, hire_date
FROM employees
WHERE hire_date LIKE '%95';

Using the LIKE Condition

- You can combine pattern-matching characters:L3

```
SELECT last_name
FROM   employees
WHERE  last_name LIKE '_o%' ;
```



	LAST_NAME
1	Kochhar
2	Lorentz
3	Mourgos

- The `%` and _ symbols can be used in any combination with literal characters.

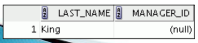
Escape wildcard characters

- When you need to have an exact match for the actual *%* and _ characters
- Write a query to search for strings that contain `SA_`
  - SELECT employee_id, last_name, job_id FROM employees WHERE job_id LIKE '%SA_%' ESCAPE '\'; L4
- The `ESCAPE` option identifies the backslash (\) as the escape character.
- In the pattern, the escape character precedes the underscore (_).

Using the NULL Conditions

- Retrieves the last names and managers of all employees who do not have a manager.
- Cannot test null value with = because a null cannot be equal or unequal to any value
- ???
- Test for nulls with the `IS NULL` operator.N1

```
SELECT last_name, manager_id
FROM   employees
WHERE  manager_id IS NULL ;
```



	LAST_NAME	MANAGER_ID
1	King	(null)

`NULL` Conditions

- The `NULL` conditions include the `IS NULL` condition and the `IS NOT NULL` condition.N2

```
SELECT last_name, manager_id
FROM   employees
WHERE  manager_id IS NOT NULL ;
```

- Display last name, job ID, and commission for all employees who are not entitled to receive a commission.N3

```
SELECT last_name, job_id, commission_pct
FROM   employees
WHERE commission_pct IS NULL;
```

Logical Conditions

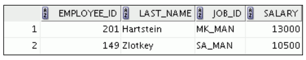
- A logical condition combines the result of two component conditions to produce a single result based on those conditions or it inverts the result of a single condition.
- A row is returned only if the overall result of the condition is true.

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:O1

```
SELECT employee_id, last_name, job_id, salary
FROM   employees
WHERE  salary >=10000
AND    job_id LIKE '%MAN%' ;
```



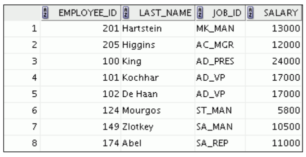
	EMPLOYEE_ID	LAST_NAME	JOB_ID	SALARY
1	201	Hartstein	MK_MAN	13000
2	149	Zlotkey	SA_MAN	10500

- No rows are returned if ‘MAN’ is not uppercase.

Using the OR Operator

- OR requires either condition to be true:O2

```
SELECT employee_id, last_name, job_id, salary
FROM   employees
WHERE  salary >= 10000
OR     job_id LIKE '%MAN%' ;
```



	EMPLOYEE_ID	LAST_NAME	JOB_ID	SALARY
1	201	Hartstein	MK_MAN	13000
2	205	Higgins	AC_MGR	12000
3	100	King	AD_PRES	24000
4	101	Kochhar	AD_VP	17000
5	102	De Haan	AD_VP	17000
6	124	Mourgos	ST_MAN	5800
7	149	Zlotkey	SA_MAN	10500
8	174	Abel	SA_REP	11000

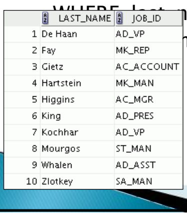
Using the NOT Operator

Note: O4

- SQL

```
SELECT last_name, job_id
FROM   employees
WHERE  job_id
       NOT IN ('IT_PROG', 'ST_CLERK', 'SA_REP') ;
```

- ... WHERE job_id NOT IN ('AC_ACCOUNT', 'AD_VP')
- ... WHERE salary NOT BETWEEN 10000 AND 15000
- ame NOT LIKE '%A%'
- ission_pct IS NOT NULL



	LAST_NAME	JOB_ID
1	De Haan	AD_VP
2	Fay	MK_REP
3	Gietz	AC_ACCOUNT
4	Hartstein	MK_MAN
5	Higgins	AC_MGR
6	King	AD_PRES
7	Kochhar	AD_VP
8	Mourgos	ST_MAN
9	Whalen	AD_ASST
10	Zlotkey	SA_MAN

Rules of Precedence

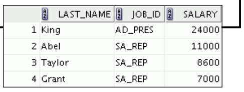
Operator	Meaning
1	Arithmetic operators
2	Concatenation operator
3	Comparison conditions
4	IS [NOT] NULL, LIKE, [NOT] IN
5	[NOT] BETWEEN
6	Not equal to
7	NOT logical condition
8	AND logical condition
9	OR logical condition

- You can use parentheses to override rules of precedence.

Rules of Precedence

```
SELECT last_name, job_id, salary
FROM   employees
WHERE  job_id = 'SA_REP'
OR     job_id = 'AD_PRES'
AND    salary > 15000;
```

1

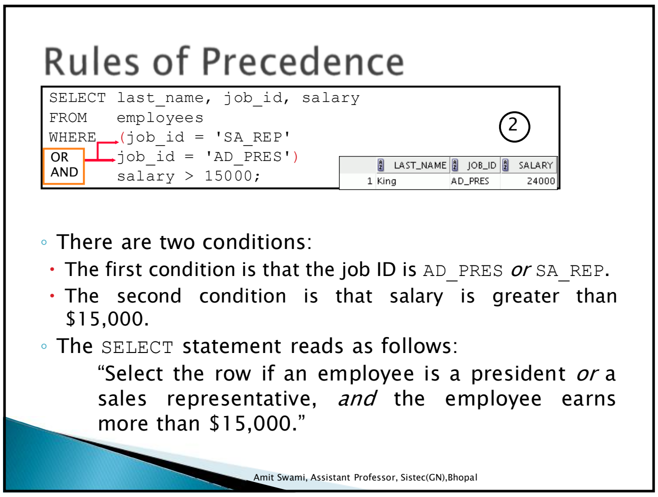


	LAST_NAME	JOB_ID	SALARY
1	King	AD_PRES	24000
2	Abel	SA_REP	11000
3	Taylor	SA_REP	8600
4	Grant	SA_REP	7000

- O4,O5
  - There are two conditions:
    - The first condition is that the job ID is AD PRES *and* the

```
SELECT last_name, job_id, salary
FROM   employees
WHERE    job_id = 'SA_REP'
AND      job_id = 'AD_PRES'
OR       salary > 15000;
```

and earns more than $15,000, *or* if the employee is a sales representative.”



Rules of Precedence

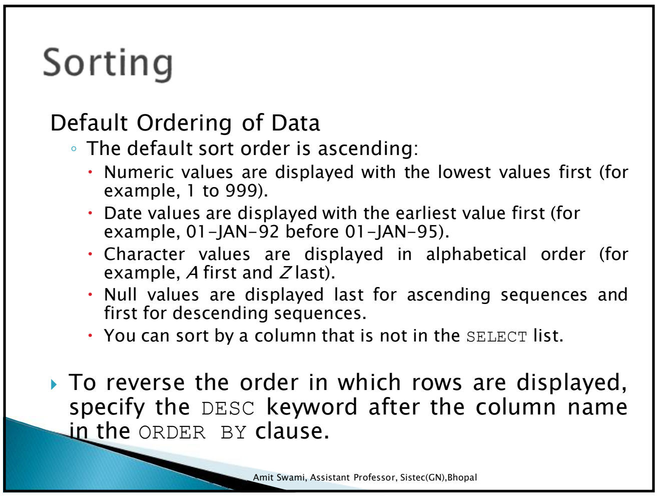
```
SELECT last_name, job_id, salary
FROM   employees
WHERE  (job_id = 'SA_REP'
OR     job_id = 'AD_PRES')
AND    salary > 15000;
```

2

	LAST_NAME	JOB_ID	SALARY
1	King	AD_PRES	24000

- There are two conditions:
  - The first condition is that the job ID is `AD_PRES` *or* `SA_REP`.
  - The second condition is that salary is greater than $15,000.
- The `SELECT` statement reads as follows:

  "Select the row if an employee is a president *or* a sales representative, *and* the employee earns more than $15,000."

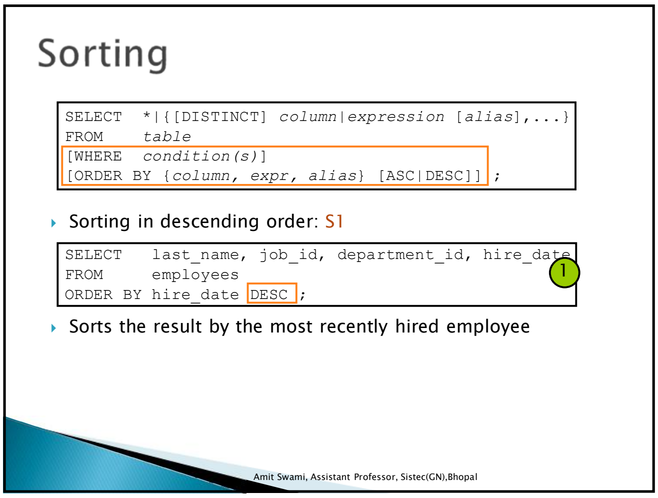


Sorting

Default Ordering of Data

- The default sort order is ascending:
  - Numeric values are displayed with the lowest values first (for example, 1 to 999).
  - Date values are displayed with the earliest value first (for example, 01-JAN-92 before 01-JAN-95).
  - Character values are displayed in alphabetical order (for example, *A* first and *Z* last).
  - Null values are displayed last for ascending sequences and first for descending sequences.
  - You can sort by a column that is not in the `SELECT` list.

- To reverse the order in which rows are displayed, specify the `DESC` keyword after the column name in the `ORDER BY` clause.



Sorting

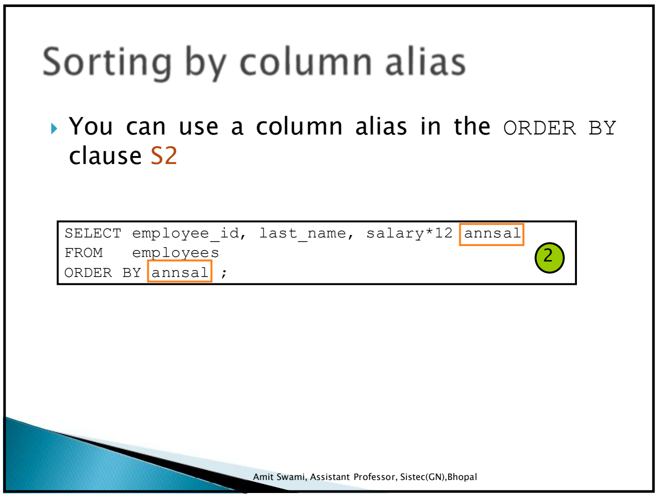
```
SELECT   *|{[DISTINCT] column|expression [alias],...}
FROM     table
[WHERE   condition(s)]
[ORDER BY {column, expr, alias} [ASC|DESC]] ;
```

- Sorting in descending order: S1

```
SELECT    last_name, job_id, department_id, hire_date
FROM      employees
ORDER BY hire_date DESC ;
```

1

- Sorts the result by the most recently hired employee

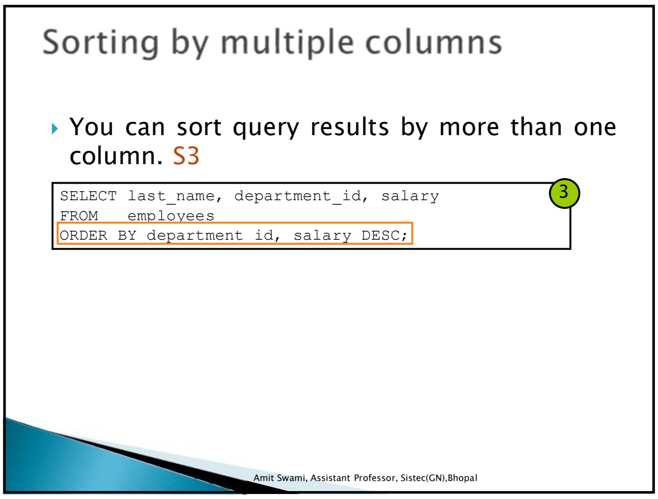


Sorting by column alias

- You can use a column alias in the `ORDER BY` clause S2

```
SELECT employee_id, last_name, salary*12 annsal
FROM   employees
ORDER BY annsal ;
```

2



Sorting by multiple columns

- You can sort query results by more than one column. S3

```
SELECT last_name, department_id, salary
FROM   employees
ORDER BY department_id, salary DESC;
```

3