THE UNIVERSITY OF WESTERN AUSTRALIA

CITS1402 Relational Database Management Systems

Week 4—SQL: Data Manipulation – part2

Sem1/2024

Dr Mehwish Nasim/ CSSE

cits1402-sem1-pmc@uwa.edu.au

Example 6.7 Range Search Condition - BETWEEN

List all staff with a salary between 20,000 and 30,000.

BETWEEN test includes the endpoints of range.

staffNo	fName	lName	position	salary
SL21	John	White	Manager	30000.00
SG5	Susan	Brand	Manager	24000.00

```
SELECT staffNo, fName, lName, position, salary
  FROM Staff
  WHERE salary BETWEEN 20000 AND 30000;
```

Example 6.7 Range Search Condition NOT

Also a negated version NOT BETWEEN.

BETWEEN does not add much to SQL's expressive power. Could also write:

```
SELECT staffNo, fName, lName, position, salary
FROM Staff
WHERE salary>=20000 AND salary <= 30000;
```

Useful, though, for a range of values.

Example 6.8 Set Membership - IN

List all managers and supervisors.

```
SELECT staffNo, fName, lName, position
FROM Staff
WHERE position IN ('Manager', 'Supervisor');
```

Table 5.8 Result table for Example 5.8.

staffNo	fName	lName	position
SL21	John	White	Manager
SG14	David	Ford	Supervisor
SG5	Susan	Brand	Manager

Example 6.8 Set Membership NOT

There is a negated version (NOT IN).

IN does not add much to SQL's expressive power. Could have expressed this as:

```
SELECT staffNo, fName, lName, position
FROM Staff
WHERE position='Manager' OR
      position='Supervisor';
```

IN is more efficient when set contains many values.

Example 6.9 Pattern Matching LIKE

Find all owners with the string 'Glasgow' in their address.

```
SELECT ownerNo, fName, lName, address, telNo
FROM PrivateOwner
WHERE address LIKE '%Glasgow%';
```

Table 5.9 Result table for Example 5.9.

ownerNo	fName	lName	address	telNo
CO87	Carol	Farrel	6 Achray St, Glasgow G32 9DX	0141-357-7419
CO40	Tina	Murphy	63 Well St, Glasgow G42	0141-943-1728
CO93	Tony	Shaw	12 Park Pl, Glasgow G4 0QR	0141-225-7025

Example 6.9 Pattern Matching LIKE

SQL has two special pattern matching symbols:

- **%: sequence of zero or more characters;**
- **_ (underscore): any single character.**

LIKE '%Glasgow%' means a sequence of characters of any length containing '*Glasgow*'.

Example 6.10 NULL Search Condition

List details of all viewings on property PG4 where a comment has not been supplied.

There are 2 viewings for property PG4, one with and one without a comment.

Have to test for null explicitly using special keyword IS NULL:

```
SELECT clientNo, viewDate
FROM Viewing
WHERE propertyNo = 'PG4' AND
comment IS NULL;
```

Example 6.10 NULL Search Condition

clientNo	viewDate
CR56	26-May-04

Negated version (IS NOT NULL) can test for non-null values.

Objectives

How to retrieve data from database using SELECT and:

- Use compound WHERE conditions.
- **Use aggregate functions.**
- Sort query results using ORDER BY.
- Group data using GROUP BY and HAVING.
- Use subqueries.
- Join tables together.
- Perform set operations (UNION, INTERSECT, EXCEPT).

SELECT Statement - Aggregates

ISO standard defines five aggregate functions:

COUNT returns number of values in specified column.

SUM returns sum of values in specified column.

AVG returns average of values in specified column.

MIN returns smallest value in specified column.

MAX returns largest value in specified column.

Example 6.13 Use of COUNT(*)

How many properties cost more than £350 per month to rent?

```
SELECT COUNT(*) AS myCount
FROM PropertyForRent
WHERE rent > 350;
```

myCount
5

SELECT Statement - Aggregates

Each operates on a single column of a table and returns a single value.

COUNT, MIN, and MAX apply to numeric and non-numeric fields

SUM and AVG may be used on numeric fields only.

Each function eliminates nulls first and operates only on remaining non-null values.

COUNT(*) that applies to all rows, null included

SELECT Statement - Aggregates

COUNT(*) counts all rows of a table, regardless of whether nulls or duplicate values occur.

Can use DISTINCT before column name to eliminate duplicates.

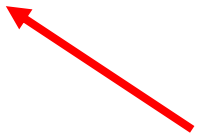
DISTINCT has no effect with MIN/MAX, but may have with SUM/AVG.

SELECT Statement - Aggregates

Aggregate functions can be used only in SELECT list and in HAVING clause.

If SELECT list includes an aggregate function and there is no GROUP BY clause, SELECT list cannot reference a column out with an aggregate function. For example, the following is illegal:

```
SELECT staffNo, COUNT(salary)
FROM Staff;
```



No aggregate function on staffNo

Warning! May work, but is wrong!

Example 6.14 Use of COUNT(DISTINCT)

How many different properties viewed in May '04?

```
SELECT COUNT(DISTINCT propertyNo) AS myCount
    FROM Viewing
    WHERE viewDate BETWEEN '1-May-04'
                AND '31-May-04';
```

myCount
2

Example 6.15 Use of COUNT and SUM

Find number of Managers and sum of their salaries.

```
SELECT COUNT(staffNo) AS myCount,
       SUM(salary) AS mySum
FROM Staff
WHERE position = 'Manager';
```

myCount	mySum
2	54000.00

Example 6.16 Use of MIN, MAX, AVG

Find minimum, maximum, and average staff salary.

```
SELECT MIN(salary) AS myMin,
       MAX(salary) AS myMax,
       AVG(salary) AS myAvg
FROM Staff;
```

myMin	myMax	myAvg
9000.00	30000.00	17000.00

Objectives

How to retrieve data from database using SELECT and:

- **Use compound WHERE conditions.**
- **Use aggregate functions.**
- **Sort query results using ORDER BY.**
- **Group data using GROUP BY and HAVING.**
- **Use subqueries.**
- **Join tables together.**
- **Perform set operations (UNION, INTERSECT, EXCEPT).**

Objectives

How to retrieve data from database using SELECT and:

- **Use compound WHERE conditions.**
- **Use aggregate functions.**
- **Sort query results using ORDER BY.**
- **Group data using GROUP BY and HAVING.**
- **Use subqueries.**
- **Join tables together.**
- **Perform set operations (UNION, INTERSECT, EXCEPT).**

SELECT Statement

```
SELECT [DISTINCT | ALL]
        {* | [columnExpression [AS newName]] [,...] }
FROM            TableName [alias] [, ...]
[WHERE          condition]
[GROUP BY       columnList]
[HAVING         condition]
[ORDER BY       columnList]
```

SELECT Statement

```
SELECT [DISTINCT | ALL]
        {* | [columnExpression [AS newName]] [,...] }
FROM            TableName [alias] [, ...]
[WHERE          condition]
[GROUP BY       columnList]
[HAVING         condition]
[ORDER BY       columnList]
```

SELECT Statement

SELECT	Specifies which columns are to appear in output.
FROM	Specifies table(s) to be used.
WHERE	Filters rows.
GROUP BY	Forms groups of rows with same column value.
HAVING	Filters groups subject to some condition.
ORDER BY	Specifies the order of the output.

SELECT Statement…wait a minute!

Before we start doing SQL statements…

Know thy
SCHEMA

DreamHome Database

Branch (branchNo, street, city, postcode)
Staff (staffNo, fName, lName, position, sex, DOB, salary, branchNo)
PropertyForRent (propertyNo, street, city, postcode, type, rooms, rent, ownerNo, staffNo, branchNo)
Client (clientNo, fName, lName, telNo, prefType, maxRent, email)
PrivateOwner (ownerNo, fName, lName, address, telNo, email, password)
Viewing (clientNo, propertyNo, viewDate, comment)
Registration (clientNo, branchNo, staffNo, dateJoined)

Example 6.1 All Columns, All Rows

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000.00	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000.00	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000.00	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000.00	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000.00	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000.00	B005

Objectives

How to retrieve data from database using SELECT and:

- Use compound WHERE conditions.
- Use aggregate functions.
- Sort query results using ORDER BY.
- Group data using GROUP BY and HAVING.
- Use subqueries.
- Join tables together.
- Perform set operations (UNION, INTERSECT, EXCEPT).

SELECT Statement

```
SELECT [DISTINCT | ALL]
        {* | [columnExpression [AS newName]] [,...] }
FROM    TableName [alias] [, ...]
[WHERE  condition]
[GROUP BY   columnList]
[HAVING     condition]
[ORDER BY   columnList]
```

SELECT Statement

SELECT	Specifies which columns are to appear in output.
FROM	Specifies table(s) to be used.
WHERE	Filters rows.
GROUP BY	Forms groups of rows with same column value.
HAVING	Filters groups subject to some condition.
ORDER BY	Specifies the order of the output.

Example 6.11 Single Column Ordering

List salaries for all staff, arranged in descending order of salary.

staffNo	fName	lName	salary
SL21	John	White	30000.00
SG5	Susan	Brand	24000.00
SG14	David	Ford	18000.00
SG37	Ann	Beech	12000.00
SA9	Mary	Howe	9000.00
SL41	Julie	Lee	9000.00

Example 6.11 Single Column Ordering

List salaries for all staff, arranged in descending order of salary.

```
SELECT staffNo, fName,
       lName, salary
FROM Staff
ORDER BY salary DESC;
```

staffNo	fName	lName	salary
SL21	John	White	30000.00
SG5	Susan	Brand	24000.00
SG14	David	Ford	18000.00
SG37	Ann	Beech	12000.00
SA9	Mary	Howe	9000.00
SL41	Julie	Lee	9000.00

Example 6.12 Multiple Column Ordering

Produce abbreviated list of properties in order of property type.

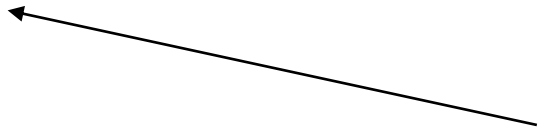
propertyNo	type	rooms	rent
PL94	Flat	4	400
PG4	Flat	3	350
PG36	Flat	3	375
PG16	Flat	4	450
PA14	House	6	650
PG21	House	5	600

Example 6.12 Multiple Column Ordering

Produce abbreviated list of properties in order of property type.

```
SELECT propertyNo, type,
       rooms, rent
FROM PropertyForRent
ORDER BY type;
```

propertyNo	type	rooms	rent
PL94	Flat	4	400
PG4	Flat	3	350
PG36	Flat	3	375
PG16	Flat	4	450
PA14	House	6	650
PG21	House	5	600



Default is to sort ascending ASC

Example 6.12 Multiple Column Ordering

Four flats in this list - as no minor sort key specified, system arranges these rows in any order it chooses.

propertyNo	type	rooms	rent
PL94	Flat	4	400
PG4	Flat	3	350
PG36	Flat	3	375
PG16	Flat	4	450
PA14	House	6	650
PG21	House	5	600

ordered by type only

order by type, then rent

propertyNo	type	rooms	rent
PG16	Flat	4	450
PL94	Flat	4	400
PG36	Flat	3	375
PG4	Flat	3	350
PA14	House	6	650
PG21	House	5	600

Example 6.12 Multiple Column Ordering

Four flats in this list - as no minor sort key specified, system arranges these rows in any order it chooses.

To arrange in order of rent, specify minor order:

Example 6.12 Multiple Column Ordering

Four flats in this list - as no minor sort key specified, system arranges these rows in any order it chooses.

To arrange in order of rent, specify minor order:

```
SELECT propertyNo, type, rooms, rent
FROM PropertyForRent
ORDER BY type, rent DESC;
```

Chapter 6 - Objectives

How to retrieve data from database using SELECT and:

- Use compound WHERE conditions.
- Use aggregate functions.
- Sort query results using ORDER BY.
- **Group data using GROUP BY** and HAVING.
- Use subqueries.
- Join tables together.
- Perform set operations (UNION, INTERSECT, EXCEPT).

SELECT Statement

```
SELECT [DISTINCT | ALL]
        {* | [columnExpression [AS newName]] [,...] }
FROM    TableName [alias] [, ...]
[WHERE  condition]
[GROUP BY  columnList]
[HAVING  condition]
[ORDER BY  columnList]
```

SELECT Statement

SELECT	Specifies which columns are to appear in output.
FROM	Specifies table(s) to be used.
WHERE	Filters rows.
GROUP BY	Forms groups of rows with same column value.
HAVING	Filters groups subject to some condition.
ORDER BY	Specifies the order of the output.

SELECT Statement - Grouping

Use GROUP BY clause to get sub-totals.

SELECT and GROUP BY closely integrated

Each item in SELECT list must be *single-valued per group*, and

SELECT clause may only contain:

- column names
- aggregate functions
- constants
- expression involving combinations of the above.

SELECT Statement - Grouping

All column names in SELECT list must appear in GROUP BY clause unless name is used only in an aggregate function.

```
SELECT type, AVG(rent)
FROM PropertyForRent
GROUP BY type;
```

rent in aggregate function

invalid syntax, propertyNo not in aggregate or GROUP BY

```
SELECT propertyNo, type,
    AVG(rent)
FROM PropertyForRent
GROUP BY type;
```

SELECT Statement - Grouping

All column names in SELECT list must appear in GROUP BY clause unless name is used only in an aggregate function.

If WHERE is used with GROUP BY, WHERE is applied first, then groups are formed from remaining rows satisfying predicate.

ISO considers two nulls to be equal for purposes of GROUP BY.

Example 6.17 Use of GROUP BY

Find number of staff in each branch and their total salaries.

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000.00	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000.00	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000.00	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000.00	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000.00	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000.00	B005

GROUP BY branchNo

Example 6.17 Use of GROUP BY

Find number of staff in each branch and their total salaries.

salary	branchNo
30000.00	B005
12000.00	B003
18000.00	B003
9000.00	B007
24000.00	B003
9000.00	B005

GROUP BY branchNo

Example 6.17 Use of GROUP BY

Find number of staff in each branch and their total salaries.

salary	branchNo
30000.00	B005
12000.00	B003
18000.00	B003
9000.00	B007
24000.00	B003
9000.00	B005

branchNo B003
COUNT of 3
SUM of 12000, 18000, 24000

GROUP BY branchNo

Example 6.17 Use of GROUP BY

Find number of staff in each branch and their total salaries.

salary	branchNo
30000.00	B005
12000.00	B003
18000.00	B003
9000.00	B007
24000.00	B003
9000.00	B005

branchNo B003
COUNT of 3
SUM of 12000, 18000, 24000

branchNo B005
COUNT of 2
SUM of 30000, 9000

GROUP BY branchNo

Example 6.17 Use of GROUP BY

Find number of staff in each branch and their total salaries.

salary	branchNo
30000.00	B005
12000.00	B003
18000.00	B003
9000.00	B007
24000.00	B003
9000.00	B005

branchNo B003
COUNT of 3
SUM of 12000, 18000, 24000

branchNo B005
COUNT of 2
SUM of 30000, 9000

branchNo B007
COUNT of 1
SUM of 9000

GROUP BY branchNo

Example 6.17 Use of GROUP BY

Find number of staff in each branch and their total salaries.

salary	branchNo
30000.00	B005
12000.00	B003
18000.00	B003
9000.00	B007
24000.00	B003
9000.00	B005

GROUP BY branchNo

branchNo	myCount	mySum
B003	3	54000.00
B005	2	39000.00
B007	1	9000.00

Example 6.17 Use of GROUP BY

Find number of staff in each branch and their total salaries.

branchNo	myCount	mySum
B003	3	54000.00
B005	2	39000.00
B007	1	9000.00

```
SELECT branchNo,
            COUNT(staffNo) AS myCount,
            SUM(salary) AS mySum
FROM Staff
GROUP BY branchNo
ORDER BY branchNo;
```

Chapter 6 - Objectives

How to retrieve data from database using SELECT and:

- Use compound WHERE conditions.
- Use aggregate functions.
- Sort query results using ORDER BY.
- **Group data using** GROUP BY and **HAVING.**
- Use subqueries.
- Join tables together.
- Perform set operations (UNION, INTERSECT, EXCEPT).

SELECT Statement

```
SELECT [DISTINCT | ALL]
        {* | [columnExpression [AS newName]] [,...] }
FROM    TableName [alias] [, ...]
[WHERE  condition]
[GROUP BY  columnList]
[HAVING  condition]
[ORDER BY  columnList]
```

SELECT Statement

SELECT	Specifies which columns are to appear in output.
FROM	Specifies table(s) to be used.
WHERE	Filters rows.
GROUP BY	Forms groups of rows with same column value.
HAVING	Filters groups subject to some condition.
ORDER BY	Specifies the order of the output.

Restricted Groupings – HAVING clause

HAVING clause is for used with GROUP BY to restrict groups that appear in final result table.

Similar to WHERE,

- **WHERE filters individual rows**
- **HAVING filters groups**
- **WHERE is applied first**

Column names in HAVING clause must also appear in the GROUP BY list or be contained within an aggregate function.

Example 6.18 Use of HAVING

For each branch with more than 1 member of staff, find number of staff in each branch and sum of their salaries.

branchNo	myCount	mySum
B003	3	54000.00
B005	2	39000.00
~~B007~~	~~1~~	~~9000.00~~

branchNo	myCount	mySum
B003	3	54000.00
B005	2	39000.00

Example 6.18 Use of HAVING

For each branch with more than 1 member of staff, find number of staff in each branch and sum of their salaries.

branchNo	myCount	mySum
B003	3	54000.00
B005	2	39000.00
B007	1	9000.00

```
SELECT branchNo,
       COUNT(staffNo) AS myCount,
       SUM(salary) AS mySum
FROM Staff
GROUP BY branchNo
HAVING COUNT(staffNo) > 1
ORDER BY branchNo;
```

branchNo	myCount	mySum
B003	3	54000.00
B005	2	39000.00