AT82.02

DATA MODELING AND MANAGEMENT

UNIT 1-3: SQL: INTRODUCTION

CHUTIPORN ANUTARIYA (CHUTI AT AIT DOT AC DOT TH)

Co-funded by the Erasmus+ Programme of the European Union

DS&AI

DDL: Data Definition Lang.

- CREATE
- ALTER
- DROP
- TRUNCATE
- COMMENT
- RENAME

DML: Data Manipulation Lang.

- SELECT
- INSERT
- UPDATE
- DELETE
- MERGE
- CALL
- EXPLAIN PLAN
- LOCK TABLE

DCL: Data Control Lang.

- GRANT
- REVOKE

TCL: Transaction Control Lang.

- COMMIT
- ROLLBACK
- SAVEPOINT
- SET TRANSACTION

The slides are based on the slides prepared by
José Machado, Paulo Novais
and Regina Sousa, University of Minho

UNDER THE DS&AI PROJECT

SQL

STRUCTURED QUERY LANGUAGE

In 1986, a standard for SQL was defined by the American National Standards Institute (ANSI), which was adopted in 1987 as an international standard by the International Organization for Standardization (ISO, 1987). This standard is called SQL and has proven to be the standard relational database language. Most Database Management Systems support this language, which makes it almost universal.

As a language, the ISO SQL standard has two main components:

- Data Definition Language (DDL) - to define the structure of the database and control data access;
- Data Manipulation Language (DML) - for data retrieval and updating.

SQL does not involve the specification of data access methods.

OBJECTIVES OF SQL

A database language should allow a user to:

- create the database and the relationship structures;
- perform simple and complex queries.
- perform basic data management tasks such as entering, modifying and deleting data from relationships;

A database language should perform these tasks with minimal effort from the user, and their structure and syntax should be easy to learn

SQL LANGUAGE

Data Manipulation Language
(DML)

SELECT

to query data in the database.

INSERT

to insert data into a table.

UPDATE

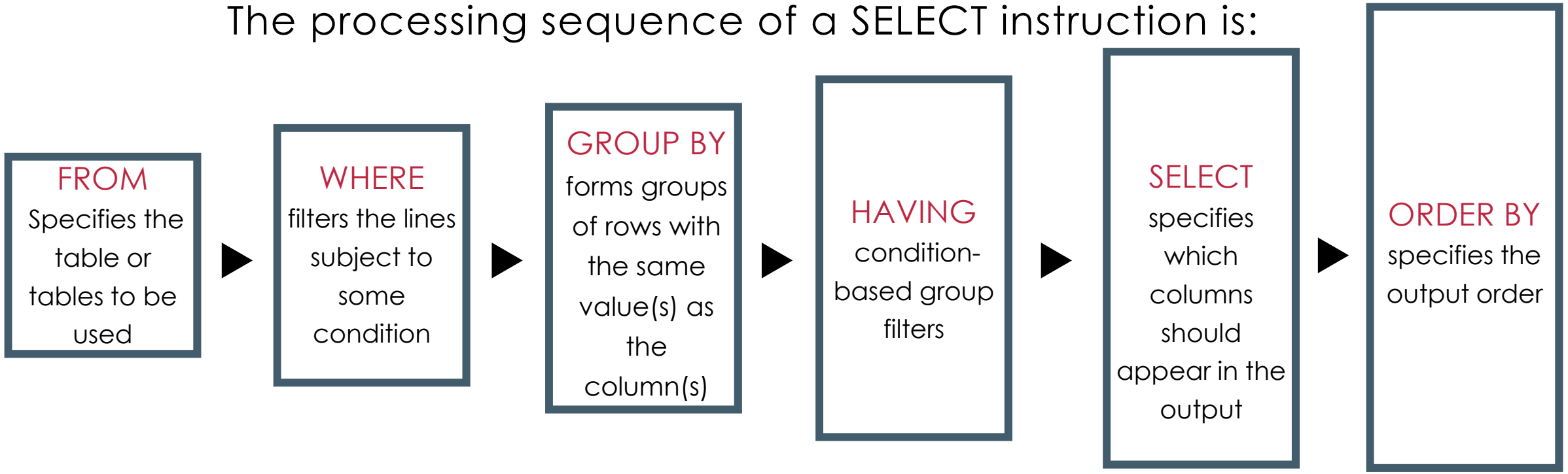
to update data from a table.

DELETE

to delete the data from a table.

SQL LANGUAGE: SELECT

The processing sequence of a SELECT instruction is:



The order of the clauses in the SELECT instruction cannot be changed.
Only the first two clauses are mandatory: SELECT and FROM; the rest are optional.

SQL LANGUAGE: DISTINCT

The SELECT DISTINCT statement is used to return only distinct (different) values.

Inside a table, a column often contains many duplicate values; and sometimes you only want to list the different (distinct) values.

```
SELECT DISTINCT column1, column2, ...
FROM table_name;
```

SQL LANGUAGE: WHERE

Match - Compares the value of one statement with the value of another statement.

Group - checks whether the value of a statement falls within a specified range of values.

It belongs - if the value of an expression belongs to a set of values.

Pattern match - if a string matches a specified pattern.

Null - test if a column has a null value (unknown).

SELECT column1, column2, ... FROM table_name WHERE condition;

SQL LANGUAGE: OPERATORS

- = equals
- <> different(ISO standard)
- ! = different(allowed in SGBD)
- < is less than
- <= is less than or equal to
- > is bigger than
- > = is bigger than or equal to

Rules for logical operators: AND, OR and NOT, with parentheses (if necessary or desired) to change normal precedence:

- evaluate an expression from left to right;
- in brackets are evaluated first;
- NOTs are evaluated first AND and OR operators;
- ANDs are evaluated before ORs.

SQL LANGUAGE: BETWEEN

The BETWEEN operator selects values within a given range. The values can be numbers, text, or dates.

The BETWEEN operator is inclusive: begin and end values are included.

```
SELECT column_name(s)
FROM table_name
WHERE column_name
BETWEEN value1 AND value2;
```

SQL LANGUAGE: IN

The IN operator allows you to specify multiple values in a WHERE clause.

The IN operator is a shorthand for multiple OR conditions.

```
SELECT column_name(s)
FROM table_name
WHERE column_name IN
(value1, value2, ...);
```

OR

```
SELECT column_name(s) FROM
table_name WHERE
column_name IN (SELECT
STATEMENT);
```

SQL LANGUAGE: LIKE

The LIKE operator is used in a WHERE clause to search for a specified pattern in a column.

SQL has two special pattern matching symbols:

- % represents any sequence of zero or more characters (wildcard).
- _ represents any unique character.

```
SELECT column1, column2, ...
FROM table_name
WHERE column LIKE pattern;
```

SQL LANGUAGE: NULL

It is not possible to test for NULL values with comparison operators, such as =, <, or <>.

We will have to use the IS NULL and IS NOT NULL operators instead.

```
SELECT column_names
FROM table_name
WHERE column_name IS NULL;
```

OR

```
SELECT column_names
FROM table_name
WHERE column_name IS NOT
NULL;
```

SQL LANGUAGE: ORDER BY

The ORDER BY clause allows rows to be displayed in ascending (ASC) or descending (DESC) order of any column or combination of columns, regardless of what appears in the result.

In some implementations ORDER BY only allows SELECT list elements. In both cases, ORDER BY should always be the last clause of the SELECT statement.

```
SELECT column1, column2, ...
FROM table_name
ORDER BY column1, column2, ... ASC|DESC;
```

SQL LANGUAGE: AGREGATION FUNCTION

The ISO standard defines five aggregate functions:

COUNT - returns the number of values in a specified column;

SUM - returns the sum of the values of a specified column;

AVG - returns the average of the values of a specified column;

MIN - returns the smallest value of a specified column;

MAX - returns the highest value of a specified column.

COUNT, MIN, MAX and apply to all column types, SUM and AVG can only be used in numeric columns. Each function deletes nulls first and operates only on the remaining non-zero values. COUNT (*) has a special use, which counts all rows of a table.

SQL LANGUAGE: GROUP BY

The columns referenced in GROUP BY are called grouping columns. When GROUP BY is used, each item in the selection list must be of unique value per group.

```
SELECT column_name(s)
FROM table_name
WHERE condition
GROUP BY column_name(s)
ORDER BY column_name(s);
```

SQL LANGUAGE: HAVING

The HAVING clause was added to SQL because the WHERE keyword could not be used with aggregate functions.

```
SELECT column_name(s)
FROM table_name
WHERE condition
GROUP BY column_name(s)
HAVING condition
ORDER BY column_name(s);
```

SQL LANGUAGE: MULTI TABLE QUERIES

A JOIN clause is used to combine rows from two or more tables, based on a related column between them.

```
SELECT Orders.OrderID, Customers.CustomerName,
Orders.OrderDate
FROM Orders
INNER JOIN Customers
ON Orders.CustomerID=Customers.CustomerID;
```

SQL LANGUAGE: EXISTS AND NOT EXISTS

They can be used in subqueries and produce a true or false result.

- EXISTS is true if, and only if, there is at least one row in the table returned by the subquery; it is false if the subquery returns empty.

- EXISTS is the opposite of EXISTING.

EXISTS and NOT EXISTS only verify the existence or not of rows in the result of the subquery.

```
SELECT column_name(s)
FROM table_name
WHERE EXISTS
(SELECT column_name FROM table_name WHERE condition);
```

SQL LANGUAGE: COMBINE TABLES

(UNION, INTERSECT, EXCEPT)

In SQL, we can use the operations of Union, Intersection and Exception to combine the results of two or more queries in a single result:

The Union of two tables, A and B, is a table with all the rows that are in the first table A or the second table B or both.

The Intersection of two tables, A and B, is a table with all rows that are common to both tables A and B.

The Exception of two tables, A and B, is a table with all lines that are in table A but not in table B.

SQL LANGUAGE: UNION, INTERSECT, EXCEPT

```
SELECT column_name(s) FROM table1
UNION
SELECT column_name(s) FROM table2;
```

```
(SELECT  column_name FROM table1)
INTERSECT
(SELECT column_name FROM table2);
```

```
(SELECT column_name FROM table1)
EXCEPT
(SELECT column_name FROM table2);
```

SQL LANGUAGE: DATABASE CHANGES

SQL is a complete data manipulation language and can be used to change database information in addition to query operations. There are three SQL commands available to modify the contents of the tables in the database:

INSERT adds new rows to a table;	UPDATE changes the data in a table;	DELETE removes rows from a table.

SQL LANGUAGE: INSERT

It is possible to write the INSERT INTO statement in two ways. The first way specifies both the column names and the values to be inserted. The second way only specifies the table and the values.

Make sure the order of the values is in the same order as the columns in the table

```
INSERT INTO table_name
(column1, column2,
column3, ...)
VALUES (value1, value2,
value3, ...);
```

OR

```
INSERT INTO table_name
VALUES (value1, value2,
value3, ...);
```

SQL LANGUAGE: UPDATE

The UPDATE statement allows the contents of existing rows in a table to be changed.

The new datavalue(s) must be compatible with the data type(s) of the corresponding column.

```
UPDATE table_name
SET column1 = value1, column2 = value2, ...
WHERE condition;
```

SQL LANGUAGE: DELETE

The DELETE statement allows rows to be deleted from a table.

Search condition is optional; if omitted, all table rows are deleted. This does not delete the table - to delete the contents and definition of a table we use the DROP TABLE command.

```
DELETE FROM table_name
WHERE condition;
```

Thank you.

Exit Slip:
Discuss 3 important things / concepts we have learned today.