# Relational Database Management Systems for Epidemiologists:

## SQL Part IV

# Outline

- Inserting, updating, and deleting rows
- Creating, altering, and dropping tables

# Adding Rows using INSERT

- Three ways to insert a row:
  - using column positions
  - using column names
  - using rows from another table

# INSERT using Column Positions

- Syntax:

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

- The number of values must equal number of columns in *table*.
- The values must be listed in the same sequence as the columns in *table*.

# INSERT using Column Names

- Syntax:

```
INSERT INTO table
     (column1, column2, column3,..., columnN)
     VALUES(value1, value2, value3,..., valueN);
```

- The number of values must equal number of columns in *table*.
- The values must be listed in the same sequence as the list of column names.

# INSERT using Another Table

- Syntax:

```
INSERT INTO table
    [(column1, column2, column3,..., columnN)]
    select_statement;
```

- The *select_statement* is any valid SELECT statement that returns rows of data to be inserted into *table*.
- The values must be listed in the same sequence as the list of columns in the resulting query from the SELECT statement.

# Updating Rows using UPDATE

- The UPDATE statement changes the values in a table's existing rows.
- One can either update all rows or specific rows in a table.
- To update rows, you specify:
  - the table to update,
  - the names of the columns to update and their new values, and
  - an optional search condition that specifies which rows to update.

# UPDATE Syntax

- Syntax:

  UPDATE *table*

  SET *column* = *expression*

  [WHERE *search_condition*];

- NOTE: the value returned by *expression* replaces the existing value in *column*.
- To change the values in multiple columns, type a list of comma-separated *column=expression* statements in the SET clause.

# Deleting Rows with DELETE

- The DELETE statement removes rows from a table.
- One can remove all rows or specific rows in a table.
- To delete rows, you specify:
  - the table whose rows to delete, and
  - an optional search condition that specifies which rows to delete.

# DELETE Syntax

- Syntax:

  DELETE FROM *table*

  [WHERE *search_condition*];

- NOTE: if the WHERE clause is omitted, every row in *table* is deleted (so be careful)!

# CREATE TABLE

- In order to create a table, you must specify:
  - table name
  - column name(s)
  - data type(s) for the column(s)
  - constraints (if any)
- Constraints define properties such as nulls, default values, keys, and permissible values.

# CREATE TEMPORARY TABLE

- One can create a temporary table that gets destroyed automatically at the end of a session or transaction.
- Temporary tables are useful for storing the result of complex queries that are used in subsequent queries or for creating a “snapshot” of the data.
- Temporary tables can either be local (available only to you) or global (available by you and other users).
- NOTE:  MS Access does NOT support temporary tables!

# CREATE TEMPORARY TABLE Syntax

- Syntax:

CREATE {LOCAL | GLOBAL} TEMPORARY
TABLE *table*
(
*column1 datatype1* [*constraints1*],
...
*columnN datatypeN* [*constraintsN*]
[, *table_constraints*]
);

# ALTER TABLE

- The use of ALTER TABLE is DBMS-specific, so check whether the following apply.
- On can:
  - add or drop a column
  - alter a column's data type
  - add, alter, or drop a column's nullability or default constraint
  - add, alter, or drop column or table constraints such as primary key, foreign key, unique, and check constraints
  - rename a column
  - rename a table.

# ALTER TABLE Syntax

- Syntax:

  ALTER TABLE *table*

  ADD [COLUMN] *column data_type*

  DROP COLUMN *column*

  ADD *constraint_definition*

  DROP CONSTRAINT *constraint_name*;

# DROP TABLE

- Use the DROP TABLE statement to remove a table from a database.
- Dropping a table destroys its structure, data, indexes, constraints, permissions, etc., ... that is, everything!
- NOTE:  you'll encounter problems with foreign keys or views that reference a dropped table unless they're altered or dropped as well.
- Syntax:

  DROP TABLE *tablename*;