Components of SQL

- data definition language (DDL)
  - creation and change of the data structures for the three levels of a database (external levels, conceptual level, physical level): definition of relation schemas, deletion of relations, creation of indexes, modification of relation schemas, creation of views
  - specification of integrity constraints
  - fixing of access rights (authorization)

- data manipulation language (DML)
  - insertion, change and deletion of data objects
  - interactive formulation of queries

- embedded DML
  - embedding of SQL-commands into an all-purpose programming language (host language) like e.g. Fortran, C, C++ or Java

- transaction control
  - commands for specifying the begin, abort or end of transactions, in some implementations explicit commands for locking data for concurrency control
5.2 Data definition language (DDL)

Data types

- primarily numbers, strings and date declarations as fundamental data types for attribute domains
- in detail:
  - `char(n)` character string of fixed length $n$, with user specified length $n$, synonym: `character(n)`
  - `varchar(n)` character string of variable length, with user specified maximum length $n$, synonym: `char varying(n)`, `character varying(n)`
  - `int` integer, value of a computer-dependent, finite subset of the whole numbers, synonym: `integer`
  - `smallint` small integer, a computer-dependent subset of the `int`-domain
  - `numeric(z, n)` fixed-point (decimal) number with user specified precision, $z =$ total number of digits, $n =$ number of the $z$ digits to the right of the decimal point, synonym: `decimal(z, n)`
  - `real` floating-point number with computer-dependent precision
- **double precision** | double-precision floating-point number with computer-dependent precision
- **float**\((n)\) | floating-point number with user specified precision of at least \(n\) digits
- **bit**\((n)\) | bit string of fixed length \(n\)
- **bit varying**\((n)\) | bit string of variable length with user specified maximum length \(n\)
- **blob** | **binary large object**, byte sequence of variable length up to 4 GB, for the representation of extremely large objects (e.g. multimedia objects, video sequences, geo-objects)
- **date** | calendar date with year (4 digits), month (2 digits), day (2 digits), format: YYYY-MM-DD
- **time** | time of day, in hours, minutes, and seconds, format: HH:MM:SS
- **time with time zone** | time difference to GMT (6 digits)
- **timestamp** | value containing date and time of date
- **interval** | relative value which can increment or decrement an absolute value of type **date**, **time** or **timestamp**, year/month- or day/hour-intervals
(restricted) declaration of a domain
- advantage: simple change of a data type for a domain which is used from several attributes in a schema
- form: `create domain < my type> as < type specification >`
  
  example: `create domain string as varchar(256)`

Specification of integrity constraints and default values

- Since SQL allows null values (`null`), an integrity constraint `not null` can be defined, if for a specific attribute a null value is `not` allowed.
- It is recommended to specify this condition for each primary key.
- definition of a default value for an attribute by attaching the clause `default < value >` to the attribute definition
- The default value is inserted into each new tuple, if an explicit value for this attribute is not specified. If a default clause is not defined, the default value is `null`.
- The clause `primary key` specifies one or more attributes that form the primary key of the relation.
- definition of a foreign key by the `foreign key` clause (referential integrity)