unique expresses that this attribute is a candidate key. If a candidate key is formed by several attributes $A_1, ..., A_n$, this is specified by the integrity constraint $\text{unique}(A_1, ..., A_n)$.

Creation of a relation schema

- in SQL no relations but tables (duplicates allowed)
- creation of a schema with the aid of the clause

```
create table R(A_1 D_1, A_2 D_2, ..., A_n D_n, 
[< integrity constraint_1 >, ..., < integrity constraint_k >])
```

$R$ relation name, $A_i$ name of an attribute in the schema of relation $R$, $D_i$ domain of $A_i$

- in BNF notation:

```
create table <relation name>(<relation comp> [, <relation comp>])* 
<relation comp> ::= <column definition> | <integrity constraint> 
<column definition> ::= <attribute name> <type> [<default value> | not null | unique] 
<default value> ::= [default <literal> | null]
```

The exact treatment of integrity constraints is discussed later.
integrity constraints

**primary key** \((A_{j_1}, ..., A_{j_m})\)

The attributes \(A_{j_1}, ..., A_{j_m}\) form the primary key of \(R\).

database schema: university schema (with incomplete integrity constraints)

```sql
create table students
    (reg-id int not null,
     name varchar(30) not null,
     sem int,
     primary key (reg-id))

create table professors
    (pers-id int not null,
     name varchar(30) not null,
     room int unique,
     rank char(2),
     primary key (pers-id))
```
create table assistants
  (pers-id int not null,
   name varchar(30) not null,
   room int unique,
   boss int,
   primary key (pers-id),
   foreign key (boss) references professors(pers-id))

create table lectures
  (id int not null,
   title varchar(30),
   credits int,
   held_by int,
   primary key (id),
   foreign key (held_by) references professors(pers-id))

create table attends
  (reg-id int not null,
   id int not null,
   primary key (reg-id, id),
   foreign key (reg-id) references students(reg-id),
   foreign key (id) references lectures(id))
create table is_precondition_of
  (predecessor   int not null,
successor      int not null,
primary key    (predecessor, successor),
foreign key    (predecessor) references lectures(id),
foreign key    (successor) references lectures(id))

create table tests
  (reg-id      int not null,
id           int not null,
pers-id      int not null,
grade        numeric(2,1),
primary key  (reg-id, id, pers-id),
foreign key  (reg-id) references students(reg-id),
foreign key  (id) references lectures(id),
foreign key  (pers-id) references professors(pers-id))
Change of a relation schema

- adding a new attribute (a new column) by the clause
  - `alter table <relation name> add <column definition>`
  - value `not null` is only allowed if a default value is specified

- deleting an attribute (a column) from a relation by the clause
  `alter table <relation name> drop <column definition>`

Deletion of a relation schema

- `drop table <relation name>`

- deletion of schema and relation instance

Deletion of a relation

- `delete from <relation name>`

- only the relation instance but not the schema is deleted
Creation of an index

- The goal of indexes is to improve query response time.
- An index relates to one or several attributes.
- A measure for the efficiency is in general the number of page accesses to the hard disc.

```sql
create [unique] index <index name> on <relation name>
(ATTRIBUTE_NAME [ORDER] [, ATTRIBUTE_NAME [ORDER]])* [cluster]
  - <ORDER> ::= Asc|Desc
  - unique: for all indexed attribute names two tuples with the same values forbidden
    ⇒ attribute fulfil key condition

- cluster: The tuples of the relation are actually inserted into the index structure and not only links to the tuples.
  ⇒ only one cluster index per relation

- example: create unique index room_index on professors (room)
```

Deletion of an index

```sql
drop index <index name>
```