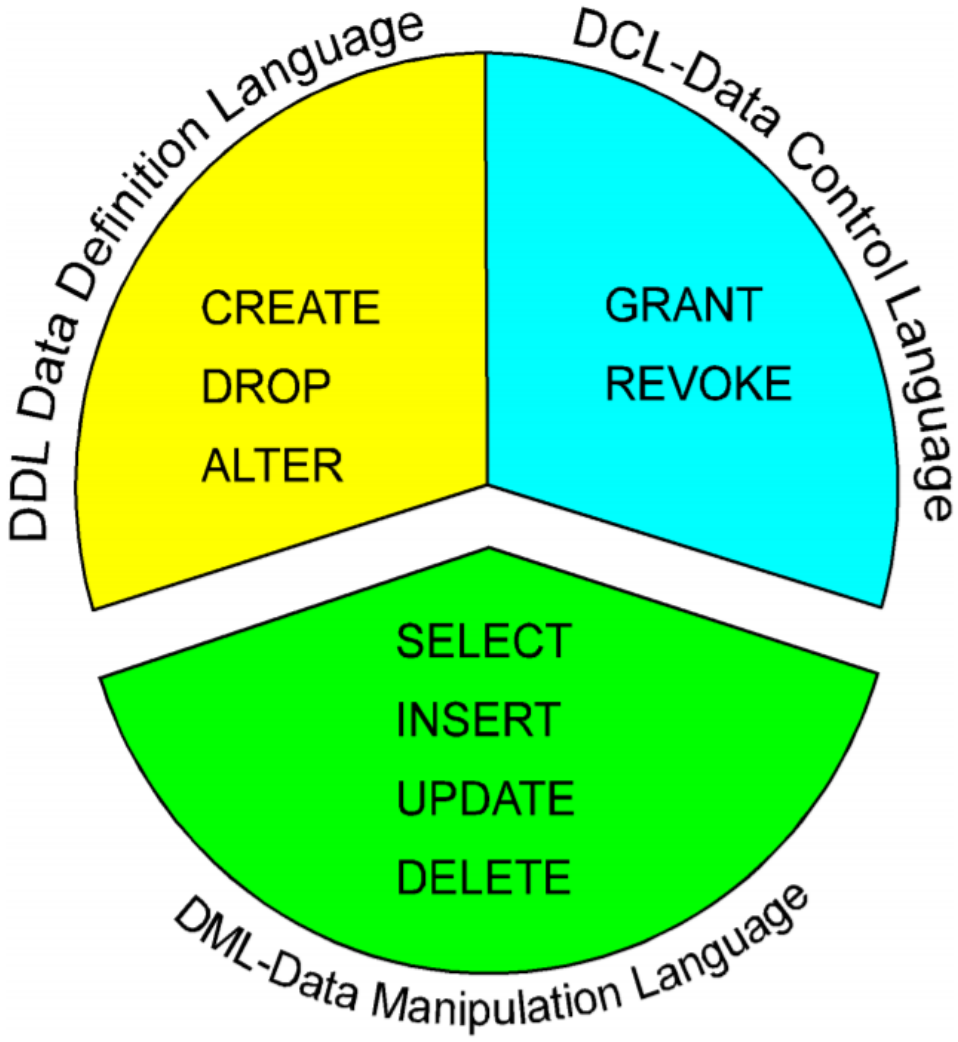
Дефиниране на схеми на релации. Ограничения.

SQL: CREATE, ALTER, DROP

SQL - Structured Query Language

- SQL е език за заявки към релационни БД. Посредством езика могат да се създават, изтриват и променят релационни схеми в СУБД.
- SQL се базира на заявки, групирани в следните под-езици:
  - Data Control Language (DCL) — контролира достъпа до обектите в базата от данни.
  - Data Definition Language (DDL) — използва се за създаване, модифициране или изтриване на обект в базата от данни.
  - Data Manipulation Language (DML) — използва се за извличане, обновяване, вмъкване или изтриване на данни.

SQL


DDL Data Definition Language
CREATE
DROP
ALTER
DCL-Data Control Language
GRANT
REVOKE
SELECT
INSERT
UPDATE
DELETE
DML-Data Manipulation Language

Дефиниране на релационна схема

Атрибутите имат типове

- CHAR(n) –низ с фиксирана дължина
- VARCHAR(n) –низ с дължина до n символа
- INT / INTEGER –цяло число, 32-bit, със знак
- FLOAT / REAL, DOUBLE
- DECIMAL(n, d) –дробно число с n цифри, d от тях след десетичната запетая (има разлика от FLOAT)
- DATE, TIME

Създаване на релация

MovieStar (name, address, gender, birthdate)

```
CREATE TABLE MovieStar
(name CHAR(30),
 address VARCHAR(255),
 gender CHAR(1),
 birthdate DATE DEFAULT CURRENT_DATE
);
```

Премахване (изтриване) на релация

- Изтриване на релация

```
DROP TABLE  MovieStar;
```

Обновяване на релация

- Добавяне на атрибут в релация

```
ALTER TABLE MovieStar
ADD phone CHAR(16);
```

- Изтриване на атрибут

```
ALTER TABLE MovieStar
DROP birthdate;
```

Ограничения

- NOT NULL ограничения
- Първичен ключ (Primary Key) - PK
- UNIQUE ограничение - UK
- Външен ключ (Foreign Key) - FK
- CHECK ограничения - CK

NOT NULL ограничение

- Не се допуска даден атрибут да приема NULL стойности;
- Декларира се САМО на ниво атрибут с ключовите думи NOT NULL

```
CREATE TABLE MovieStar
(name CHAR(30) not null,
 address VARCHAR(255)
);
```

- След като таблицата е създадена, се декларира като промяна на дефиницията на колоната

```
ALTER TABLE T
ALTER COLUMN col1 col1_type NOT NULL
```

Първичен ключ (PRIMARY KEY)

- Дефиниция: Едно множество от атрибути S на релацията R, наричаме първичен ключ, ако е изпълнено, че всеки два кортежа от R, се различават по стойността на поне един атрибут от множеството S

- Стойностите в една колона, декларирана като първичен ключ са уникални и не могат да бъдат NULL

- Една релация може да има само едни първичен ключ.

Деклариране на първичен ключ върху един атрибут

- Декларация на ниво атрибут

```
CREATE TABLE MovieStar(
  name CHAR(30)NOT NULL PRIMARY KEY,
  address VARCHAR(255),
  gender CHAR(1),
  birthdate DATE
);
```

- На ограничението се поставя служебно име.

Деклариране на първичен ключ върху един атрибут

- Декларация на ниво таблица

```
CREATE TABLE MovieStar(
  name CHAR(30) NOT NULL,
  address VARCHAR(255),
  gender CHAR(1),
  birthdate DATE ,
  PRIMARY KEY (name)
);
```

- На ограничението се поставя служебно име.

Деклариране на съставен първичен ключ

- Възможно е САМО на ниво таблица.

```
CREATE TABLE Movie(
  title varchar(50) NOT NULL,
   year  integer NOT NULL,
  length integer,
  inColor char(1),
  studioName varchar(50),
  PRIMARY KEY (title,year)
);
```

Деклариране на първичен ключ след създаване на таблицата

```
ALTER TABLE Movie
ADD [ CONSTRAINT Movie_pk ]
PRIMARY KEY (title,year)

ALTER TABLE MovieStar
ADD [ CONSTRAINT MovieStar_pk ]
PRIMARY KEY (name)
```

Сурогатен ключ

- Изкуствено създаден първичен ключ, когато нямаме подходяща колона с уникални стойности
- На пример автоматично номериране на кортежите
- Синтаксис в DB2

```
GENERATED ALWAYS AS IDENTITY
(START WITH 1, INCREMENT BY 1)
```

Сурогатен ключ - пример

```
CREATE TABLE MyTable(
   id int NOT NULL GENERATED
      ALWAYS AS IDENTITY
     (START WITH 1,INCREMENT BY 1)
  year  integer NOT NULL,
  length integer,
  PRIMARY KEY (id)
);
```

Ограничение за уникалност (UNIQUE)

- Ограничението UNIQUE също като PRIMARY KEY гарантира уникалност на стойностите в колоните, върху които е поставено.

- За разлика от PRIMARY KEY
  - Можем да имаме повече от един UNIQUE ограничение в една таблица;
  - Ограничението UNIQUE допуска NULL стойности в ключовите атрибути (включително и NULL във всички атрибути на ключа)

Деклариране на UNIQUE ограничение

- Аналогично на PRIMARY KEY

```
CREATE TABLE Movie(
  title varchar(50) not null,
  year  integer not null,
  length integer,
  inColor char(1),
  studioName varchar(50),
  producerC# integer,
  UNIQUE(title,year));
```

Външен ключ (FOREIGN KEY)

- Ограничение за референтна цялостност
- Гарантира, че стойностите на атрибутите от релацията, в която е дефиниран външния ключ, се срещат в съответните атрибути от релацията, която се реферира от външния ключ.
- Реферираните атрибути от втората релация трябва да бъдат декларирани като UNIQUE или PRIMARY KEY.
- Връзка (Parent – Child). FOREIGN KEY се декларира в таблицата Child към първичния ключ на Parent

Деклариране на външен ключ

- Декларация на ниво атрибут

```
REFERENCES  <parent_table>
(<parent_table_attribute>)
```

- На ниво таблица

```
FOREIGN KEY
(<child_table_attributes>)
REFERENCES <parent_table>
(<parent_table_attributes>)
```

Деклариране на външен ключ върху един атрибут

- Декларация на ниво атрибут

```
CREATE TABLE Studio (
  name CHAR(30) PRIMARY KEY,
  address VARCHAR(255),
  presC# INTEGER
   REFERENCES MovieExec(cert#)
);
```

- Тъй като полето presC# допуска NULL стойности, разрешените стойности за него са NULL или реално съществуващ (регистриран в MovieExec) режисьор.

Деклариране на съставен външен ключ

- САМО на ниво таблица

```
CREATE TABLE StarsIn(
  movietitle varchar(50),
  movieyear integer,
  name char(30),
  FOREIGN KEY (movietitle, movieyear)
  REFERENCES Movie(title, year)
);
```

Деклариране на външен ключ след създаване на таблицата

```
ALTER TABLE StarsIn
ADD [ CONSTRAINT MovieStar_FK ]
  FOREIGN KEY (starname)
  REFERENCES MovieStar(name);

ALTER TABLE StarsIn
ADD [ CONSTRAINT MovieStar_FK ]
  FOREIGN KEY (movietitle, movieyear)
  REFERENCES Movie(title, year);
```

Политики при външен ключ

- При създаване на външен ключ, могат да бъдат зададени три политики за налагане на ограничение по референтна цялостност. Тези политики определят поведението на СУБД при обновяване или изтриване на реферираният запис. Политиките са:
  - **RESTRICT / NO ACTION** - всяко действие, което нарушава референтна цялостност се отхвърля от СУБД. Например, ако се опитаме да изтрием стойност от реферираната колона, за която има реферициращи записи, то това действие ще бъде отхвърлено от СУБД и ще бъде изведено съобщение за грешка. Тази политика е и политиката по подразбиране.
  - **CASCADE** - при тази политика, ако се опитаме да изтрием или обновим стойност в реферираната колона, това ще доведе до автоматичното изтриване или обновяване на съответната стойност в реферициращите записи.
  - **SET NULL** - при тази политика, ако се опитаме да изтрием или обновим стойност в реферираната колона, това ще доведе до поставяне на стойност NULL в реферициращите записи, за променената или изтрита стойност. Тази политика е възможна само ако реферициращата колона позволява NULL стойности.

Добавяне на външен ключ с политика

```
CREATE TABLE StarsIn(
  movietitle varchar(50),
  movieyear integer,
  name char(30),
  FOREIGN KEY (movietitle, movieyear)
  REFERENCES Movie(title, year)
 ON DELETE CASCADE
 ON UPDATE RESTRICT;

);
```

Добавяне на CHECK ограничение

- Друг тип ограничение е CHECK. То проверява дали записите, които се вмъкват или обновяват отговарят на предварително зададено условие.
- Като условие, може да се използва всяко условие, което може да се постави и в WHERE клаузата на една SQL заявка, само че условията трябва да са прости и да съдържат логически изрази свързани с AND и/или OR, т.е. не могат да съдържат подзаявки.
- Условието в CHECK ограничението се проверява всеки път, когато вмъкваме запис в таблицата или обновяваме запис от таблицата, при което атрибутът за когото е дефиниран CHECK ограничението получава нова стойност (в резултат на INSERT или UPDATE).
- Ограничението CHECK, може да бъде създадено на ниво таблица и на ниво кортеж.

Добавяне на CHECK ограничение

- **Ограничение CHECK на ниво кортеж (при създаване на таблицата)**

```
CREATE TABLE my_table1
( col1 int NOT NULL,
 col2 char(2) NOT NULL check (col2 in ('BG', 'FR'))
);
```

- **Ограничение CHECK на ниво таблица (при създаване на таблицата)**

```
CREATE TABLE my_table1
( col1 int NOT NULL,
 col2 char(2) NOT NULL,
 check (col2 in ('BG', 'FR'))
);
```

Добавяне на CHECK ограничение (след като таблицата е вече създадена)

```
ALTER TABLE my_table1
ADD CONSTRAINT ck_col2
CHECK(col2 in ('BG', 'FR'));
```

- Обърнете внимание на това, че ако за col2 е създадено едно CHECK ограничение с едно условие, за същата колона може да бъде добавено и друго CHECK ограничение с друго условие.
- При тези случаи всички условия от дефинираните CHECK ограничения за една и съща колона се свързват логически с оператор AND. Например:

```
ALTER TABLE my_table1
ADD CONSTRAINT ck1_col2
CHECK(col2 in ('BG', 'FR'));

ALTER TABLE my_table1
ADD CONSTRAINT ck2_col2
CHECK(col2 in ('GR', 'RU'));
```

Ограничения на ниво таблица - Обобщение

Ограничение	Кратко име	Описание	Кога се проверява?
PRIMARY KEY	Първичен ключ	Ограничава стойностите да са уникални и да са различни от NULL.	При INSERT, при UPDATE на стойност от колоната, при DELETE на първичния ключ
UNIQUE	Ограничение за уникалност	Ограничава стойностите да са уникални	При INSERT, при UPDATE на стойност от колоната, при DELETE
FOREIGN KEY	Външен ключ	Ограничава стойностите да са от определено множество от стойности	При INSERT, при UPDATE на стойност от колоната, при DELETE на стойност от колоната на реферираната таблица
CHECK	Ограничение за проверка	Ограничава стойностите да отговарят на дадено условие	При INSERT, при UPDATE на стойност от колоната
NOT NULL	Ограничение за не NULL стойности		При INSERT, при UPDATE на стойност от колоната