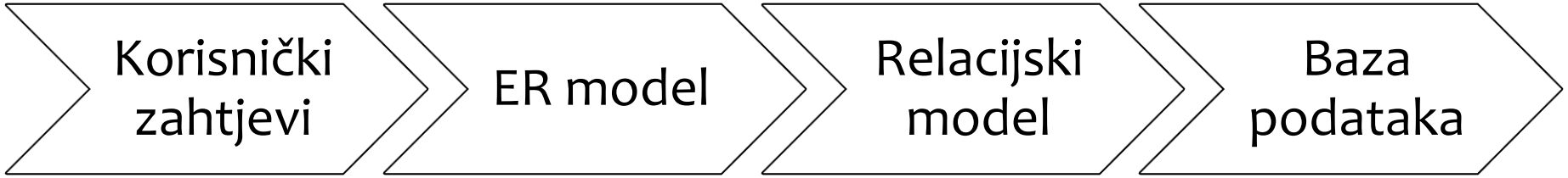
ALGEBRA BERNAYS SVEUČILIŠTE

UVOD U BAZE PODATAKA

Predavanje 3

Kratki uvod



- Klasifikacija odnosa među entitetima **prema broju instanci** koje sudjeluju u odnosu
  - 1:1, 1:N, N:1, M:N
- ER model se prikazuje **ER dijagramom** i **dijagramom entiteta**
- Složeniji odnosi:
  - Involuirani
  - Podskupovi

Sto je Involuirani odnos?

- Primjer?

Primjer ER dijagrama (1/4)

- Poslovno okruženje: **štand na placu**

- Korisnički zahtjevi:
  - Prodajem voće koje nabavljam od dobavljača
  - Od svakog voća prodajem više sorti (primjerice, kruške viljamovke, abate i kaiser, jabuke jonagold i idared, …)
  - Svako voće ima svoju nabavnu i prodajnu cijenu
  - U svakom trenutku me zanima koliko imam kojeg voća jer za svako voće imam količinu ispod koje radim novu narudžbu
  - Za svako voće imam nekoliko dobavljača

Primjer ER dijagrama (2/4)

- Je li sve jasno iz korisničkih zahtjeva?
  - Koji je odnos entiteta Voće i Sorta?

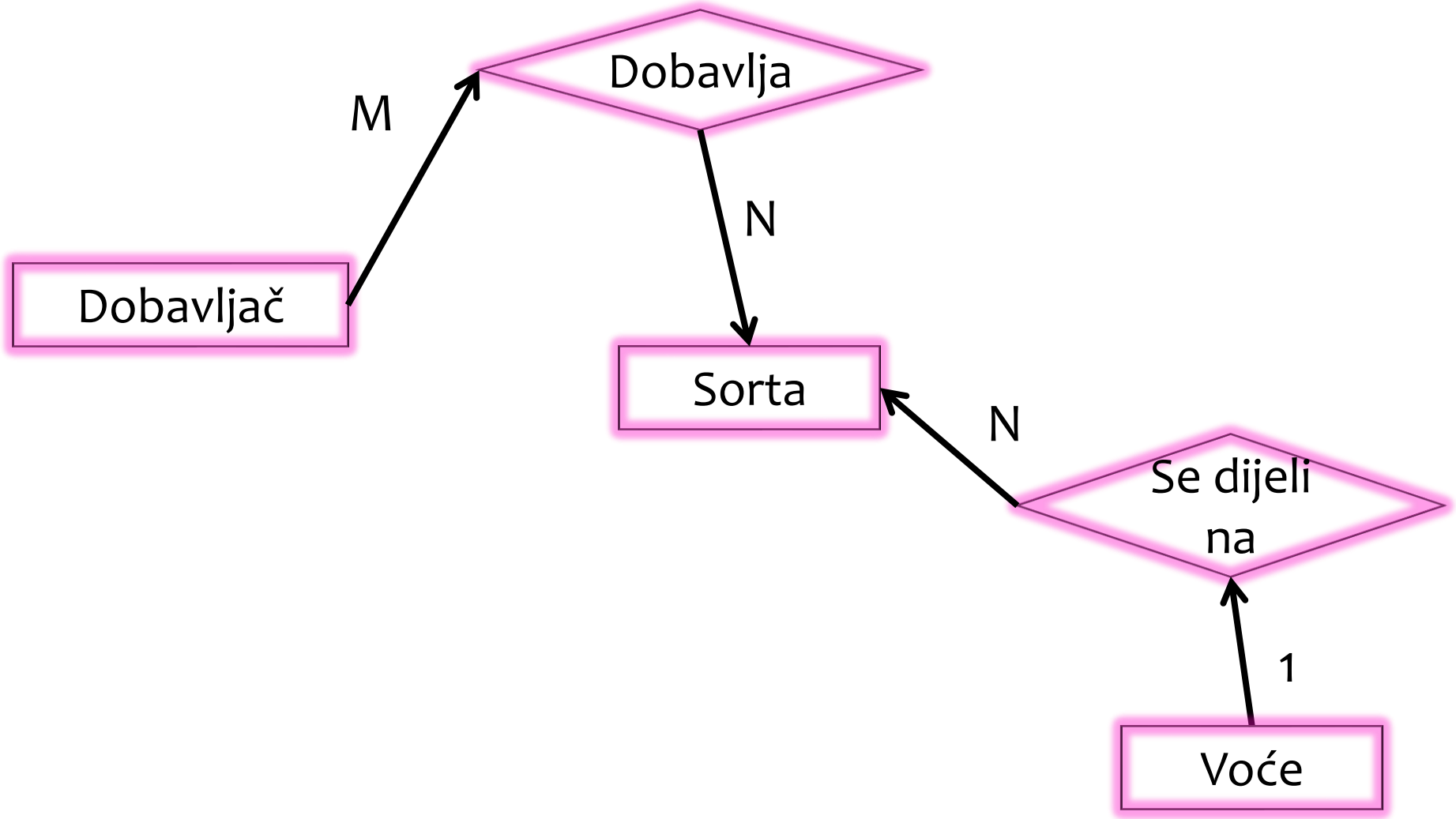
- Izmijenjeni korisnički zahtjevi:
  - Prodajem voće koje nabavljam od dobavljača
  - Od svakog voća prodajem više sorti (primjerice, kruške viljamovke, abate i kaiser, jabuke jonagold i idared, …)
  - Svako ~~voće~~ (sorta) ima svoju nabavnu i prodajnu cijenu
  - U svakom trenutku me zanima koliko imam kojeg ~~voća~~ (sorta) jer za svako ~~voće~~ (sorta) imam količinu ispod koje radim novu narudžbu
  - Za svako ~~voće~~ (sorta) imam nekoliko dobavljača

Primjer ER dijagrama (3/4)

- Entiteti i atributi:
  - **Voće**
    - IDVoće, Naziv
  - **Sorta**
    - IDSorta, Naziv, Minimalna količina, Trenutna količina, Nabavna cijena, Prodajna cijena
  - **Dobavljač**
    - IDDobavljač, Ime, Prezime, Adresa, Broj mobitela

- Odnosi:
  - Voće **se dijeli na** sorte
  - Dobavljač **dobavlja** sorte

Primjer ER dijagrama (4/4)



Relacijski model

Povijest relacijskog modela

- Teoretsku podlogu je 60-tih godina 20. stoljeća postavio **Edgar Frank Codd** (19 August 1923 – 18 April 2003)
- Početne implementacije su bile spore i neefikasne
  - Relacijski model samo implicitno (pomoću atributa) naznačava odnose između instanci entiteta
  - Problem: treba vremena za uspostavljanje veza pri upitu
  - Prednost: **veze definira korisnik**!
  - Žrtvovano malo brzine za fleksibilnost
- Porast računalne snage omogućio razvoj komercijalnih **RDBMS**-ova
- Od sredine 80-tih postaje prevladavajući model

Terminologija

- Za isti element relacijskog modela postoji više naziva:

Matematički naziv	Naziv u relacijskom modelu	Tradicionalni naziv
Relacija	Tablica	Datoteka
N-torka	Red ili redak	Zapis
Atribut	Stupac ili kolona	Polje
Vrijednost atributa	Pojedinačni podatak	Ćelija

Tablica

- Osnovni elementi relacijskog modela su **međusobno povezane tablice**
- Svaka tablica obavezno ima:
  - **Naziv** koji je uvijek u jednini
  - **Strukturu** koju čine stupci (naziva se i **relacijska shema**)
  - **Podatke** koje čine reci

Tablica Drzava

IDDrzava	Naziv
1	Hrvatska
2	Njemačka
3	Bosna i Hercegovina

Imenovanje

- Sve elemente relacijskog modela ćemo imenovati slično kao varijable u C++
  - Pokušat ćemo koristiti samo slova engleske abecede
  - Svaki naziv će se sastojati od samo jedne riječi
- Primjeri:
  - Atribut **Mjesto rođenja** iz ER modela ćemo nazvati **MjestoRodjenja**
  - Atribut **IDDržava** iz ER modela ćemo nazvati **IDDrzava**
  - Entitet **Prodavač** iz ER modela ćemo nazvati **Prodavac**

Pojedinačni podatak

- **Pojedinačni podatak** (engl. a*tomic data value*) je najmanji element relacijskog modela
- U tablici predstavljen **ćelijom**
- Ne može se rastaviti na dijelove bez gubljenja semantičkih svojstava
- Primjer pojedinačnog podatka:
  - Ime člana obitelji **Marko** – ima značenje
  - Sastavni elementi **M**, **a**, **r**, **k**, **o** – nemaju značenje

Stupac

- Stupci čine strukturu tablice
- Razlikujemo:
  - Stupci s jednostavnim vrijednostima:
    - Primjerice: **ImeRadnika**, **OmiljenaBoja**, **BrojCipela**
    - Vrijednosti u takvim stupcima su nedjeljive (pojedinačni podaci)
  - Stupci sa složenim vrijednostima:
    - Primjerice: **DatumRodjenja** i **AdresaPrebivalista**
    - Vrijednosti u takvim stupcima čine više pojedinačnih podataka
      - **DatumRodjenja** čini dan, mjesec i godina
- Složene stupce je moguće rastaviti na više jednostavnih – **iskustvo!**

Domena stupca

- Skup svih vrijednosti koje stupac može poprimiti
  - Kažemo da je stupac zadan na domeni

- Svaki stupac ima točno jednu domenu
- Primjerice:
  - Za stupac **BojaKose** domena je skup svih mogućih boja
  - Za stupac **Cijena** domena je skup svih realnih brojeva

- Više stupaca može imati istu domenu
- Primjerice:
  - **MjestoRodjenja** i **MjestoBoravka**
  - **BojaKose**, **BojaAuta** i **BojaFontaNaWebStranici**

Redak

- Stupci definiraju strukturu
- **Redak predstavlja podatak**
- Svaki redak sadržava po jednu vrijednost u svakom stupcu
  - Ako stupac nije obavezan, redak ne mora sadržavati vrijednost
- Svaki redak mora biti **jedinstven unutar tablice**
  - Mora se po nekim vrijednostima razlikovati od svih ostalih redaka u tablici (ključni atribut)
- Skup svih redaka u tablici čini **stanje tablice**
- Skup svih redaka u bazi podataka čini **stanje baze podataka**
  - Kažemo da baza može prijeći iz jednog stanja u drugo

Svojstva tablica

- Četiri osnovna svojstva tablica:
  1. Tablica ne može sadržavati dva stupca istog naziva
  2. Redoslijed stupaca u tablici nije bitan
  3. Tablica ne može sadržavati dva jednaka retka
  4. Redoslijed redaka u tablici nije bitan

- **Pogrešno** je pouzdavati se u redoslijed redaka u tablici:
  - U tablicu bilježimo slijetanje zrakoplova u zračnu luku, po jedan redak za svako slijetanje
  - Kako odgovoriti na pitanje: koji zrakoplov je prvi sletio nekog dana?
  - Ne smijemo se pouzdati u redoslijed umetnutih redaka

Ključevi

Uvod

- U relacijskom modelu koristimo **dvije vrste ključeva** za rješavanje dvaju problema:
  - Kako osigurati da je svaki redak jedinstven u tablici?
  - Kako povezati dvije tablice?

- Za osiguravanje jedinstvenosti u tablici koristimo **primarni ključ**

- Za povezivanje dviju tablica koristimo **strani ključ**

Primarni ključ

- Svaki redak u tablici mora biti jedinstven
- **Primarni ključ** (engl. *primary key*) čini jedan (ili više) stupaca čije vrijednosti su jedinstvene za svaki redak u tablici: ekvivalentno ključnom atributu u ER modelu
- Problem: može postojati više mogućih primarnih ključeva i svaki mogući primarni ključ se može sastojati od više atributa
- Rješenje: uvodimo surogatni primarni ključ
  - Nastaje preslikavanjem ključnog atributa iz ER modela

IDDrzava	Naziv
1	Hrvatska
2	Njemačka
3	Bosna i Hercegovina

Adresibilnost

- Adresibilnost je svojstvo relacijske baze podataka

- To znači da svaki pojedinačni podatak (ćeliju) u bazi podataka možemo **adresirati** (jednoznačno odrediti, pronaći) pomoću:

  - Naziva tablice

  - Vrijednosti primarnog ključa

  - Naziva stupca

Primjer adresibilnosti

- Primjer:
  - Neka strukturu tablice Grad čine stupci IDGrad, Naziv i DrzavaID
  - Neka je IDGrad primarni ključ
  - Ako tražimo pojedinačni podatak **Berlin**, moramo znati naziv tablice (Grad), vrijednost primarnog ključa (6) i naziv stupca (Naziv)

IDGrad	Naziv	DrzavaID
1	Zagreb	1
2	Osijek	1
3	Pula	1
4	Rijeka	1
5	Split	1
6	Berlin	2
7	Dresden	2

Strani ključ

- **Strani ključ** (engl. *foreign key*) se koristi za povezivanje tablica u semantičku strukturu

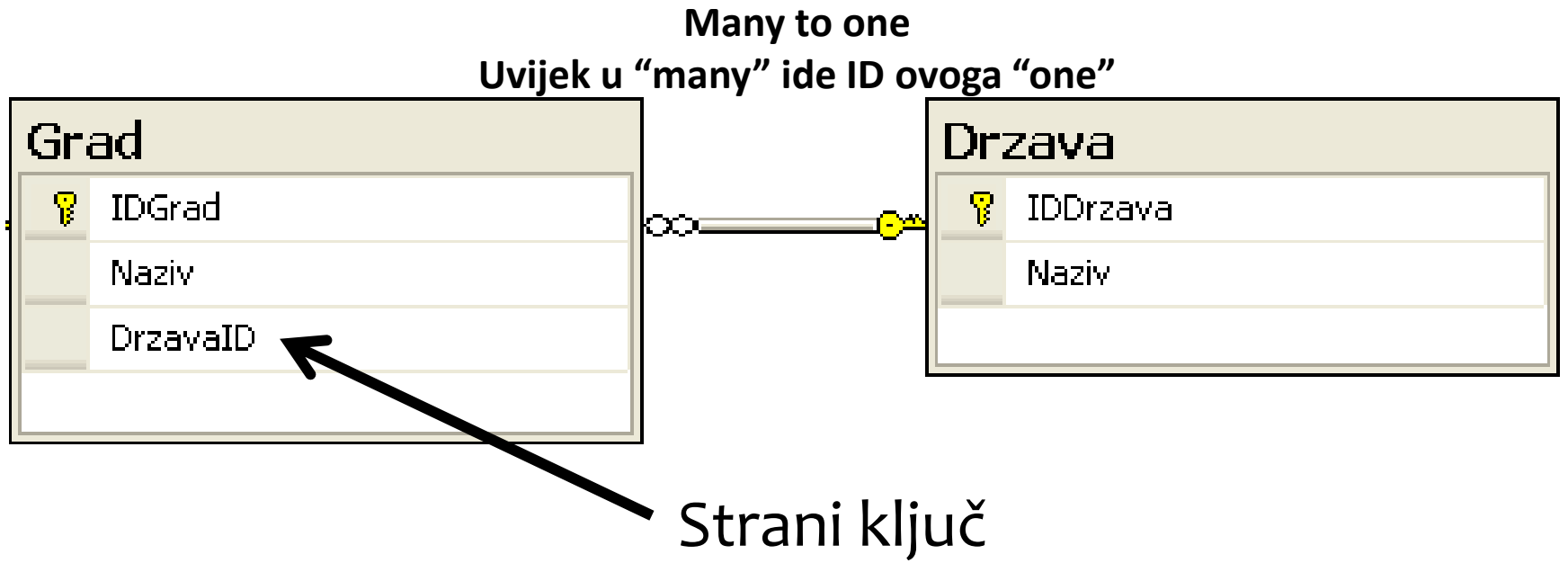
- Strani ključ čine jedan (ili više) stupaca koji referenciraju primarni ključ neke tablice
  - **Pozivajuća tablica** je ona koja sadrži strani ključ
  - **Ciljna tablica** je ona koja sadrži primarni ključ na koji strani ključ pokazuje

- Ako su svi primarni ključevi u bazi surogatni, onda su i svi strani ključevi surogatni

- Pozivajuća i ciljna tablica mogu biti iste:
  - Tablica **Zaposlenik** ima stupac **IDZaposlenik** (primarni ključ) i strani ključ **NadredjeniID** (strani ključ)

Primjer stranog ključa

- Tablica **Drzava** ima primarni ključ **IDDrzava**
- Tablica **Grad** ima primarni ključ **IDGrad**
- Tablica **Grad** ima strani ključ **DrzavaID** koji referencira primarni ključ **IDDrzava** iz tablice **Drzava**



Primjer stranog ključa

```sql
CREATE TABLE Persons (
    PersonID int,
    LastName varchar(255),
    FirstName varchar(255),
    Address varchar(255),
    City varchar(255)
);
```

```sql
CREATE TABLE Orders (
    OrderID int NOT NULL,
    OrderNumber int NOT NULL,
    PersonID int,
    PRIMARY KEY (OrderID),
    FOREIGN KEY (PersonID) REFERENCES Persons(PersonID)
);
```

```sql
CREATE TABLE Orders (
    OrderID int NOT NULL,
    OrderNumber int NOT NULL,
    PersonID int,
    PRIMARY KEY (OrderID),
    CONSTRAINT FK_PersonOrder FOREIGN KEY (PersonID)
    REFERENCES Persons(PersonID)
);
```

Primjer stranog ključa

```sql
CREATE TABLE Persons (
    PersonID int,
    LastName varchar(255),
    FirstName varchar(255),
    Address varchar(255),
    City varchar(255)
);
```

```sql
CREATE TABLE Orders (
    OrderID int NOT NULL,
    OrderNumber int NOT NULL,
    PersonID int,
    PRIMARY KEY (OrderID),
    FOREIGN KEY (PersonID) REFERENCES Persons(PersonID)
);
```

Zasto bih dodali “ime” FK?

```sql
CREATE TABLE Orders (
    OrderID int NOT NULL,
    OrderNumber int NOT NULL,
    PersonID int,
    PRIMARY KEY (OrderID),
    CONSTRAINT FK_PersonOrder FOREIGN KEY (PersonID)
    REFERENCES Persons(PersonID)
);
```

Primjer stranog ključa

```sql
CREATE TABLE Persons (
    PersonID int,
    LastName varchar(255),
    FirstName varchar(255),
    Address varchar(255),
    City varchar(255)
);
```

```sql
CREATE TABLE Orders (
    OrderID int NOT NULL,
    OrderNumber int NOT NULL,
    PersonID int,
    PRIMARY KEY (OrderID),
    FOREIGN KEY (PersonID) REFERENCES Persons(PersonID)
);
```

```sql
CREATE TABLE Orders (
    OrderID int NOT NULL,
    OrderNumber int NOT NULL,
    PersonID int,
    PRIMARY KEY (OrderID),
    CONSTRAINT FK_PersonOrder FOREIGN KEY (PersonID)
    REFERENCES Persons(PersonID)
);
```

Zasto bih dodali “ime” FK?

```sql
ALTER TABLE Orders
DROP FOREIGN KEY FK_PersonOrder;
```

Ograničenja vrijednosti

- **Ograničenja vrijednosti stupaca** u tablici koristimo da bismo spriječili unos neispravnih podataka u bazu podataka
- Odraz zakonitosti iz realnog svijeta
- Dijelimo ih u dvije grupe:

  - Ograničenja neovisna o vrijednostima drugih stupaca:
    - Vrijednost stupca **GodinaRodjenja** ne može biti **2153**
    - Vrijednost stupca **BrojCipela** ne može biti **150**

  - Ograničenja ovisna o vrijednostima drugih stupaca (zavisnosti)
    - Ako je vrijednost stupca **GodinaRodjenja** jednaka **1992**, onda vrijednost stupca **GodinaRadnogStaza** ne može biti **30**
    - Ako je vrijednost stupca **GodinaRodjenja** jednaka **1951**, onda može!

“Defaultne” vrijednosti

Automatic Value Insertion

Pojednostavljuje kreiranje I koristenje baze

Optional in SQL: The DEFAULT constraint is optional. Not all columns require a default value.

Moze biti aplicirano na bilo koji data type

NULL vrijednosti su NULL, nemaju default

“Defaultne” vrijednosti

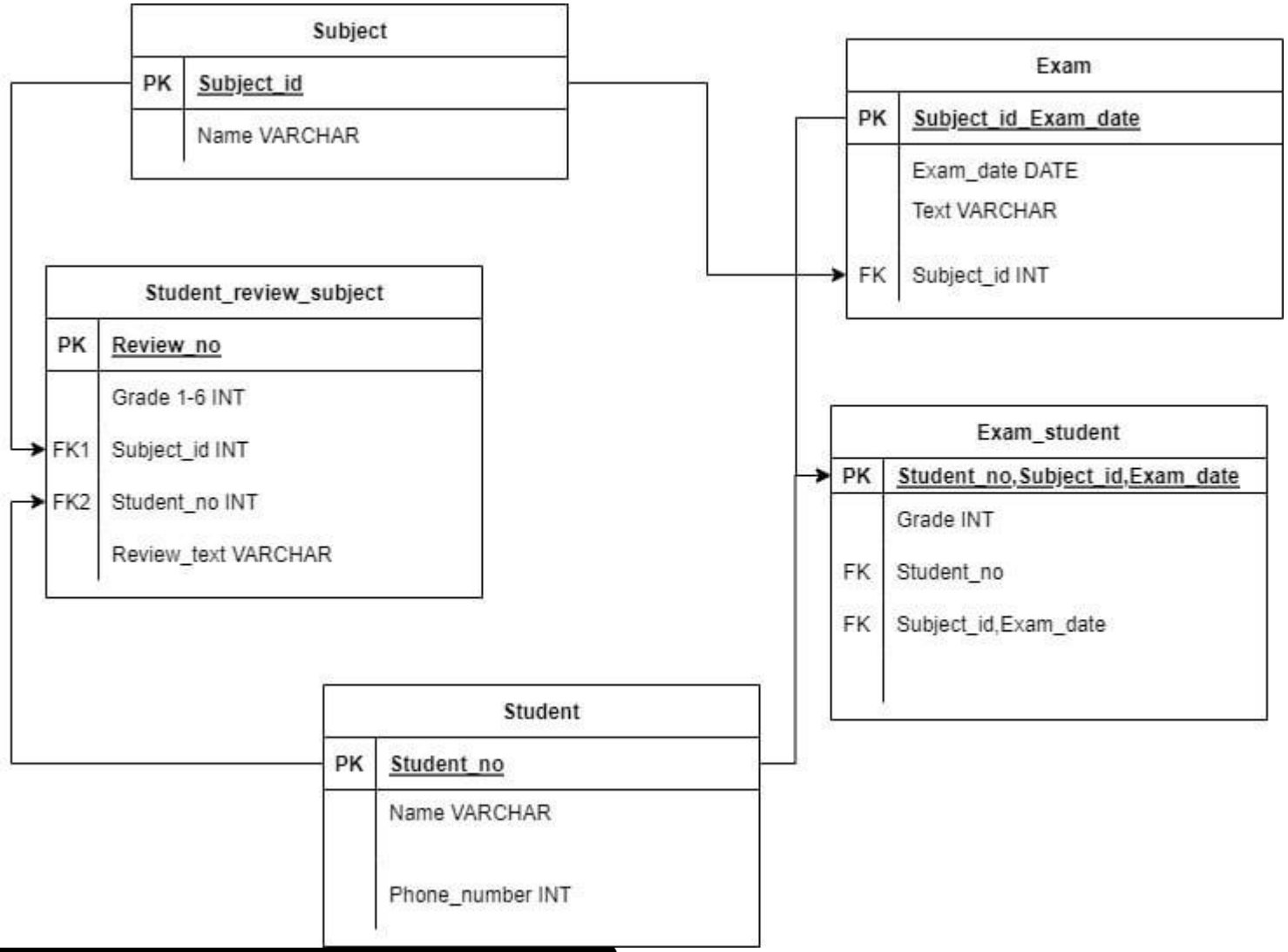
```
CREATE TABLE Geeks (
    ID INT NOT NULL,
    Name VARCHAR(255),
    Age INT,
    Location VARCHAR(255) DEFAULT 'Noida'
);

-- Explicit value
INSERT INTO Geeks (ID, Name, Age, Location) VALUES (4, 'Mira', 23, 'Delhi');

-- Using the DEFAULT constraint
INSERT INTO Geeks (ID, Name, Age, Location) VALUES (5, 'Hema', 27);
```

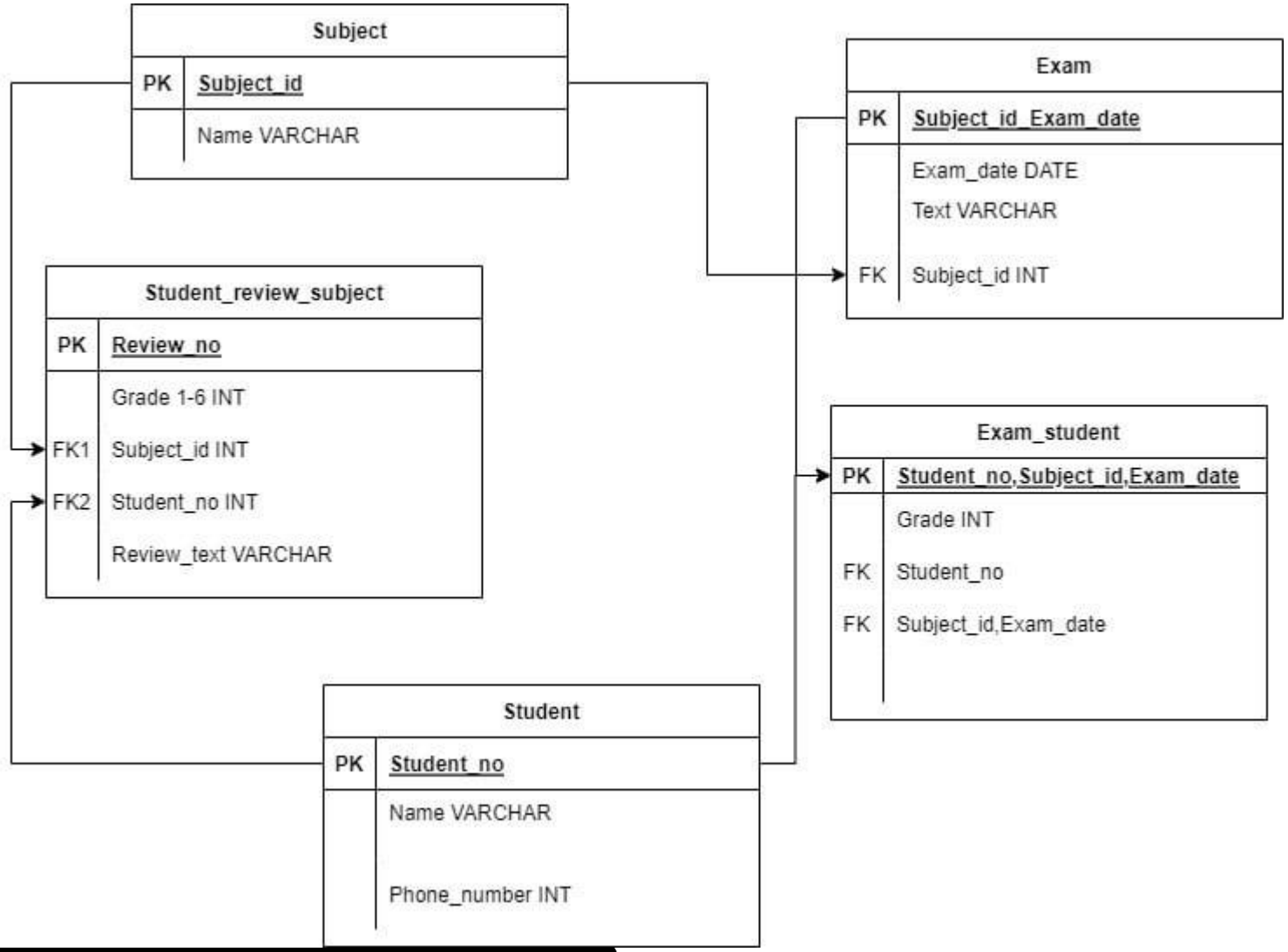
Shema diagram

- Detaljno objasnjava bazu podataka
- Sadrzi tipove podataka
- Sadrzi FK I PK
  - Tj relacije
- Sadrzi sve potrebne attribute



Shema diagram

- the “blueprint” of a database
- Zadnji korak prije kreiranje baze kroz MySQL tj RDBMS



LO2:

- Izraditi bazu podataka upotrebom naredbi DDL na temelju relacijskog modela.