SQL Components:
More details for your Own Study + Exercise 1

- DDL = Data Definition Language
  - CREATE TABLE, DROP TABLE, ALTER TABLE
  - CREATE DATABASE, …
- DML = Data Manipulation Language
  - INSERT, DELETE, UPDATE, SQL

Creating (Declaring) a Relation

- Simplest form is:

```
CREATE TABLE <name> (
  <list of elements>
);
```

- To delete a relation:

```
DROP TABLE <name>;
DROP TABLE IF EXISTS <name>;
```

Example: Create Table

```
CREATE TABLE Coffees
(
  name VARCHAR(20),
  manufacturer VARCHAR(20)
);

CREATE TABLE Sells (
  coffeehouse CHAR(20),
  coffee VARCHAR(20),
  price REAL
);
```

Kinds of Constraints

- **Keys**
- **Foreign-key**, or referential-integrity
- **Value-based constraints**
  - Constrain values of a particular attribute
- **Tuple-based constraints**
  - Relationship among components
- **Assertions**
  - any SQL boolean expression

Declaring Single-Attribute Keys

- Place PRIMARY KEY or UNIQUE after the type in the declaration of the attribute
- Equivalent examples:

```
CREATE TABLE Coffees (
        name VARCHAR(20) PRIMARY KEY,
        manufacturer VARCHAR(20)
);

CREATE TABLE Coffees (
        name VARCHAR(20),
        manufacturer VARCHAR(20),
        PRIMARY KEY (name)
);
```

Declaring Multi-Attribute Keys

- A key declaration can also be another element in the list of elements of a CREATE TABLE statement

- This form is essential if the key consists of more than one attribute
  - May be used even for one-attribute keys

Example: Multi-Attribute Key

- The coffeehouse and coffee together are the key for *Sells*:

```
CREATE TABLE Sells (
  coffeehouse    VARCHAR(20),
  coffee         VARCHAR(20),
  price          REAL,
  PRIMARY KEY (coffeehouse, coffee)
);
```

Declaring Keys
PRIMARY KEY vs. UNIQUE

- An attribute or list of attributes may be declared PRIMARY KEY or UNIQUE
- Both say that no two tuples of the relation may agree in all the attribute(s) in the list
- There can be only one PRIMARY KEY for a relation, but several UNIQUE attributes
  - No attribute of a PRIMARY KEY can ever be NULL in any tuple.
  - Attributes declared UNIQUE may have NULL values, and there may be several tuples with NULL values

Foreign Keys

- Values appearing in attributes of one relation must be key of another relation

- **Example**:
  in *Sells*(coffeehouse, coffee, price), we might expect that a coffee value also appears in *Coffees*.name

Expressing Foreign Keys

- Use keyword REFERENCES, either:
  1. After an attribute (for one-attribute keys)
  2. As an element of the schema:

```
FOREIGN KEY (<list of attributes>)
     REFERENCES <relation>
   (<attributes>)
```

- Referenced attributes must be declared PRIMARY KEY or UNIQUE

Example: With Attribute

```
CREATE TABLE Coffees(
  name VARCHAR(20) PRIMARY KEY,
  manufacturer VARCHAR(20)
);

CREATE TABLE Sells (
  coffeehouse VARCHAR(20)
              REFERENCES Coffeehouses(name),
  coffee VARCHAR(20) REFERENCES
  Coffees(name),
  price REAL
);
```

Example: As Schema Element

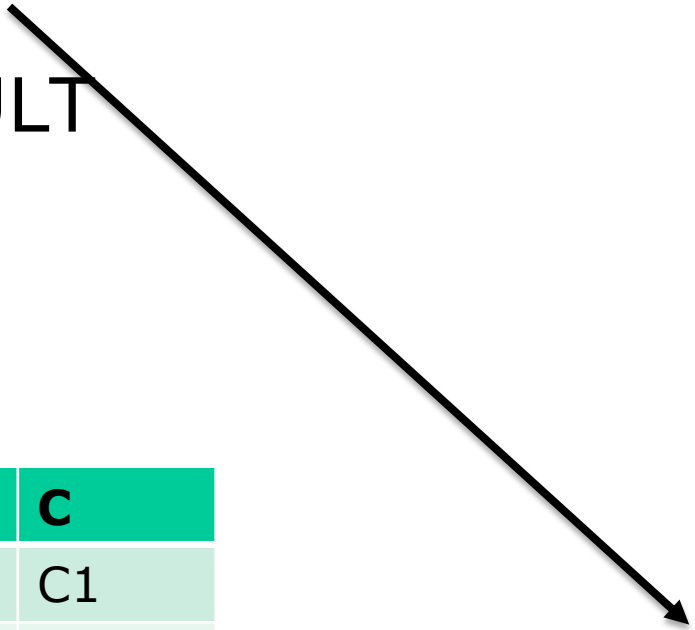
```
CREATE TABLE Coffees(
  name VARCHAR(20) PRIMARY KEY,
  manufacturer VARCHAR(20)
);

CREATE TABLE Sells (
  coffeehouse VARCHAR(20),
  coffee VARCHAR(20),
  price REAL,
  FOREIGN KEY(coffee) REFERENCES
  Coffees(name)
  FOREIGN KEY(coffeehouse) REFERENCES
                  Coffeehouses(name)
);
```

Equivalent
to previous slide!

FK: What Happens on Delete?

- NO ACTION (= error!) ← Default and usually correct!
- CASCADE
- SET DEFAULT
- SET NULL

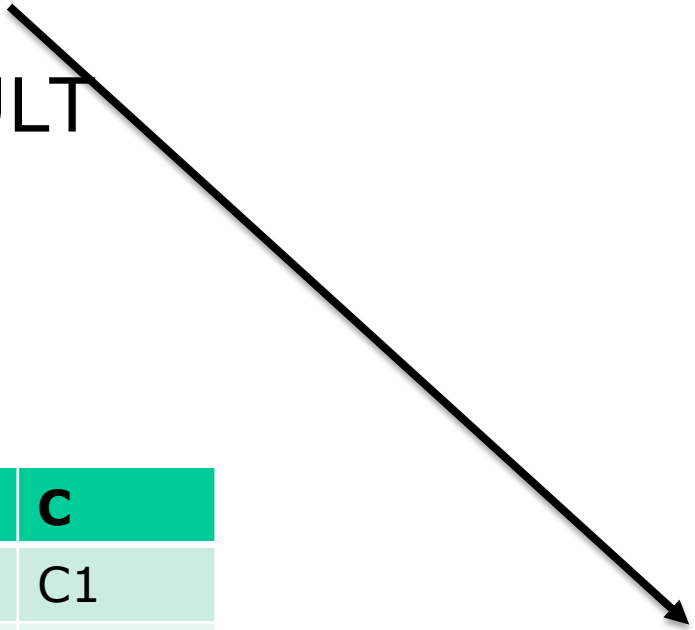


A	B	C
A1	B1	C1
A1	B2	C1
A2	B1	C2
A3	B1	C3
A3	B3	C1

C	D
~~C1~~	~~D1~~
C2	D2
C3	D3

FK: What Happens on Update?

- NO ACTION (= error!) ← Default and usually correct!
- CASCADE
- SET DEFAULT
- SET NULL



A	B	C
A1	B1	C1
A1	B2	C1
A2	B1	C2
A3	B1	C3
A3	B3	C1

C	D
C4	D1
C2	D2
C3	D3

More Complex Forms

- https://www.postgresql.org/docs/current/sql-createtable.html

```
CREATE TABLE IF NOT EXISTS People (
  ID integer PRIMARY KEY,
  kennitala CHAR(10) NOT NULL UNIQUE,
  name varchar(40) NOT NULL CHECK (name <> ''),
  numchildren integer DEFAULT 0,
  zipcode integer REFERENCES ZipCodes,
  age integer CHECK (age > 0),
  salary integer NOT NULL,
  CONSTRAINT salarycheck CHECK (salary > age)
);
```

Changing a Relation

- Possible to change relations, e.g.:
  - Add / Drop column
  - Add / Drop constraint
  - Add / Drop default values
- Called "Schema Maintenance"

Inserting Data

- INSERT
  INTO Relation (Column, …)
  VALUES (Value, ...)

```
insert
into Coffees (name, manufacturer)
values ('Slop', 'Braga');
```

Updating Data

- UPDATE table
  SET column = value (expression)
  WHERE condition

Examples!

- Change Johan's rating of Blue Mountain to 10

```
update Likes
set rating = 10
where drinker = 'Johan'
  and coffee = 'Blue Mountain';
```

- Raise all prices by 10%

```
update Sells
set price = price * 1.1;
```

Coffees(name, manufacturer)
Coffeehouses(name, address, license)
Drinkers(name, address, phone)
Likes(drinker, coffee, rating)
Frequents(drinker, coffeehouse)
Sells(coffeehouse, coffee, price)

Deleting Data

- DELETE
  FROM table
  WHERE condition

```
delete
from Sells
where coffeehouse = ‘Mocha';
```

Examples!

```
delete
from Sells
where coffeehouse = 'Mocha';

delete
from Frequents
where coffeehouse = 'Mocha';

delete
from Coffeehouses
where name = 'Mocha';
```

- What happens if I try to delete the coffeehouse Mocha?
- Step 1:
  - Mocha no longer sells anything
  - Nobody frequents Mocha
- Step 2: Mocha no longer exists

Coffees(name, manufacturer)
Coffeehouses(name, address, license)
Drinkers(name, address, phone)
Likes(drinker, coffee, rating)
Frequents(drinker, coffeehouse)
Sells(coffeehouse, coffee, price)