# CS 235: Introduction to Databases

Svetlozar Nestorov

*Lecture Notes #12*

## Defining a Database Schema

- CREATE TABLE name (list of elements).
- Principal elements are attributes and their types, but key declarations and constraints also appear.
- Similar CREATE *X* commands for other schema elements *X*: views, indexes, assertions, triggers.
- DROP *X* name
  - deletes the created element of kind *X* with that name.

## Example

- DROP TABLE Bars;
- CREATE TABLE Bars (

```
      bar CHAR(20),
      beer VARCHAR(20)
      price REAL
);
```

## Types

1. INT or INTEGER
2. REAL or FLOAT.
3. CHAR(*n*) = fixed length character string, padded with “pad characters.”
4. VARCHAR(*n*) = variable-length strings up to *n* characters.
5. Also, TEXT, BLOB
6. ENUM(‘val1’, ‘val2’, …)

## More Types

- Dates. SQL form is DATE 'yyyy-mm-dd'
- Times. Form is TIME 'hh:mm:ss[.ss…]‘
- DATETIME
- TIMESTAMP
  - In MySQL the first TIMESTAMP column in a table is automatically set to the date and time of the most recent operation except when the operation explicitly changed it.

## Declaring Keys

- Use PRIMARY KEY or UNIQUE.
- But only one primary key, many *uniques* allowed.
- SQL does not allow nulls in primary key, but allows them in *unique* columns (which may have two or more nulls, but not repeated non-null values).

## Declaring Keys

Two places to declare:

1. After an attribute's type, if the attribute is a key by itself.
2. As a separate element.
   – Essential if key is >1 attribute.

## Example

```
CREATE TABLE Sells (
        bar CHAR(20),
        beer VARCHAR(20),
        price REAL,
        invoice NUMBER UNIQUE,
        PRIMARY KEY(bar,beer)
);
```

## Other Properties You Can Give to Attributes

- NOT NULL = every tuple must have a value for this attribute.
- DEFAULT value = a value to use whenever no other value of this attribute is known.

```
CREATE TABLE Drinkers (
name CHAR(30) PRIMARY KEY,
addr CHAR(50)
    DEFAULT ‘1100 E 58th str’,
 phone CHAR(16)
);
```

## Example

- INSERT INTO Drinkers(name) VALUES('Sally') results in the following tuple:
  (Sally, ’1100 E 58th str.’, NULL)
- Primary key is by default not NULL.
- This insert is legal.
  – OK to list a subset of the attributes and values for only this subset.
- But if we had declared

  ```
  phone CHAR(16) NOT NULL
  ```

  then the insertion could not be made.

## Changing Columns

- Add an attribute of relation *R* with
  ALTER TABLE *R* ADD <column declaration>;
- ALTER TABLE Bars ADD phone CHAR(16)
  DEFAULT 'unlisted';
- Columns may also be dropped.
- ALTER TABLE Bars DROP license;

## Avoiding Schema Modifications

- Only if absolutely necessary
  – Never modify the schema of “live” database.
- A better approach may be to recreate database (and update create scripts).
- Schema modifications may result in:
  – Disk fragmentation
  – Rebuilding index
  – Re-optimizing queries
  – Run time errors for SQL queries

# Views

- An expression that describes a table without creating it.
- View definition form is:

```
CREATE VIEW <name> AS  <query>;
```

- Create a view CanDrink for the set of drinker-beer pairs such that the drinker frequents at least one bar that serves the beer.

# Querying Views

- Treat the view as if it were a materialized relation.
- What beers can Sally drink?

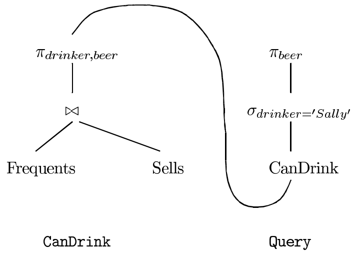
```
SELECT beer
FROM CanDrink
WHERE drinker = 'Sally';
```

# Semantics of View Use

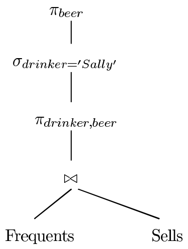
- Combine the view definition and the query and derive a new SQL expression.

1. SQL query over view to relational algebra
2. SQL view def. to relational algebra.
3. Combine and optimize.
4. Translate back to SQL.

# Example

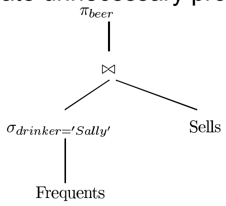


# Compose



# Optimize Query

1. Push selections down tree.
2. Eliminate unnecessary projections.


$\pi_{beer}$
$\bowtie$
$\sigma_{drinker='Sally'}$
Sells
Frequents