Database Basics

A basic introduction to relational databases

James Brucker

Relational or NoSQL

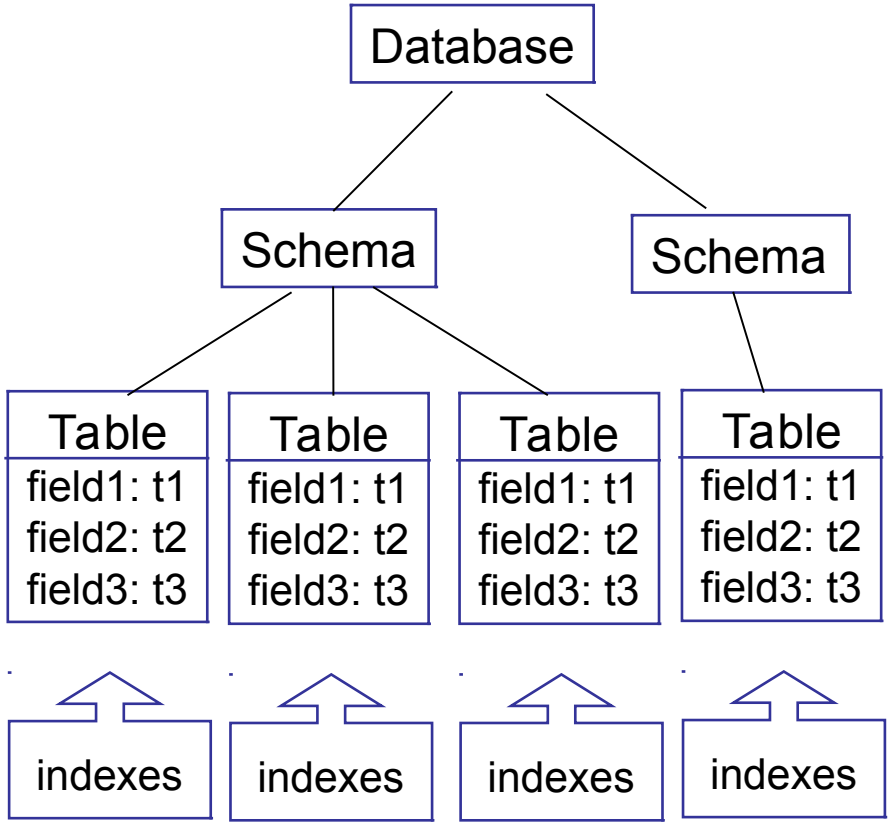
Relational Databases - data stored in structured format, with data in fields (columns), one record per row, in tables. Tables are similar to tables in a spreadsheet, but columns have fixed size.
Data in different tables can be related based on common values or expressions.

Examples: Sqlite, MySQL, Oracle, H2

NoSQL Databases - any database not organized like a Relational Database. Some forms are document-oriented and graph databases.

Examples: MongoDB, CouchDB (document),
Neo4j (graph)

Database Structure



A database contains **schema**, which describe its structure.

A schema can contain:

tables - containing data

index files - for fast lookup of data in tables

stored procedures, constraints, triggers, and more

SQLite databases have only one schema, so its not shown.

A Table

- A table contains the actual data in records (rows).
- A record is composed of fields (columns).
- Each record contains one set of data values.

records

(rows)

ID	Name	CCode	District	Populatn
3320	Bangkok	THA	Bangkok	6320174
3321	Nonthaburi	THA	Nonthaburi	292100
3323	Chiang Mai	THA	Chiang Mai	171100

fields (columns)

Key field to Identify Rows

- A table contains a *primary key* that uniquely identifies a row of data.
- Each record must have a distinct value of primary key
- The primary key is used to relate (join) tables.

ID is the primary key in City table.

ID	Name	CCode	District	Populatn
3320	Bangkok	THA	Bangkok	6320174
3321	Nonthaburi	THA	Nonthaburi	292100
3323	Chiang Mai	THA	Chiang Mai	171100

Structure of a Table

Every field has:

- a name
- a data type and length

To view the structure of a table use:

`DESCRIBE` *`tablename`*

```
sql> DESCRIBE City;
+-------------+-----------+-----+-----+---------+----------------+
| Field       | Type      | Null| Key | Default | Extra          |
+-------------+-----------+-----+-----+---------+----------------+
| ID          | int(11)   | NO  | PRI |         | auto_increment |
| Name        | char(35)  | NO  |     |         |                |
| CountryCode | char(3)   | NO  |     |         |                |
| District    | char(20)  | NO  |     |         |                |
| Population  | int(11)   | NO  |     | 0       |                |
+-------------+-----------+-----+-----+---------+----------------+
```

Field types and attributes

Each field (column) has an SQL data type, like **`char(20)`**.

Fields can have constraints (`not null`) and default values.

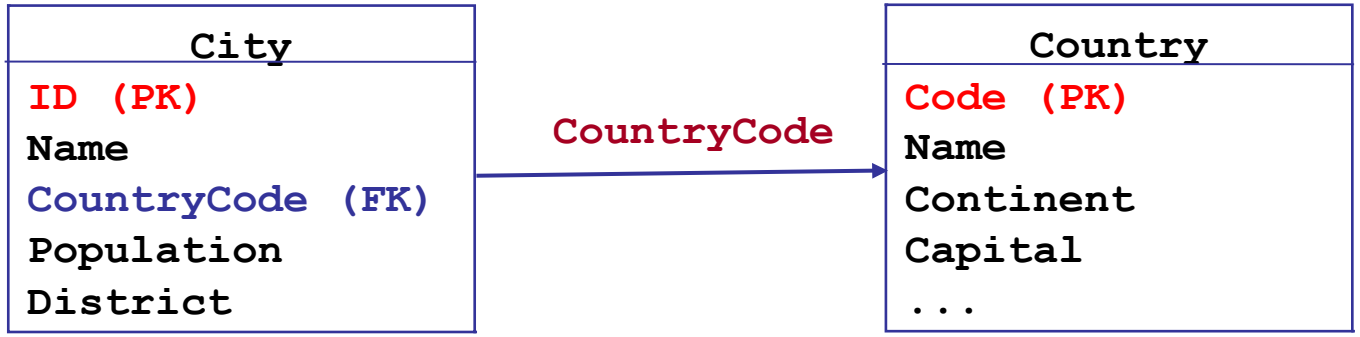
```
sql> SHOW columns FROM City;
+-------------+-----------+----+-----+---------+----------------+
| Field       | Type      | Null| Key | Default | Extra          |
+-------------+-----------+----+-----+---------+----------------+
| ID          | int(11)   | NO | PRI |         | auto_increment |
| Name        | char(35)  | NO |     |         |                |
| CountryCode | char(3)   | NO |     |         |                |
| District    | char(20)  | NO |     |         |                |
| Population  | int(11)   | NO |     | 0       |                |
+-------------+-----------+----+-----+---------+----------------+
```

A default value to use if value is not assigned explicitly.

Keys

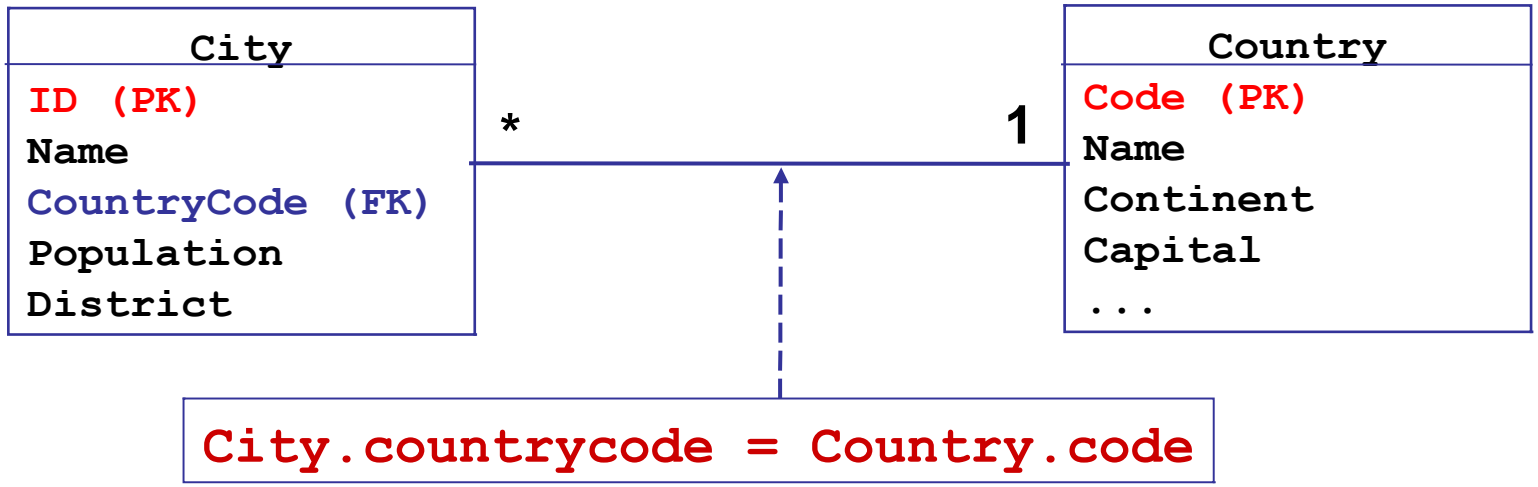
Every table should have a primary key that uniquely identifies each row.

```
sql> DESCRIBE Country;
+-------------+-----------+-----+-----+---------+----------------+
| Field       | Type      | Null| Key | Default | Extra          |
+-------------+-----------+-----+-----+---------+----------------+
| Code        | char(3)   | NO  | PRI |         |                |
| Name        | char(52)  | NO  |     |         |                |
| ...         |           |     |     |         |                |
```



Joining Tables

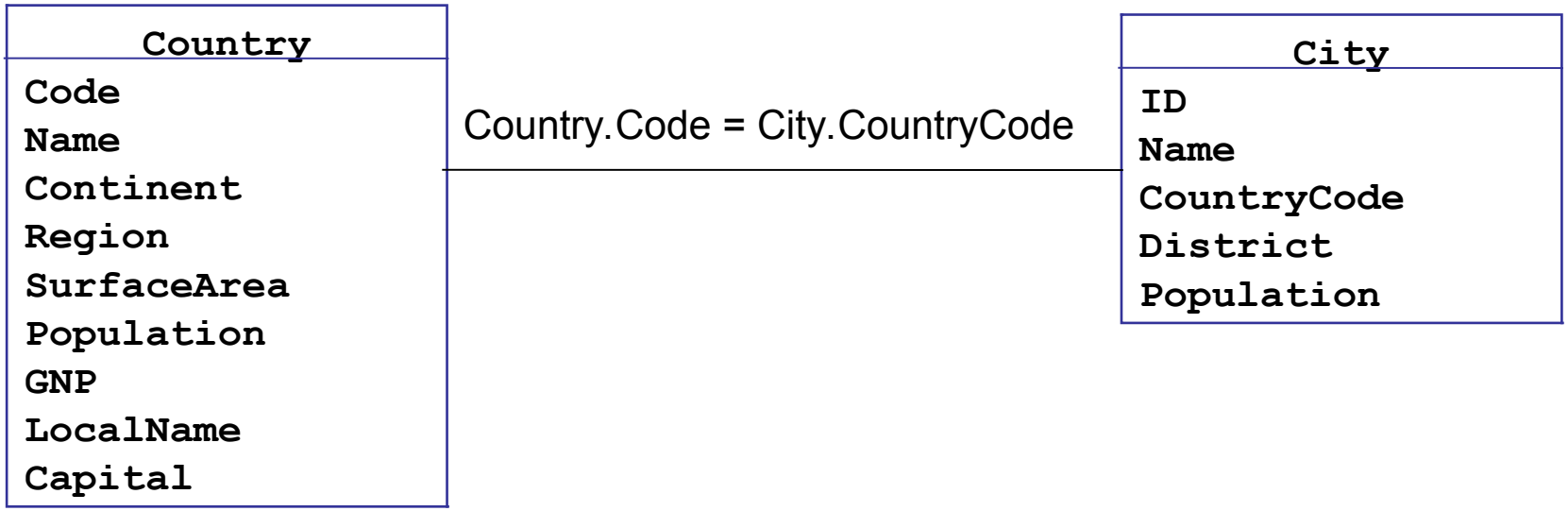
- Relate or "join" data in different tables.
- This is what makes an RDB so powerful and useful.
- City contains the CountryCode for the country it belongs to. This is called a **Foreign Key**.



Example: Join Country and City

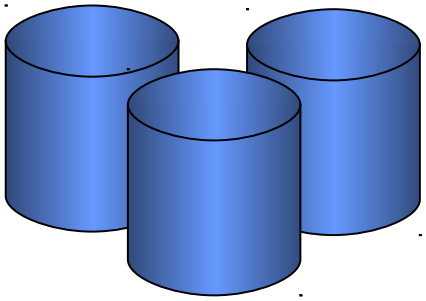
Use "table.field_name" to qualify a field name, e.g. `Country.code`. SQL to join Country and City:

```
SELECT City.Name, City.Population
   FROM   Country, City
   WHERE  Country.Code = City.CountryCode
   AND    Country.Name = 'Thailand';
```



Structure of a Database

- A database *system* may contain many **databases**.
- Each database is composed of **schema** and **tables**.

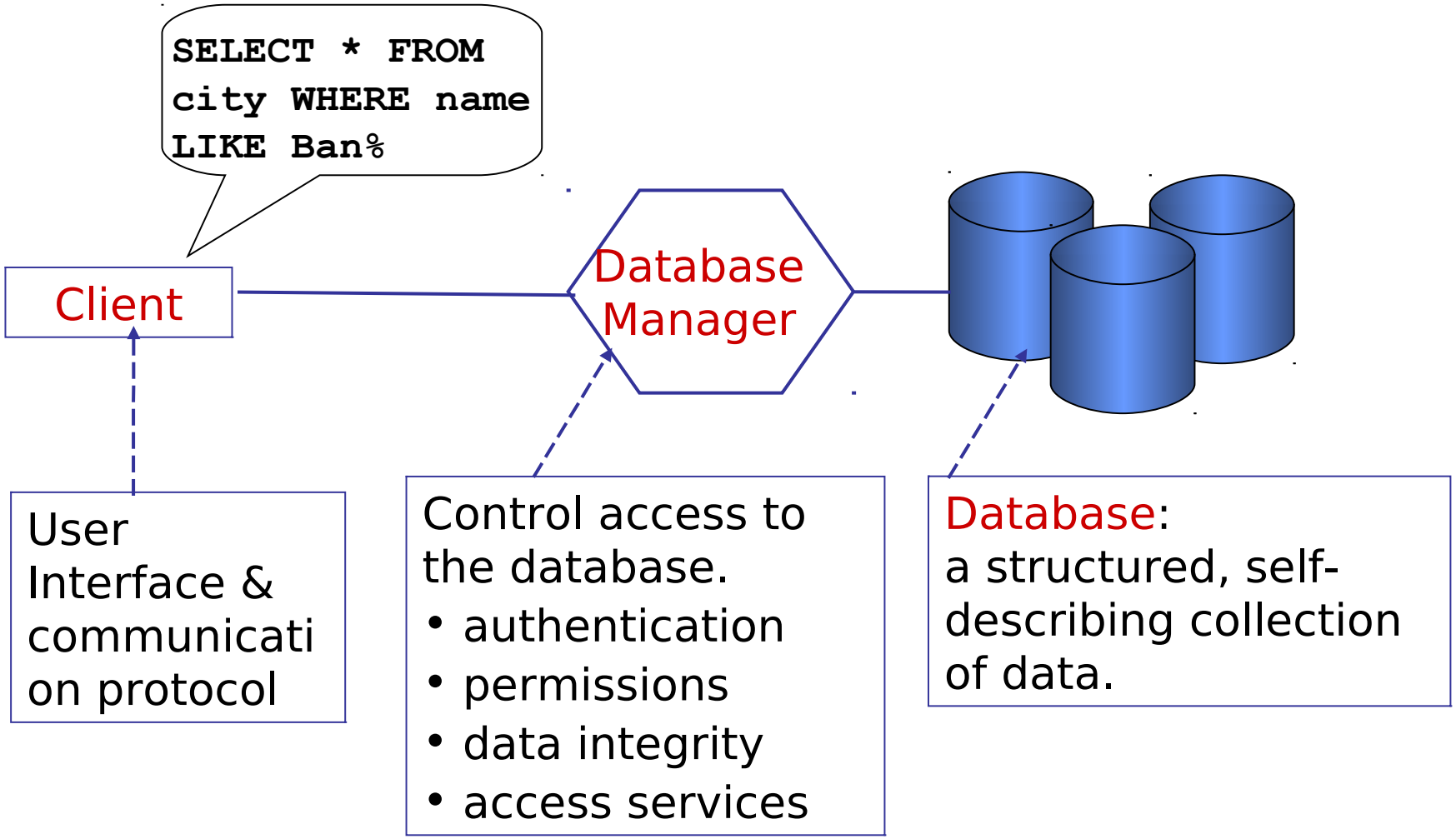


```
sql> SHOW databases;
+--------------+
| Database     |
+--------------+
| mysql        |
| test         |
| bank         |
| world        |
+--------------+
```

"shows databases" only shows db that the user has permission to access.

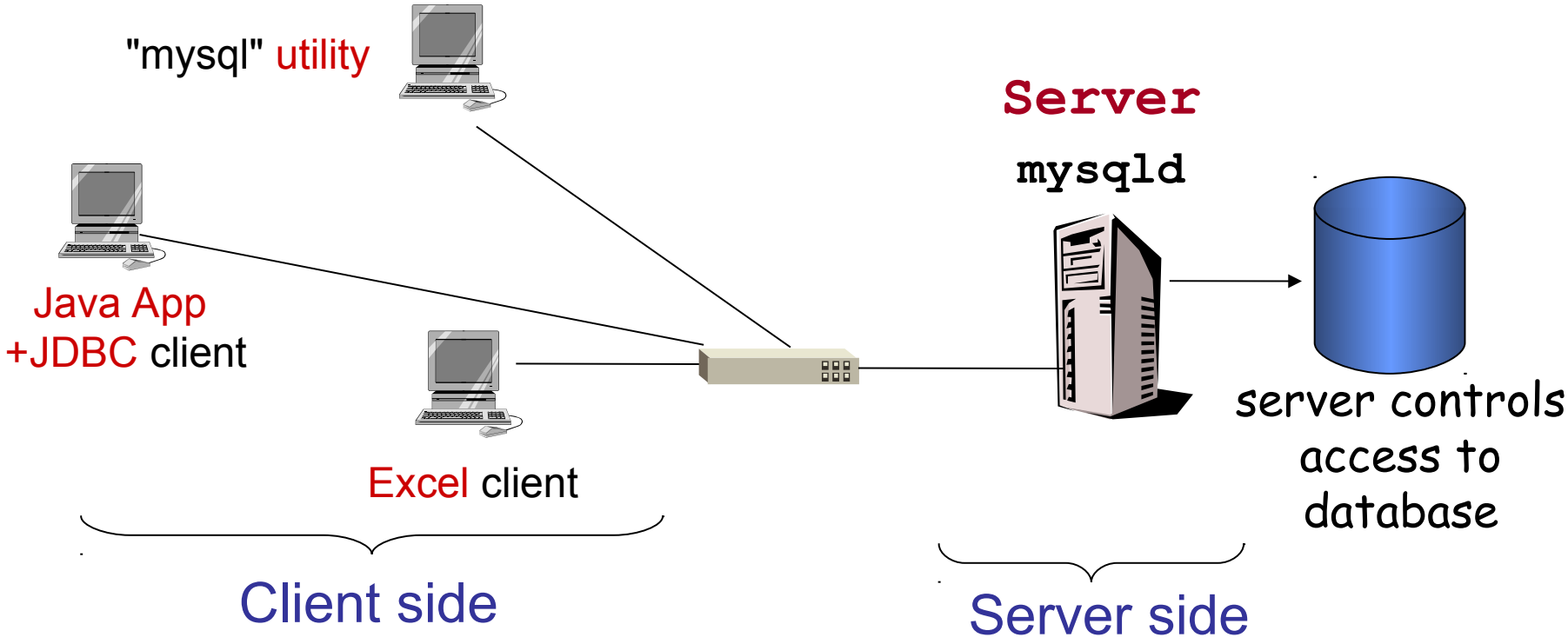
```
sql> USE world;
sql> SHOW tables;
+-----------------+
| Tables_in_world |
+-----------------+
| countries       |
| city            |
+-----------------+
```

Database Management System



Client - Server Databases

- Database Server is a separate *process* on a host.
- Clients can be on any machine.
- Many programs may be clients using a standard API.



4 Basic Database Operations

The 4 most common operations:

- SELECT query (search) the data
- INSERT add new records to a table(s)
- UPDATE modify existing record(s)
- DELETE delete record(s) from a table

What is CRUD?

Programmers call these operations "CRUD".

What does CRUD stand for?

Exercise: O-O Analogy of a Table?

Database	**Object Oriented Construct**
table	_____
record (row)	_____
fields (columns)	_____

records (rows)

```
+------+------------+--------------+----------+
| ID   | Name       | District     | Popula..}
+------+------------+--------------+----------+
| 3320 | Bangkok    | Bangkok      | 6320174 |
| 3321 | Nonthaburi | Nonthaburi   |  292100 |
| 3323 | Chiang Mai | Chiang Mai   |  171100 |
+------+------------+--------------+----------+
```

fields (columns)

Exercise

The database for the Django Polls project is db.sqlite3.

Use a database browser to answer some questions (separate file).

Tools

- sqlite3 command line tool, included with Sqlite
- sqlitebrowser free GUI tool. Works on all platforms.
- DBeaver popular database editor/browser that works with (almost) any database. Uses Java.