Databases - Relational Data Model

An abstract view

The abstract structure of a modern DBMS is a 3-level architecture as follows:

```
+-----------------+       +-----------------+       +-----------------+
| External Schema 1|       | External Schema 2|       | External Schema 3|
| Conceptual Schema|       | Physical Schema  |
| Storage          |       |                  |
```

External Schemata

An external schema is essentially a user’s view of the database.

A single external schema is designed for one particular group of users, and presents to them a particular view of the database tailored to their requirements — it provides a fairly high-level logical view of the data.

For example, an external schema may be used to
- Hide sensitive or irrelevant data from certain users
- Combine existing data in specialized ways

A database may have many external schemata (schemas).

Conceptual Schema

The conceptual schema is a logical description of the data that is actually contained in the database.

The conceptual schema describes what is contained in the database — it is a logical view of all the underlying data from which the external schemata can be created.

Constructing a conceptual schema is often a complicated and highly-skilled task, and in a large organization is one of the main roles of the database administrator (DBA).

A database has just one conceptual schema.
Physical Schema

The physical schema describes how the data is actually stored in the database.

Although many of the lowest level details (file names, compression etc) are part of the functionality of the DBMS itself, there are a range of choices that the DBA must make concerning the physical storage and the metadata.

- Choice of “storage engine”
- Choice of indexes

Although these choices do not affect the logical behaviour of the database, they can have a significant effect on its performance.

Data Models

A data model is an integrated collection of concepts for describing and manipulating data, the relationships between data and constraints on the data.

The data model specifies the general mechanism by which the user constructs the various schemata.

- Specifies the structure of the database
- Populates the database, i.e. enters data
- Queries the database
- Enforces integrity constraints

Many data models

Many data models have been proposed:

- Hierarchical Model
- Network Model
- Relational Model
- Object-Relational Model
- Object-Oriented Model

The hierarchical and network models are older models that provide only limited functionality, while the object-oriented model is not yet practical.

The dominant model

Currently, the relational model, introduced by Codd in 1970 is by far the dominant data model, and the vast majority of modern DBMS use this model.

- It is a declarative model both for specification and query
  This means that – to some extent – the user specifies what data they wish to specify or query and the DBMS then works out how to satisfy that request efficiently.

- It has just one fundamental concept called a relation
  This concept is expressive enough to model a useful portion of an organization’s activities, yet it is simple enough that it can be completely analysed mathematically.
A practical example

The description above is very theoretical so we will consider an extended example that will ground these concepts.

This uses an example derived from the following book

- MySQL
  - Paul DuBois
  - ISBN 0672326736

This book is a comprehensive MySQL reference and tutorial.

Relations

The fundamental concept of the relational data model is the relation. Later we will define a relation mathematically, but informally we can view a relation as a table consisting of rows and columns:

<table>
<thead>
<tr>
<th>Surname</th>
<th>Name</th>
<th>State</th>
<th>Born</th>
<th>Died</th>
</tr>
</thead>
<tbody>
<tr>
<td>Washington</td>
<td>George</td>
<td>VA</td>
<td>1799-12-14</td>
<td>1732-02-22</td>
</tr>
<tr>
<td>Adams</td>
<td>John</td>
<td>MA</td>
<td>1826-07-04</td>
<td>1735-10-30</td>
</tr>
<tr>
<td>Jefferson</td>
<td>Thomas</td>
<td>VA</td>
<td>1826-07-04</td>
<td>1743-04-13</td>
</tr>
<tr>
<td>Madison</td>
<td>James</td>
<td>VA</td>
<td>1836-06-28</td>
<td>1751-03-16</td>
</tr>
<tr>
<td>Monroe</td>
<td>James</td>
<td>VA</td>
<td>1831-07-04</td>
<td>1758-04-28</td>
</tr>
</tbody>
</table>

Part of a table describing American presidents.

Structure of a relation

A relation has a fixed number of columns, sometimes called attributes each of which has a name and a type.

Each row of a relation represents a single data item and specifies actual values for each of the attributes.

Therefore the relation AmericanPresident models an American president by his surname, name, state, birth and death date.

Each row of the table represents a specific American President.

A bit like a class!

```java
class AmericanPresident {
    String surname;
    String name;
    String state;
    Date birth;
    Date death;
}
```

There is a helpful correspondence

- A class description $\iff$ A conceptual schema
- An object $\iff$ A row in the table
**Structured Query Language** or SQL is the “standard” database language for specifying and querying a relational database. Initially developed by IBM in the mid-1970s, it was widely adopted and later standardized by ANSI/ISO.

- SQL-92
- SQL:1999
- SQL:2003
- SQL:2006→7→8?

Despite the existence of a standard version of SQL, every database vendor implements only an approximation to the standard.

---

**Creating a database**

A single MySQL server maintains a number of different databases. Each database consists of a number of tables — this explains our loose definition of a relation as a table.

Typically an administrator will create a database for a specific user or group of users, who would themselves have more limited privileges.

```
mysql> CREATE DATABASE sampdb;
Query OK, 1 row affected (0.12 sec)
```

The `mysql>` is called the prompt at which the user (or administrator) enters commands.

---

**Using a database**

A user can only be “using” one database at a time, and any table-names refer to tables of that name in the “currently used” database.

```
mysql> use sampdb;
Database changed
```

This restriction makes it possible for different databases to have tables of the same name with no risk of confusion.
Creating a table

Creating a table involves specifying the table name and then the names and types of the columns (attributes).

```sql
CREATE TABLE president
(
    last_name VARCHAR(15),
    first_name VARCHAR(15),
    state VARCHAR(2),
    birth DATE,
    death DATE
);
```

This command is part of the *data definition language* (DDL).

Did it work?

We can check to see if the table was created as we expected it to be by using the `DESCRIBE` command.

```sql
mysql> DESCRIBE president;
```

+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
|------------+-------------+------|-----+---------+-------+
| last_name  | varchar(15) | YES  |     | NULL    |       |
| first_name | varchar(15) | YES  |     | NULL    |       |
| state      | char(2)     | YES  |     | NULL    |       |
| birth      | date        | YES  |     | NULL    |       |
| death      | date        | YES  |     | NULL    |       |
+------------+-------------+------|-----+---------+-------+
5 rows in set (0.00 sec)

The response of the system gives us some unexpected details, but is basically clear.

MySQL types

SQL and hence MySQL have many different *data types*, some of them being familiar types such as `INT` while others are much less familiar to us.

The types used in `president` are

- `VARCHAR(15)`
  A variable-length string of up to 15 characters; when assigned a value MySQL stores both its value and its length.
- `CHAR(2)`
  A fixed-length string that holds *exactly* two characters.
- `DATE`
  A date value, stored in `CCYY-MM-DD` format.

Adding data

The *data manipulation language* (DML) consists of the commands that are used to insert, delete, modify and query the rows in a table.

```sql
mysql> INSERT INTO president VALUES
    | 'Washington', 'George', 'VA', '1732-02-22', '1799-12-14';
```

Query OK, 1 row affected (0.00 sec)

To insert a new row into a table, we use the command `INSERT INTO` and simply list the values for each of the columns/attributes.
Extended Example

Querying Data

The fundamental SQL command for querying data is the `SELECT` command.

```sql
mysql> SELECT * FROM president;
+------------+------------+-------+------------+------------+
| last_name  | first_name | state | birth      | death      |
+------------+------------+-------+------------+------------+
| Washington | George     | VA    | 1732-02-22 | 1799-12-14 |
| Adams      | John       | MA    | 1735-10-30 | 1826-07-04 |
| Jefferson  | Thomas     | VA    | 1743-04-13 | 1826-07-04 |
| Madison    | James      | VA    | 1751-03-16 | 1836-06-28 |
| Monroe     | James      | VA    | 1758-04-28 | 1831-07-04 |
+------------+------------+-------+------------+------------+
5 rows in set (0.00 sec)
```

The `*` is a wildcard character that means “everything” so this statement says “Select everything from the table `president`”.

(GF Royle 2006-8, N Spadaccini, 2008)

Extended Example

Restricting Queries I

Of course, we don’t usually want to select *everything* from a database, so we can specify exactly which columns we *do* want.

```sql
mysql> SELECT last_name, first_name FROM president;
+------------+------------+
| last_name  | first_name |
+------------+------------+
| Washington | George     |
| Adams      | John       |
| Jefferson  | Thomas     |
| Madison    | James      |
| Monroe     | James      |
+------------+------------+
5 rows in set (0.00 sec)
```

(GF Royle 2006-8, N Spadaccini, 2008)

Extended Example

Restricting Queries II

We can also select only certain *rows*:

```sql
mysql> SELECT * FROM president WHERE state = "MA";
+------------+------------+-------+------------+------------+
| last_name  | first_name | state | birth      | death      |
+------------+------------+-------+------------+------------+
| Adams      | John       | MA    | 1735-10-30 | 1826-07-04 |
+------------+------------+-------+------------+------------+
1 row in set (0.00 sec)
```

This command says to select each row that satisfies the *condition* expressed after the `WHERE` part of the statement.

(GF Royle 2006-8, N Spadaccini, 2008)

Extended Example

And much more . . .

A language such as SQL is not as expressive as a general purpose programming language — one reason for this is that every query must terminate.

In general this means that each new construct is just added to the language as another keyword with its own — sometimes complex — syntax.

Thus learning MySQL involves mastering the most important constructs, and always working with a copy of the documentation alongside!

(GF Royle 2006-8, N Spadaccini, 2008)