
# SQL

# Content

- What is SQL?
- One Rule – One Convention
- Comments

# Content

- What is SQL?
- One Rule – One Convention
- Comments

# SQL

## Structured Query Language

based on relational algebra


- *Easy to learn – hard to master*
- *Developers deal with SQL daily, yet very few speak SQL fluently.*

# Why?

## Structured Query Language

based on relational algebra


- *You need it anyway to get the data*
- *It's powerful and fast*

# Easy to learn…

## Short list of commands

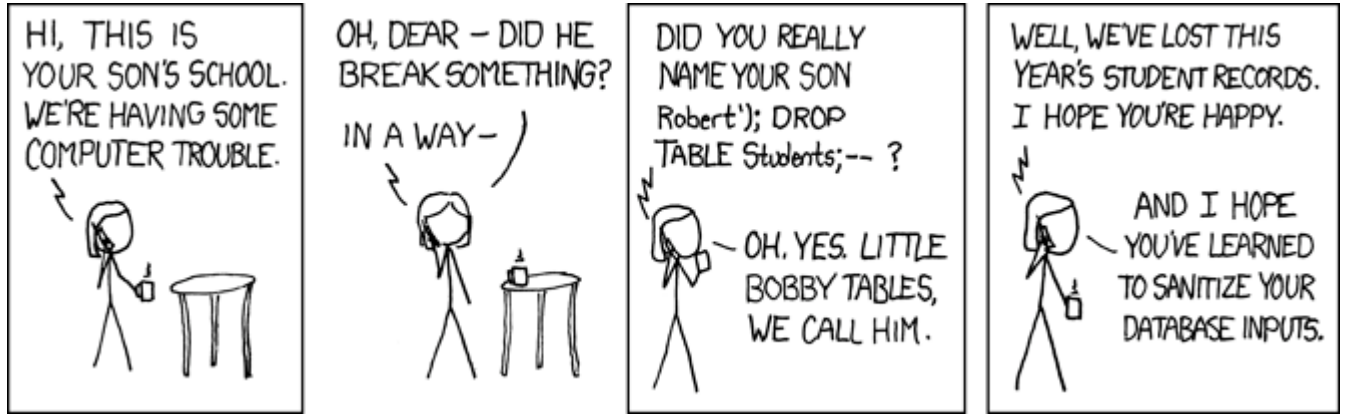
- CREATE
- DROP
- INSERT
- SELECT
- JOIN
- WHERE
- UPDATE

⚠ not exhaustive

Simple, yet powerful commands

*"With great power comes great responsibility"*

Benjamin Parker
*uncle of Peter Parker (Spiderman)*



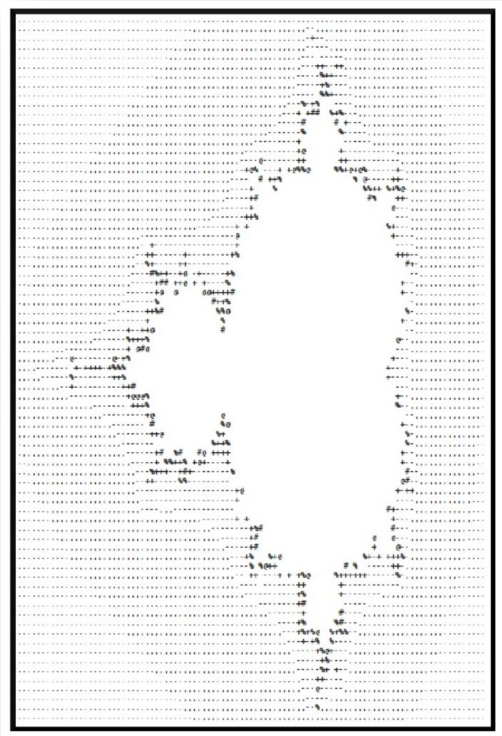

→ xkcd.com

# …hard to master

## Queries with high complexity

```
WITH RECURSIVE x(i) AS
(VALUES(0) UNION ALL SELECT i + 1 FROM x WHERE i < 101),
Z(Ix, Iy, Cx, Cy, X, Y, I)
AS (
    SELECT Ix, Iy, X::float, Y::float, X::float, Y::float, 0
    FROM (SELECT -2.2 + 0.031 * i, i FROM x) AS xgen(x,ix)
        CROSS JOIN
    (SELECT -1.5 + 0.031 * i, i FROM x) AS ygen(y,iy)
        UNION ALL
    SELECT
        Ix, Iy, Cx, Cy, X * X - Y * Y + Cx AS X,
        Y * X * 2 + Cy, I + 1
    FROM Z
    WHERE X * X + Y * Y < 16.0 AND I < 27),
Zt (Ix, Iy, I) AS (
    SELECT Ix, Iy, MAX(I) AS I
    FROM Z GROUP BY Iy, Ix
    ORDER BY Iy, Ix
)
SELECT array_to_string(
    array_agg(
        SUBSTRING(' .,,,-----++++%%%%@@@@#### ',
            GREATEST(I,1), 1)
    ),''
)
FROM Zt GROUP BY Iy ORDER BY Iy;
```

(yes, this query is SQL-spec compliant)



→ SlideShare, eggyknap, “*Fun With SQL*”

# …hard to master

## Queries with high complexity

```sql
select
GROUP_CONCAT(DISTINCT id) as id,
remarks,
begin_date,
end_date,
`type`,
subject_id,
class_id,
teachers,
GROUP_CONCAT(DISTINCT `day`) as `days`,
start_time,
end_time,
periods,
GROUP_CONCAT(DISTINCT `date`) as dates,
rooms
from (
    select
    GROUP_CONCAT(DISTINCT id) as id,
    remarks,
    begin_date,
    end_date,
    `type`,
    subject_id,
    class_id,
    teachers,
    `day`,
    start_time,
    end_time,
    periods,
    `date`,
    GROUP_CONCAT(DISTINCT room_id) as rooms
    from (
        select
        GROUP_CONCAT(DISTINCT id) as id,
        remarks,
        begin_date,
        end_date,
        `type`,
        subject_id,
        class_id,
        GROUP_CONCAT(DISTINCT teacher_id) as teachers,
        `day`,
        start_time,
        end_time,
        periods,
        `date`,
        room_id
        from (
            select
            GROUP_CONCAT(DISTINCT lessons.id) as id,
            lessons.remarks,
            DATE_FORMAT(STR_TO_DATE(lessons.begin_date, '%Y%m%d'), '%Y-%m-%d') as begin_date,
            DATE_FORMAT(STR_TO_DATE(lessons.end_date, '%Y%m%d'), '%Y-%m-%d') as end_date,
            lessons.`type`,
            lessons.subject_id,
            lessons.class_id,
            REPLACE(LOWER(lessons.teacher_id), 'tr_', '') as teacher_id,
            CASE lessons.`type` WHEN `current` THEN NULL ELSE lessons_times.`day` END as `day`,
            MIN(DATE_FORMAT(STR_TO_DATE(lessons_times.start_time, '%H%i'), '%H:%i')) as start_time,
            MAX(DATE_FORMAT(STR_TO_DATE(lessons_times.end_time, '%H%i'), '%H:%i')) as end_time,
            GROUP_CONCAT(DISTINCT lessons_times.timeperiod_id ORDER BY lessons_times.timeperiod_id) as periods,
            CASE lessons.`type` WHEN 'frame' THEN NULL ELSE DATE_FORMAT(STR_TO_DATE(lessons_times.`date`, '%Y%m%d'), '%Y-%m-%d') END as `date`,
            REPLACE(lessons_times.room_id, 'RM_', '') as room_id
            from zhawlsfm_bachelor_schedules_lessons as lessons
            right join zhawlsfm_bachelor_schedules_lessons_times as lessons_times on lessons.id = lessons_times.lesson_id
            left join zhawlsfm_bachelor_schedules_rooms as rooms on lessons_times.room_id = rooms.id
            left join zhawlsfm_bachelor_schedules_subjects as subjects on lessons.subject_id = subjects.id
            left join zhawlsfm_bachelor_schedules_classes as classes on lessons.class_id = classes.id
            left join zhawlsfm_bachelor_schedules_teachers as teachers on lessons.teacher_id = teachers.id
            group by
```

vs

```sql
SELECT
id,
fid,
term_begin_date,
term_end_date,
xml_lesson_id,
remarks,
begin_date,
end_date,
`type`,
subject_id,
class_id,
GROUP_CONCAT(teacher_id ORDER BY teacher_id),
periods,
`day`,
`date`,
start_time,
end_time,
room_id
FROM (
    SELECT
    ROUND(REPLACE(zhawlsfm_bachelor_schedules_lessons.xml_lesson_id, 'LS_', ''), -2) AS grpid,
    zhawlsfm_bachelor_schedules_lessons.id,
    zhawlsfm_bachelor_schedules_lessons.fid,
    zhawlsfm_bachelor_schedules_lessons.term_begin_date,
    zhawlsfm_bachelor_schedules_lessons.term_end_date,
    zhawlsfm_bachelor_schedules_lessons.xml_lesson_id,
    zhawlsfm_bachelor_schedules_lessons.remarks,
    zhawlsfm_bachelor_schedules_lessons.begin_date,
    zhawlsfm_bachelor_schedules_lessons.end_date,
    zhawlsfm_bachelor_schedules_lessons.`type`,
    zhawlsfm_bachelor_schedules_lessons.subject_id,
    zhawlsfm_bachelor_schedules_lessons.class_id,
    zhawlsfm_bachelor_schedules_lessons.teacher_id,
    GROUP_CONCAT(zhawlsfm_bachelor_schedules_lessons_times.timeperiod_id ORDER BY zhawlsfm_bachelor_schedules_lessons_times.timeperiod_id) AS periods,
    zhawlsfm_bachelor_schedules_lessons_times.`day`,
    zhawlsfm_bachelor_schedules_lessons_times.`date`,
    MIN(zhawlsfm_bachelor_schedules_lessons_times.start_time) as start_time,
    MAX(zhawlsfm_bachelor_schedules_lessons_times.end_time) as end_time,
    zhawlsfm_bachelor_schedules_lessons_times.room_id
    FROM
    zhawlsfm_bachelor_schedules_lessons
    INNER JOIN zhawlsfm_bachelor_schedules_lessons_times
    ON zhawlsfm_bachelor_schedules_lessons.id = zhawlsfm_bachelor_schedules_lessons_times.lesson_id
    WHERE zhawlsfm_bachelor_schedules_lessons.`type` = 'current'
    GROUP BY
    zhawlsfm_bachelor_schedules_lessons.fid,
    ROUND(REPLACE(zhawlsfm_bachelor_schedules_lessons.xml_lesson_id, 'LS_', ''), -2),
    zhawlsfm_bachelor_schedules_lessons.`type`,
    zhawlsfm_bachelor_schedules_lessons.teacher_id,
    zhawlsfm_bachelor_schedules_lessons_times.`day`,
    zhawlsfm_bachelor_schedules_lessons_times.`date`,
    zhawlsfm_bachelor_schedules_lessons_times.room_id
) lessons
GROUP BY
lessons.fid,
lessons.grpid,
lessons.`type`,
lessons.`day`,
lessons.`date`,
lessons.room_id
```

# Many Systems* – one Language

*DataBase-Management-Systems (DBMS)

MySQL

PostgreSQL

Microsoft SQL Server

MariaDB

⚠ again not exhaustive

- While all DBMS use SQL, there are differences!
- All the listed DBMS usually run on a server

*Is there a small, server-less DBMS?*

SQLite

# Alternatives?

## SEQUEL - since 1970

There are no alternatives!

There aren't any alternatives to SQL for speaking to relational databases,
but there are many alternatives to writing SQL in your applications.

- SchemeQL and CLSQL
- LINQ (in .Net)
- ScalaQL and ScalaQuery (in Scala)
- SqlStatement
- ActiveRecord
- HaskellDB
- Hibernate
- …

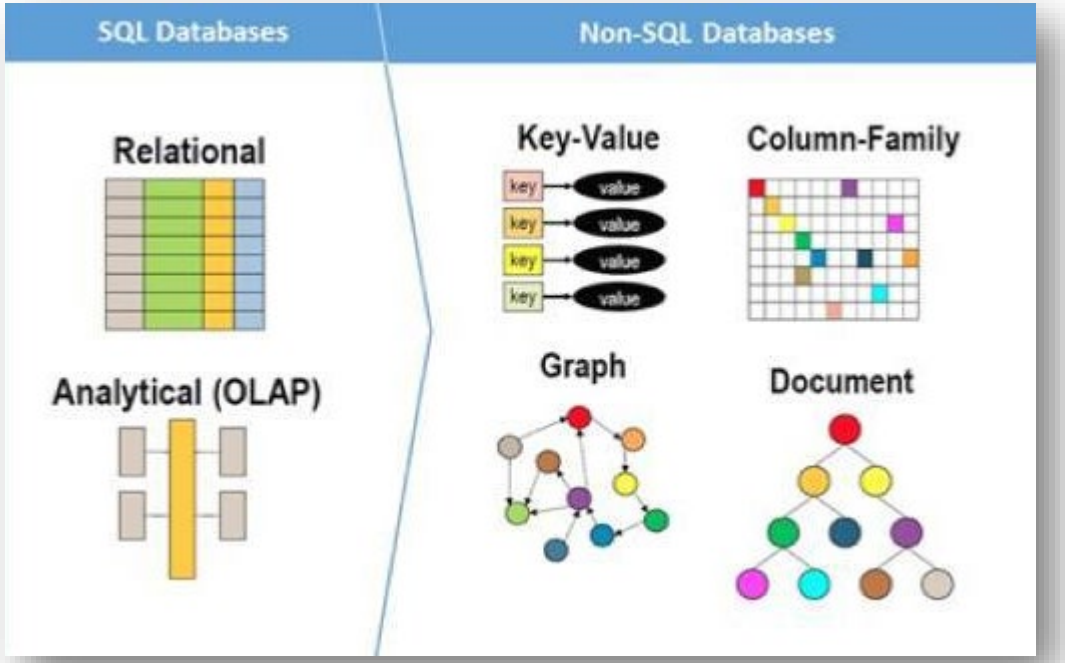
⚠ obviously not exhaustive

```
//Query syntax:
IEnumerable<int> numQuery1 =
    from num in numbers
    where num % 2 == 0
    orderby num
    select num;

//Method syntax:
IEnumerable<int> numQuery2 = numbers.Where(num => num % 2 == 0).OrderBy(n => n);
```

# NoSQL

## Not only SQL




Maybe a better name would be *SQLam* – SQL and more

You will still face SQL

# Content

- What is SQL?
- One Rule – One Convention
- Comments

# One Rule – One Convention

## Rule

SQL statements are terminated with semicolon (;)

```
SELECT * FROM Clients
WHERE Age>30 AND
      Age<50 OR
      NOT FirstName='Thomas';
```

# One Rule – One Convention

## Convention

It is common to write SQL commands in capital letters

```
SELECT * FROM Clients
WHERE Age>30 AND
      Age<50 OR
      NOT FirstName='Thomas';
```

# Content

- What is SQL?
- One Rule – One Convention
- Comments

# Comments

## Single Line

Single-line comments start with --

```
-- Select all
SELECT * FROM Clients;
```

# Comments

## Single Line

Multi-line comments are framed by **/*** and ***/**

```
/* Select all the columns
of all the records
in the Clients table */
SELECT * FROM Clients;
```