Data Management

# Historiek CRUD

## Introduction to databases

### The basis of a database … Data

* Data = better linked
* E.g.: Favourite Course can be separate database

A picture containing text, screenshot, font, number

Description automatically generated

A screenshot of a computer

Description automatically generated with low confidence

### Terminology

* A **Database** is a name for a system which **collects**, **organizes** and **links** data together
* A Database Management System (DBMS) is **software** that allows the **real data**  from a **database** to be **read, edited** and **deleted**

### Database types

|  |  |  |
| --- | --- | --- |
| Type | Description | Example |
| Relational DB | Database models with relations and constraints between tables | MySQL, MSSQL |
| Document DB | Store documents in a tree structure, most likely JSON format | MongoDB, DynamoDB |
| Graph DB | Store in a multidimensional structure with nodes, properties and connections (edges) | Neo4J |
| Time Series DB | when time is the most important feature(logging, search, analysis), use a Time Series Database | InfluxDB |
| Search Engines | Documents with live and fast searches with fulltext | ElasticsSearch |

### Relational Database Management System: RDBMS

* Based on the relation model, which organizes data into one or more tables with rows and columns
* E.g.:
  + Access
  + SQL Server
  + MySQL
  + MariaDB
  + PostgreSQL
  + SQLite
  + Oracle

### Historiek MySQL

* Open source
  + Sourcecode is available
  + In 2010 acquired by Oracle
  + The day Oracle announced the purchase of Sun, Michael “Montey” Widenius forked MySQL
* MariaDB
  + ‘Fork’ of MySQL
  + Works very similar
  + Better performance
* Part of LAMP stack
  + Linux – Apache – MySQL - PHP

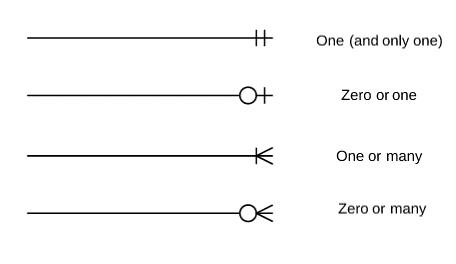
## More details regarding relational databases

### More Terminology – Relational DB

* Tables / Entities
  + Contains data which belongs together
  + Please ensure do duplicate or false data is / can be entered
* Fiels or columns
  + Contains certain datatype
* Record
  + 1 row in a table
* Keys
  + Primary Key
    - Identification record
    - Unique value in this column
    - Auto increment column
  + Foreign Key = Reference Key
    - Connects to a PK in another table and forms a relation
    - Is not unique in the column
  + Keys have the same datatype
  + Check integrity
  + Removing PK is not allowed if FK exist
* NOTNULL
  + Field cannot be a Null value
    - Null ≠ 0
* CRUD
  + Create
  + Read
  + Update
  + Delete
* Migrations
  + Changes to the structure or data of a database into a new version is called a migration
  + This also applies to data that has been merged together from multiple other databases

### Crow’s Foot Notation

A picture containing text, screenshot, font, diagram

Description automatically generated

* Cardinality
  + 1:1 One to One
  + 1:N One to Many
  + N:1 Many to One
  + N:M Many to Many
* 2 entities having a N:M relationship
  + A picture containing text, screenshot, font, diagram

    Description automatically generatedExtra table!!!

A picture containing text, screenshot, font, line

Description automatically generated

One lecturer resides in multiple buildings, and always needs at least 1 building.

One building contains multiple lecturers and needs at least 1 lecturer

## Data Manipulation Language

### Query database using SQL

* SQL: Structured Query Language
* T-SQL: Transact-SQL
  + Microsoft

### SELECT: Get values from one table

A screenshot of a computer

Description automatically generated with low confidenceA blue sign with white text

Description automatically generated with low confidence

* E.g.: SELECT Productname, PricePerUnit FROM tblProducts
* Multiple column names allow to fetch multiple fields
  + \* to fetch all
* Perform calculations on columns to reformat result
* Use AS to set a column name/alias
* If column has spaces: `Column Name`
* Concat() to combine 2 things into 1 field

### WHERE: filter values

* Used to extract only records that fulfill a condition
* E.g.: SELECT name, street, zipcode FROM tblcustomers WHERE city=’kortrijk’
* Comparison operators:
  + =, < , > , <= , >= ,**<> of !=**
* Arithmetics expressions
  + + - \* / % (module of rest na deling)
* Logical operators
  + AND, OR, NOT, XOR
* Isnull condition
  + IS NULL, IS NOT NULL

# DML Functions

## Comments

* Single line comment
  + -- This is a comment
* Multi-line comment
  + /\* This is a

Multi-line comment \*/

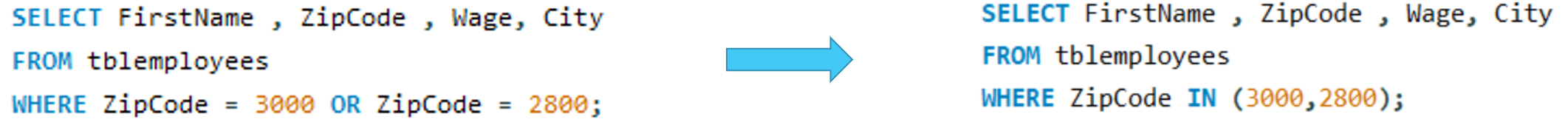
## Operators

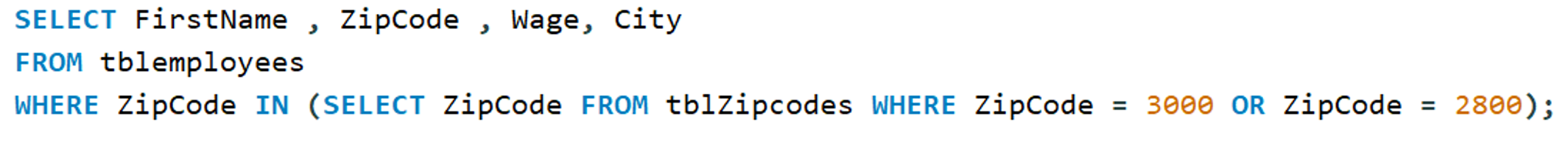
### BETWEEN Operator

* Used to search for values that are within a range of values
* Often in conjunction with AND keyword to specify upper and lower bounds of the range
* Inclusive
* <=> NOT BETWEEN

### IN Operator

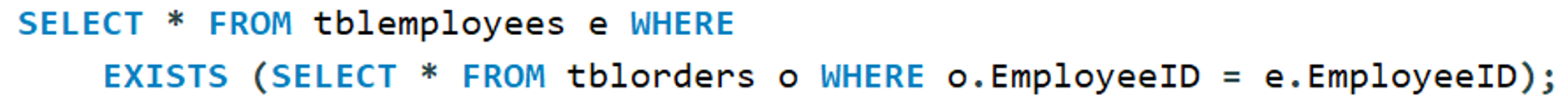
* Used to check wether value matches any value in a list of fixed values
* Often used as shorthand for series of OR conditions
* Can be used with a subquery
* <=> NOT IN





### EXISTS Operator

* Boolean operator used in a subquery to test whether inner query returns any rows
  + If true: outer query proceeds
  + If false: outer query does not execute -> returns nothing



* Will return all rows from tblemployees, where there is a row in tblorders with the same EmployeeID
* <=> NOT EXISTS

#### Difference between IN and EXISTS

A picture containing text, screenshot, font, number

Description automatically generated

A picture containing text, font, line, screenshot

Description automatically generated

### LIKE Operator

* Used to match a pattern with a string value to filter rows based on a pattern
* Not case sensitive
* Allows wildcards
  + %: any string of 0 or more characters
  + \_: any single character
  + \: escape character to use % or \_ as normal character instead of wildcard
* E.g.:
  + WHERE name LIKE ‘%Van’; returns all names with ‘van’ in it
  + WHERE name LIKE ‘\_ander’; returns all names 1 character followed by ander
  + WHERE name LIKE ‘V%n’; returns all names that start with v and end with n
  + ‘%\\_examle%’; any string that contains ‘\_example’

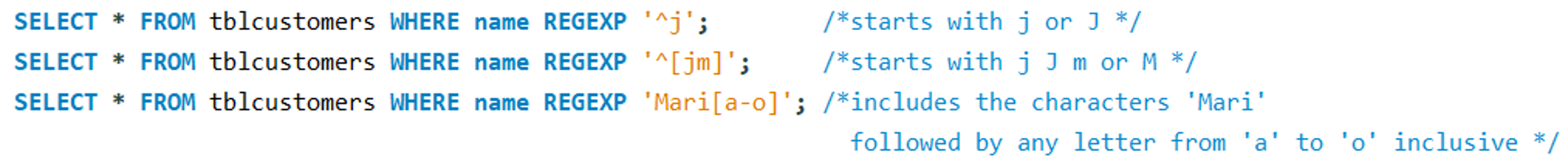


* /: delimiters used to enclose expression
* ^: matches start of a line
* \(: opening of parenthesis with escape \
* 0: digit 0
* \d{2}: 2 digits, \d: digit, {2}: 2 of them
* \): closing of parenthesis with escape \
* \s: any whitespace character
* \d{3}: 3 digits
* -: -
* \d{4}: 4 digits
* $: end of line
* /: delimiter to enclose expression pattern
* g: global

E.g.: ‘(012) 345-6184’

### Regular Expressions - REGEXP

* Powerful tool for pattern matching
* Allow for complex patterns within a string
* 3 types of metacharacters
  + matching metacharacters
  + counting methacharacters
  + positional metacharacters
* REGEXP BINARY for case sensitive search
* Not all metacharacters supported in MySQL
  + E.g.: \d digit doesnt work
  + Use . instead (matches any character)





A screenshot of a computer code

Description automatically generated with low confidence

A screenshot of a computer

Description automatically generated with low confidence A picture containing text, screenshot, font, number

Description automatically generated

### ORDER BY

* Used to sort results of query based on 1 or more columns
* After where clause
* ASC ⬄ DESC

### LIMIT

* Used to limit number of rows returned by query
* After ORDER BY

### DISTINCT

* Return only unique values from query result
* Immediately after SELECT

## MySQL Functions

* Built-in operations is MySQL
* 2 types
  + Aggregate functions
  + Scalar functions

### Why?

* To concatenate columns
* To convert data
* To extract specific parts
* To perform calculations
* To format data
* To search for data
* To replace data

### Aggregate Functions

* Operate on a set of rows
* Return single value
* Can be used to perform calculations
* E.g.: AVG(), SUM(), MIN(), COUNT()

### Scalar Functions

* Operate on single value
* Return single value
* Can be used to perform tasks such as
  + Formatting data
  + Converting data types
  + Manipulating strings
* E.g.: CONCAT(), UPPER(), LOWER(), DATE\_FORMAT(), SUBSTRING(), REPLACE(), CAST()

## NULL

* IS NULL:
  + Operator
  + Column\_name IS NULL
  + If column contains a NULL value: true
* ISNULL():
  + Function
  + ISNULL(column\_name)
  + If column contains a NULL value: true
* IFNULL():
  + Function to replace NULL values
  + IFNULL(column\_name, replacement\_value)