1 Table of contents

2 Objective

1. Understanding basic concepts of Relational Databases
2. DDL sentences
    a. How to implement constraints in relational databases

3 Terminology in RDBMS

Relational Database Concepts	Equivalent Database Concepts
Relation	Table
Tuple	Row or record
Attribute	Column or field
Cardinality	Number of rows of a table
Degree	Number of columns of a table
Domain	Pool of legal values

Relational Database Concepts	Equivalent Database Concepts
Key	**Field or fields** in a table that identify one record
Super Key	The set of attributes which can uniquely identify a tuple is known as Super Key: • Adding attributes to candidate key generates super key
Candidate key	The minimal set of attribute which can uniquely identify a tuple is known as candidate key
Primary key	There can be more than one candidate key in a relation out of which one can be chosen as primary key
Foreign key	If an attribute/s can only take the values which are present as values of some other attribute/s, it will be foreign key to the attribute/s to which it refers. The relation which is being referenced is called referenced relation and corresponding attribute is called referenced attribute and the relation which refers to referenced relation is called referencing relation and corresponding attribute is called referencing attribute
NULL	It is used to signify *missing* or *unknown* values (a non mandatory value). The keyword NULL is used to indicate these values. NULL really isn't a specific value as much as it is an indicator. Don't think of NULL as similar to zero or blank, it isn't the same. Zero (0) and blanks " ", are values
Referential Integrity	In the context of relational databases, it requires every value of one or more columns of a table to exist as a value of another columns in a different (or the same) table **OR have null value**

4 Data Type in ANSI SQL

4.1 Introduction

Standard data types in ANSI SQL can change their names depending on the RDBMS
https://www.w3schools.com/sql/sql_datatypes.asp

4.2 Strings

- CHARACTER(*n*) or CHAR(*n*) — fixed-width *n*-character string, padded with spaces as needed
- CHARACTER VARYING(*n*) or VARCHAR(*n*) — variable-width string with a maximum size of *n* characters
- NATIONAL CHARACTER(*n*) or NCHAR(*n*) — fixed width string supporting an international character set
- NATIONAL CHARACTER VARYING(*n*) or NVARCHAR(*n*) — variable-width NCHAR string

4.3 Bit strings

- BIT(*n*) — an array of *n* bits
- BIT VARYING(*n*) — an array of up to *n* bits

4.4 Numeric

- INTEGER and SMALLINT
- FLOAT, REAL and DOUBLE PRECISION
- NUMERIC(*precision*, *scale*) or DECIMAL(*precision*, *scale*)
    - E.g. 123.45 has a precision of 5 and a "scale" of 2

4.5 Date and time

- DATE — for date values (e.g., 2011-05-03)
- TIME — for time values (e.g., 15:51:36). The granularity of the time value is usually a tick (100 nanoseconds).
- TIME WITH TIME ZONE or TIMESTAMP — the same as TIME, but including details about the time zone in question.
- TIMESTAMP — this is a DATE and a TIME put together in one variable (e.g., 2011-05-03 15:51:36).
- TIMESTAMP WITH TIME ZONE or TIMESTAMPTZ — the same as TIMESTAMP, but including details about the time zone in question.

5 Data types in MySQL 5.0

http://dev.mysql.com/doc/refman/5.0/es/column-types.html

6 Constraints of Relational Data Model.

6.1 Inherent model-based constraints or implicit constraints

They are those which are Inherent in the data model. For example:

- Duplicate tuples are not allowed in a relation
- The records have no particular order
- Every field is single-valued
- The records have a unique identifying field or composite field, called the primary key field

6.2 Semantic Integrity Constraint Types

https://info.teradata.com/HTMLPubs/DB_TTU_16_00/index.html#page/Database_Management/B035-1094-160K/pzj1472240593571.html

Business rules are expressed in relational databases by means of various types of integrity constraints. Four types of integrity constraint are supported:

- Domain: Fundamentally, a domain is a data type with a set of valid values for a column
- Column-Level: Column-level constraints define more elaborate integrity rules for columns than those defined by simple domain constraints. For example, a column constraint on a column typed as INTEGER might further specify that only values between 0 and 49999 or between 99995 and 99999 are permitted for that column.
    - E.g. CONSTRAINT emp_no CHECK (emp_no >= 10001 AND emp_no <= 32001)
- Primary key
- Unique
- Mandatory or not (not null or null)
- Referential integrity

7 Create, alter, delete of Databases en MYSQL

- **CREATE DATABASE.**

The database can be understood as the container of tables.

```
CREATE {DATABASE | SCHEMA} [IF NOT EXISTS] db_name
    [create_specification] ...

create_specification:
    [DEFAULT] CHARACTER SET [=] charset_name
  | [DEFAULT] COLLATE [=] collation_name
```

DATABASE or SCHEMA is equivalent. SCHEMA is used in Oracle.

For the exercises we have to create the Database called EjemplosClase.

- To modify the database definition: **ALTER DATABASE**

```
ALTER {DATABASE | SCHEMA} [db_name]
    alter_specification ...
ALTER {DATABASE | SCHEMA} db_name
    UPGRADE DATA DIRECTORY NAME

alter_specification:
    [DEFAULT] CHARACTER SET [=] charset_name
  | [DEFAULT] COLLATE [=] collation_name
```

- To delete database and its contents: **DROP DATABASE**

```
DROP {DATABASE | SCHEMA} [IF EXISTS] db_name
```

8 CREATE TABLE in MySQL

The reduced version is as follow.
For a complete definition: http://dev.mysql.com/doc/refman/5.0/es/create-table.html

```
CREATE [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name
    [(create_definition,...)]
    [table_options] [select_statement]
```

```
create_definition:
    column_definition
  | [CONSTRAINT [symbol]] PRIMARY KEY [index_type] (index_col_name,...)
  | INDEX [index_name] [index_type] (index_col_name,...)
  | [CONSTRAINT [symbol]] UNIQUE [INDEX]
        [index_name] [index_type] (index_col_name,...)
  | [CONSTRAINT [symbol]] FOREIGN KEY(index_col_name,...) [reference_definition]
  | CHECK (expr)

column_definition:
    col_name type [NOT NULL | NULL] [DEFAULT default_value]
        [AUTO_INCREMENT] [UNIQUE [KEY] | [PRIMARY] KEY]
        [COMMENT 'string'] [reference_definition]

reference_definition:
    REFERENCES tbl_name [(index_col_name,...)]
               [ON DELETE reference_option]
               [ON UPDATE reference_option]

reference_option:
    RESTRICT | CASCADE | SET NULL | NO ACTION
```

You can use the `TEMPORARY` keyword when creating a table. A `TEMPORARY` table is visible only within the current session, and is dropped automatically when the session is closed. For more information, see Section 13.1.18.3, "CREATE TEMPORARY TABLE Syntax".

8.1 Table options in MySQL

```
table_option:
    {ENGINE|TYPE} = engine_name
  | AUTO_INCREMENT = value
  | [DEFAULT] CHARACTER SET charset_name [COLLATE collation_name]
  | COMMENT = 'string'
  | MAX_ROWS = value
  | MIN_ROWS = value
  | DATA DIRECTORY = 'absolute path to directory'
  | INDEX DIRECTORY = 'absolute path to directory'
```

https://dev.mysql.com/doc/refman/8.0/en/create-table.html#create-table-options

Look for:

- ENGINE
- CHARACTER SET
- DATA DIRECTORY

9 CREATE TABLE Exercise 1

Create a table which for storing students' data:

- Registration Number,
- DNI
- Name of a maximum of 50 characters. If you don't know the name it should be "Unknown" as the default value
- Birth date
- Average mark form bachillerato from 0 to 10.
- Sex with allowed values "M" o "F"
- We want to know the timestamp (CPU time) when the student was created
- We want to know if the student has abandoned its students (TRUE o FALSE). By default it has to be FALSE.

Registration numbed and DNI cannot be repeated.

The solutions are written for Access and MySQL
Note: In MySQL before creating the tables, we have to create the database.

CREATE SCHEMA `ejemplosClase` ;
O bien
CREATE DATABASE `ejemplosClase` ;

9.1 1st .Approach. No restriction

In this approach tables are created without considering any constraint, neither inherent to relational model nor semantic one

Access 2002	MySQL 5.5
Create table alumno1 (	**Create table** ejemplosclase.alumno1 (

```
        nMatricula int,
        dni char(10),
        nombre varchar(50),
        fechaNacimiento date,
        notaMedia numeric,
        sexo char(1),
        baja char(1),
        fechaCPU Timestamp
);
```

```
        nMatricula int,
        dni char(10),
        nombre varchar(50),
        fechaNacimiento date,
        notaMedia decimal(4,2),
        sexo char(1),
        baja boolean,
        fechaCPU Timestamp
);
```

Comments:

1. There is no primary key. So dni or nMatricula can be repeated.
2. Access
    a. Search how to express the number of decimals using the design view in access
    b. Search how to define a Boolean value in access (design view→ validation rules)

9.2 2nd Approach. Null and default values

We can state which columns admit null values (non mandatory fields) as well as which is the default value. Only written in MySQL
The default value can be written for a mandatory or non mandatory field.

MySQL 5.5

```
Create table ejemplosclase.alumno2(
        nMatricula int NOT NULL,
        dni char(10)    NOT NULL,
        nombre varchar(50) NOT NULL DEFAULT 'Unknown',
        fechaNacimiento date NULL,
        notaMedia decimal(4,2) NOT NULL,
        sexo char(1) NOT NULL,
        baja boolean NOT NULL default FALSE,
        fechaCPU Timestamp default CURRENT_TIMESTAMP()
);
```

Comments:

- nombre is mandatory, but if you don't know the value the database will consider the default value "Unknown",. It applies also to field Baja, by default it will be 'False'
- The key fields must be declared as NOT NULL
- If you don't say anything about NULL or NOT NULL, the field is considered as NULL
- String fields must be between quotation marks. They can be single or double depending of the RDBMS
- Numeric and Boolean fields must be without quotation marks
- We can also use functions as default value. A function will return one value:
    - E.g. NOW(), CURRENT_TIMESTAMP(), CURRENT_TIME(), CURRENT_DATE()

9.3 3rd Approach. Primary Key

The previous tables couldn't be edited in MySQL because the primary key is not defined.
We will consider for this example that the primary key is the Registration number (nMat)

We have to add the constraint Primary Key to the table. The constraint can be written following the field definition (but only in the case that the primary key is only one field). **The suggestion** is to write it **always after the definition of all columns**

MySQL 5.5

```
Create table ejemplosclase.alumno3(
        nMatricula int NOT NULL Primary Key,
        dni char(10)    NOT NULL,
        nombre varchar(50) NOT NULL DEFAULT 'Unknown',
        fechaNacimiento date NULL,
        notaMedia decimal(4,2) NOT NULL,
        sexo char(1) NOT NULL,
        baja boolean NOT NULL default FALSE,
        fechaCPU Timestamp default CURRENT_TIMESTAMP()
);
```

If the primary key is more than one field we have to use the following syntax:

```
Create table ejemplosclase.alumno4(
        nMatricula int NOT NULL,
        dni char(10)    NOT NULL,
        nombre varchar(50) NOT NULL DEFAULT 'Unknown',
        fechaNacimiento date NULL,
        notaMedia decimal(4,2) NOT NULL,
        sexo char(1) NOT NULL,
        baja boolean NOT NULL default FALSE,
        fechaCPU Timestamp default CURRENT_TIMESTAMP(),
        PRIMARY KEY (nMatricula)
);
```

Constraints SHOULD have a NAME!!!!. It will be useful to know which restriction is not accomplished. If we don't choose a name, the system will do for us!

```
Create table ejemplosclase.alumno5(
        nMatricula int NOT NULL,
        dni char(10)    NOT NULL,
        nombre varchar(50) NOT NULL DEFAULT 'Unknown',
        fechaNacimiento date NULL,
        notaMedia decimal(4,2) NOT NULL,
        sexo char(1) NOT NULL,
        baja boolean NOT NULL default FALSE,
        fechaCPU Timestamp default CURRENT_TIMESTAMP(),
        Constraint PK_alumno5 PRIMARY KEY (nMatricula)
);
```

Comments:

- We haven't resolved already the uniqueness for the field dni

9.4 4th Approach. UNIQUE constraint

1. **After the column definition. This syntax is only possible when the alternate key is only one field**

```
Create table ejemplosclase.alumno6(
        nMatricula int NOT NULL,
```

```
        dni char(10)    NOT NULL UNIQUE,
        nombre varchar(50) NOT NULL DEFAULT 'Unknown',
        fechaNacimiento date NULL,
        notaMedia decimal(4,2) NOT NULL,
        sexo char(1) NOT NULL,
        baja boolean NOT NULL default FALSE,
        fechaCPU Timestamp default CURRENT_TIMESTAMP(),
        Constraint PK_alumno6 PRIMARY KEY (nMatricula)
);
```

2. After the definition of Primary Key constraint. This is the RECOMMENDED option

```
Create table ejemplosclase.alumno7(
        nMatricula int NOT NULL,
        dni char(10)    NOT NULL,
        nombre varchar(50) NOT NULL DEFAULT 'Unknown',
        fechaNacimiento date NULL,
        notaMedia decimal(4,2) NOT NULL,
        sexo char(1) NOT NULL,
        baja boolean NOT NULL default FALSE,
        fechaCPU Timestamp default CURRENT_TIMESTAMP(),
        Constraint PK_alumno7 PRIMARY KEY (nMatricula),
        UNIQUE (dni)
);
```

3. And the following is more recommended. Always name constraints.

```
Create table ejemplosclase.alumno8(
        nMatricula int NOT NULL,
        dni char(10)    NOT NULL,
        nombre varchar(50) NOT NULL DEFAULT  'Unknown',
        fechaNacimiento date NULL,
        notaMedia decimal(4,2) NOT NULL,
        sexo char(1) NOT NULL,
        baja boolean NOT NULL default FALSE,
        fechaCPU Timestamp default CURRENT_TIMESTAMP(),
        Constraint PK_alumno8 PRIMARY KEY (nMatricula),
        Constraint UQ_dni UNIQUE (dni)
);
```

9.5 5th Approach. CHECK for validation rules

Work is not finished yet. There are more semantic constraints:

- Values for sex attribute: ‘M’ o ‘F’
- Average note: between 0 and 10

We’re going to use CONSTRAINT CHECK for data validation:

```
Create table ejemplosclase.alumno9(
        nMatricula int NOT NULL,
        dni char(10)    NOT NULL UNIQUE,
        nombre varchar(50) NOT NULL DEFAULT 'Unknown',
        fechaNacimiento date NULL,
        notaMedia decimal(4,2) NOT NULL,
        sexo char(1) NOT NULL,
        baja boolean NOT NULL default FALSE,
```

```
        fechaCPU Timestamp default CURRENT_TIMESTAMP(),
        Constraint PK_Alumnos9  PRIMARY KEY (nMatricula),
        Constraint UQ_dni UNIQUE (dni),
        Constraint cNotaMedia CHECK ( notaMedia>=0 and notaMedia <=10),
        Constraint cSexo CHECK (sexo='M' or sexo='F')
);
```

Comments:

- SURPRISE!!! : If we don't write it correctly the compiler will give us an error. But, then, MySQL doesn't validate it!!!! At least with 5.5 version
  - http://dev.mysql.com/doc/refman/5.5/en/create-table.html
- Operators : "between" and "in"
  - CHECK (notamedia between 0 and 10)
  - CHECK (sexo in ('M', 'F'))

10 CREATE TABLE Exercise 2

We also want to know who the tutor of a student is. The tutor has to be one of the teachers. We want also to store the teacher's data from whom we need:

- Código /code
- Nombre / name
- Edad /age

The teacher will be identified by a unique code and the rest of the data will be mandatory.

One student will have as much 1 tutor . One professor can be the tutor of several students.

Further explanations:

In order to be able to answer the question "I want to know the tutor's name of a student or students" is necessary to create a relationship between the student and teacher's tables. That's to say:

1. Table teacher and student have to exist
2. Teacher table's primary key will be "codigo"
3. In the students table there have to be one or several fields which need to be related with the primary key of teacher table . They will be the foreign key

An example of possible table contents is as follows:

Student Table

nMat	Dni	nombre	fNac	nota	sexo	baja	fechaCPU	codTutor
1	D1	N1	19900101	5	M	F	20111119171314	1
2	D2	N2	19890101	5	F	F	20111119171314	NULL

Teacher Table

codigo	Nombre	Edad
1	N1	45
2	N2	48

  - Please note: If the tutor value for a student were compulsory the null value wouldn't be allowed

10.1 Step 1. Creation of tables

As student table will refer teacher table, they must be created in the right order (first Teacher, after Student),

Teacher

```
Create table ejemplosclase.teacher(
        Codigo int NOT NULL,
```

```
        nombre varchar(50) Not null,
        edad int not null,
        Primary key (codigo)
);
```

Student

```
Create table ejemplosclase.student10(
        nMatricula int NOT NULL,
        dni char(10)    NOT NULL UNIQUE,
        nombre varchar(50) NOT NULL DEFAULT 'Unknown',
        fechaNacimiento date NULL,
        notaMedia decimal(4,2) NOT NULL,
        sexo char(1) NOT NULL,
        baja boolean NOT NULL default FALSE,
        fechaCPU Timestamp default CURRENT_TIMESTAMP(),
        codTutor  int,
        Constraint PK_student10 PRIMARY KEY (nMatricula),
        Constraint uq_dni UNIQUE (dni)
        Constraint cNotaMedia CHECK ( notaMedia>=0 and notaMedia <=10),
        Constraint cSexo CHECK (sexo='M' or sexo='F'),
        Constraint FK_student_teacher FOREIGN KEY (codTutor) References Teacher(codigo)
);
```

10.2 Step 2. Understanding referential integrity

We have to answer the following questions:

1. What happens if we want to delete a teacher who is the tutor of one or more student?
   The previous sentence won't allow it

   ```
   Constraint FK_student_teacher FOREIGN KEY (codTutor) References Teacher(codigo)
   ```

   What we have written is equivalent to

   ```
   Constraint FK_student_teacher FOREIGN KEY (codTutor) References Teacher(codigo) ON DELETE RESTRICT
        or
   Constraint FK_student_teacher FOREIGN KEY (codTutor) References Teacher(codigo) ON DELETE NO ACTION
   ```

2. What can we do if, at the same time we delete a teacher we want to delete all the students who have a relation with the teacher we want to delete ? **(NORMALLY THIS IS NOT RECOMMENDED)**

   ```
   Constraint FK_student_teacher FOREIGN KEY (codTutor) References Teacher(codigo) ON DELETE CASCADE
   ```

3. And what can we do if we want to delete the teacher and leave the students without a tutor?

   ```
   Constraint FK_student_teacher FOREIGN KEY (codTutor) References Teacher(codigo) ON DELETE SET NULL
   ```

When the RDBS allows updating the value of a primary key, we can write the same possibilities as above but with "… ON UPDATE"

```
Constraint FK_student_teacher FOREIGN KEY (codTutor) References Teacher(codigo) ON UPDATE RESTRICT
Constraint FK_student_teacher FOREIGN KEY (codTutor) References Teacher(codigo) ON UPDATE CASCADE
Constraint FK_student_teacher FOREIGN KEY (codTutor) References Teacher(codigo) ON UPDATE SET NULL
```

11 Alter tables

11.1 Alter Table syntax in MySQL

ALTER TABLE allows changing or altering the structure of an existing table. For instance, we can add or delete columns, create or drop indexes, change the data type of existing columns or rename the column or table name….

Reduced syntax for Alter Table

```
ALTER  TABLE tbl_name
    alter_specification [, alter_specification] ...
```

```
alter_specification:
    ADD [COLUMN] column_definition [FIRST | AFTER col_name ]
  | ADD [COLUMN] (column_definition,...)
  | ADD INDEX [index_name] [index_type] (index_col_name,...)
  | ADD [CONSTRAINT [symbol]]
        PRIMARY KEY [index_type] (index_col_name,...)
  | ADD [CONSTRAINT [symbol]]
        UNIQUE [index_name] [index_type] (index_col_name,...)
  | ADD [CONSTRAINT [symbol]]
        FOREIGN KEY [index_name] (index_col_name,...)
        [reference_definition]
  | ALTER [COLUMN] col_name {SET DEFAULT literal | DROP DEFAULT}
  | CHANGE [COLUMN] old_col_name column_definition
        [FIRST|AFTER col_name]
  | MODIFY [COLUMN] column_definition [FIRST | AFTER col_name]
  | DROP [COLUMN] col_name
  | DROP PRIMARY KEY
  | DROP INDEX index_name
  | DROP FOREIGN KEY fk_symbol
  | RENAME [TO] new_tbl_name
  | CONVERT TO CHARACTER SET charset_name [COLLATE collation_name]
  | [DEFAULT] CHARACTER SET charset_name [COLLATE collation_name]
  | table_options
```

11.2 Exercise 3

Perform the following changes to the tables from exercise 2

1. Alter student table to change the restriction of the average note. Now the possible values have to be between 0 and 100
2. Drop the foreign key between student and
    a. It will be possible to assign a tutor to a student which doesn't exist as a teacher?
3. Add to the student table, after the name, an attribute to register the telephone number. It has to be mandatory
4. Alter student table. The dni becomes the primary key instead of registration number. Registration number has to remain unique
5. Now, alter student table to drop the column "fecha de alta"
6. Add in the student table one column to register de date when the student left his studies.
    a. Do you see any relation between this new field and other existing one? Do you think that this change can lead to data inconsistency.
7. We want to add another field in teacher table that is the college name. We will call it "IES". Write the sentences to allow this modification.

a. You cannot drop the tables and create them again

8. Create another table for storing Subject's name and students marks in each subject considering the necessary relationships. One student can have several marks in one subject, only one per day

12 Dropping of tables

Syntax DROP TABLE en MySQL

```
DROP [TEMPORARY] TABLE [IF EXISTS]
    tbl_name [, tbl_name] ...
    [RESTRICT | CASCADE]
```

12.1 Exercise 4

9. Drop teacher table. Have you been able to do it?
10. Drop all the tables from the previous exercise in the correct order
11. If you had to create again the tables, which would be the right order for the sentences.

13 Indexes

Indexes can be created when we create the table. The system will create unique indexes .

- When we establish that a set of attributes is primary key
- When we establish that a set of attributes is Unique.
- …

Indexes can also be created after the table has been created (it can take a long time, depending on how many rows are in the table

13.1 CREATE INDEX

```
CREATE [UNIQUE|FULLTEXT|SPATIAL] INDEX index_name
    [USING index_type]
    ON tbl_name (index_col_name,...)
```

```
index_col_name:
    col_name [(length)] [ASC | DESC]
```

13.2 DROP INDEX

```
DROP INDEX index_name ON tbl_name
```