**Introduction to Database Technology FMA**

Mindaugas Pranaitis

Mprana01

Jerry Smallwood

Submission date: 30/07/2019

Table of Contents

[Introduction 3](#_Toc15373343)

[1.Information Retrieval 3](#_Toc15373344)

[1.A Create a view 4](#_Toc15373345)

[1.B Display children activity count 4](#_Toc15373346)

[1.C Display children with their Carers, activity name and activity count. 5](#_Toc15373347)

[1.D Amount earned by each activity 6](#_Toc15373348)

[2.Referential Integrity 6](#_Toc15373359)

[2.A Childactivity table add and delete record. 7](#_Toc15373360)

[2.B Carer table record update and Deletion 7](#_Toc15373361)

[2.C ON delete set null action 8](#_Toc15373362)

[2.D On Delete Action settings 8](#_Toc15373363)

[3.Database Extension 8](#_Toc15373364)

[3.A ER diagram 9](#_Toc15373365)

[3.B Table Description 11](#_Toc15373366)

[5. References 13](#_Toc15373367)

**The Playscheme Database Design**

## Introduction

SQL (structured query language) is the most powerful query language for relational database systems. It allows us store and manipulate data also retrieve information from database. When we talk about SQL we need to mention DML and DDL languages which are briefly explained bellow.

DML (Data Manipulation Language) – Is a language used to retrieve information from a database using SELECT command, add new information to a database using the INSERT command, update information stored in a database with the UPDATE command, and remove information from a database with the DELETE command. [1]

DDL (Data Definition Language) – is language used to generate a new database table using CREATE TABLE statement, modify the structure of a database table with command ALTER TABLE and delete a database table using DROP TABLE. [[2](https://www.lifewire.com/what-is-sql-1019769)]

Both DML and DDL languages has been used in Play scheme database design project. The purpose of Playscheme database is store data for a school holiday playscheme. My project will cover information retrieval, referential integrity and database extension design topics.

# 1.Information Retrieval

No doubt that information retrieval plays pivotal role in SQL, because to store data without being able to access it won’t make sense in data driver world. Now, as I mentioned above, to retrieve data - DML should be used.

The simplest query that can be used is SELECT \* *FROM table name*, which would display all the records belonging to the chosen table for ex. SELECT \* FROM Child would display data stored in Child table. But what if we want to query multiple tables? Here comes powerful table JOINS[more 3].

In addition to queries mentioned above there are number of built in functions that could be used to perform calculations (aggregate functions), sort data or display a table view. Below are shown queries to get various information from Playscheme database.

### A Create a view

CREATE VIEW

Childlist AS

SELECT

CONCAT(child\_fname, " " ,child\_sname) AS 'Child name',

activity\_name AS 'Activity name',

activity\_day AS 'Activity day'

FROM

Child C,

Activity A,

Childactivity CA

Where C.child\_id = CA.child\_id AND CA.activity\_id = A.activity\_id

ORDER BY

child\_sname,

child\_fname,

activity\_name;

### 1.B Display children activity count

SELECT

CONCAT(C.child\_fname, ' ' ,C.child\_sname) AS 'Child',

COUNT(CA.activity\_id) AS 'Activity count'

FROM Child C

LEFT JOIN Childactivity CA ON C.child\_id = CA.child\_id

GROUP BY

C.child\_id

### 

### 1.C Display children with their Carers, activity name and activity count.

SET @carerfname = 'John';

SET @carersname = 'Little';

SELECT

CONCAT(C.child\_fname, ' ' ,C.child\_sname) AS 'Child',

CONCAT(CR.carer\_fname,' ' , CR.carer\_sname) AS 'Carer',

A.activity\_name AS 'Activity',

A.activity\_day

FROM Child C

JOIN Carer CR ON CR.carer\_id = C.child\_carer

JOIN Childactivity CA ON C.child\_id = CA.child\_id

JOIN Activity A ON A.activity\_id = CA.activity\_id

ORDER BY

C.child\_sname,

C.child\_fname

### 1.D Amount earned by each activity

### SELECT

### COUNT(CA.child\_id) AS 'Child number',

### A.activity\_name,

### ROUND(COUNT(activity\_name) \* A.activity\_fee,2) AS 'Total earned'

### FROM

### Activity A, Childactivity CA

### WHERE

### A.activity\_id = CA.activity\_id

### GROUP BY

### A.activity\_id

# 2.Referential Integrity

Referential integrity is a reliable mechanism that prevents accidental database inconsistencies when you perform inserts, merges, updates, and deletes. Also, it ensures the relationships between tables in a database remain accurate by applying constraints to prevent users or applications from entering inaccurate data or pointing to data that doesn't exist. [4]

MySQL supports five relational integrity actions [5]:

* **RESTRICT**Generates an error and prevents the modification if an attempt to alter a referenced primary key value occurs. This is the default referential integrity action.
* **SET NULL**Sets all foreign keys that reference the modified primary key to NULL.
* **SET DEFAULT**Sets all foreign keys that reference the modified primary key to the default value for that column (as specified in the table definition).
* **CASCADE**When used with ON UPDATE, this action updates all foreign keys that reference the updated primary key to the new value. When used with ON DELETE, this action deletes all rows containing foreign keys that reference the deleted primary key.
* **NO ACTION** Is the same as RETRICT.

In next 4 sections I will talk about when insertions and deletions can occur, use of SET NULL delete action and appropriate ON DELETE settings for all foreign keys in Playscheme database.

### 2.A Childactivity table add and delete record.

First, looking at Figure 1 ER diagram we recognize that the relationship between children and activities is many-to-many: a child may attend one or more activities; an activity may be attended by one or more children.

Second, is important to understand terms: parent table, child table and foreign key before inserting or deleting database table records. So a parent is the table that stores the primary key, a child is any table that references the parent with a foreign key and a foreign key is a field in a table that matches another field of another table. A foreign key places constraints on data in the related tables, which enables referential integrity in database.

Third, if we look at Figure 2 Table description Childactivity table, we see two composite keys: child\_id and activity\_id. Foreign key ca\_child references Child table child\_id and foreign key ca\_activity references Activity Table activity\_id, therefore both Child and Activity tables are parent and Childactivity is a child table. If record needs to be inserted into Childactivity table both parent tables/or referenced tables must have records, otherwise Foreign Key Violation error will appear, meaning you trying to add row in a child table when parent table record doesn’t exist.

Fourth, considering Figure 2, Childactivity table description we can identify NULL values exists for child\_id and activity\_id columns, therefore if want to delete (update rows to NULL values) activity record or child\_id we need to ensure that we allow foreign keys to have NULL values, otherwise error *column can’t be null* will be displayed. However, deletion of child\_id and activity\_id is permitted only if records deleted at the same time.

### 2.B Carer table record update and Deletion

Let’s look at Figure 1. We identify that children are cared by specific Carer and there is one to many relationships between these entities. Carer entity is Parent table and Child entity is Child table (many side relationship). To delete record from Carer table we could set foreign key cascade delete action, which will ensure that once Carer is deleted, associated record is deleted. However, in our case child\_carer (Figure 2 table description Child table) allows NULL values (child can exist without carer) therefore child record will not be destroyed. Record can be successfully inserted in the Carer table as long as all columns that are NOT NULL contains relative data.

In addition to above is important to remember that foreign key and referencing primary key data type must match, otherwise any update or deletion will fail.

### 2.C ON delete set null action

On delete set null action - specifies that the child data is set to NULL when the parent data is deleted. Considering that this action is set for Childactivity foreign keys consequences will be deleted specific Child and Activity records.

Disadvantages of set null action:

* loss of associated data (if parent table record delete all the foreign key rows are deleted).
* Orphaned rows are unlikely [6]
* Query’s perform slowly

Advantages:

* Ensures data integrity – prevents of data being in the table which is no longer existing.

### 2.D On Delete Action settings

To choose appropriate setting for all foreign key in database we need to look at Figure 2. Starting with Child table we have two foreign keys. Since columns they are referencing is set to default null, meaning child can exist without Carer and do not have school, the best option would be on delete no action. The rest foreign/composite keys in Childactivity and Instructoractivity I would set on delete set null, because when parent table record deleted associated records will be deleted to, ensuring data integrity (no orphaned records present).

# 3.Database Extension

My purpose in this section is to extend database so the children can attend play scheme on certain days. With that in mind, I would create Playday entity (red rectangular) see Figure 1.A Extended ER diagram. Relationship between Playday and Child is one to many, therefore foreign key was created in Child table (refer in red) to reference Playday records see Figure 2A. Looking at the same figure we see data tape for for column play\_day is DATE see Playday table because to ensure data integrity we should always store DATEs as DATEs - use the appropriate datatype at all times and then derive/calculate the values we want to display **from** these correct datatypes.

### 3.A ER diagram

Figure 1 ER Diagram

Instructor

Instructoractivity

Activity

Childactivity

Child

School

Carer

*Figure SEQ Figure \\* ARABIC 1ER Diagram*

*Figure SEQ Figure \\* ARABIC 2 ER diagram extension*

Instructor

Activity

Instructoractivity

Childactivity

Child

School

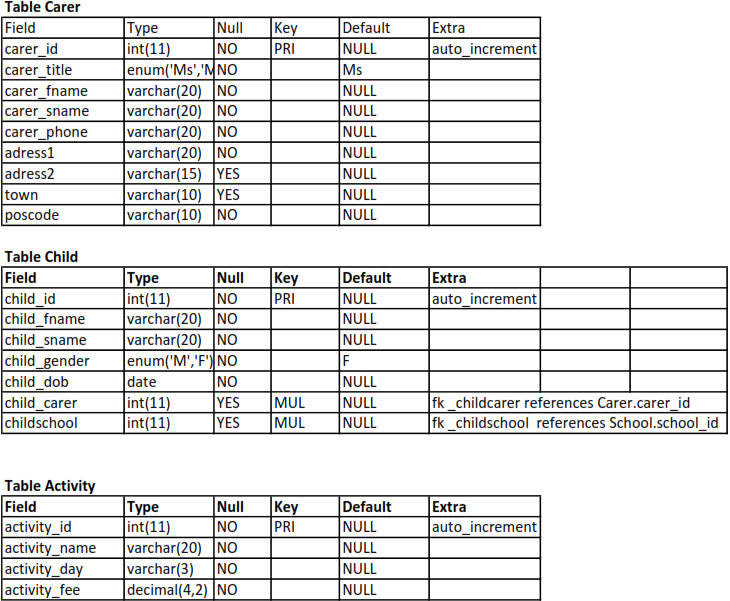
Carer

Playday

Figure 1.A Extended er diagram

### 

### 3.B Table Description



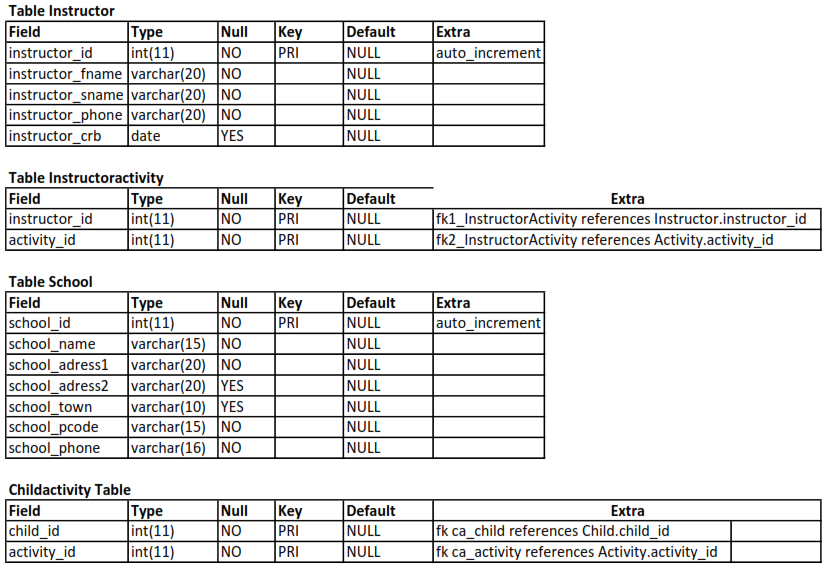
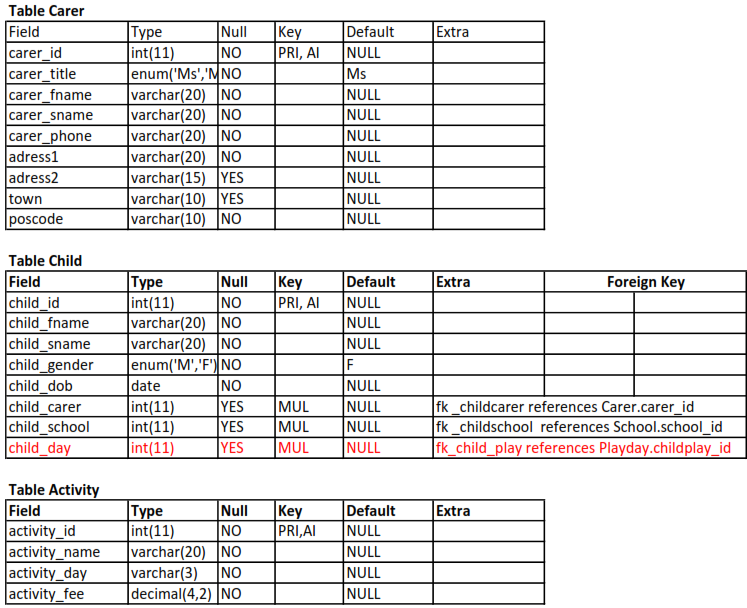


Figure 2 Table description



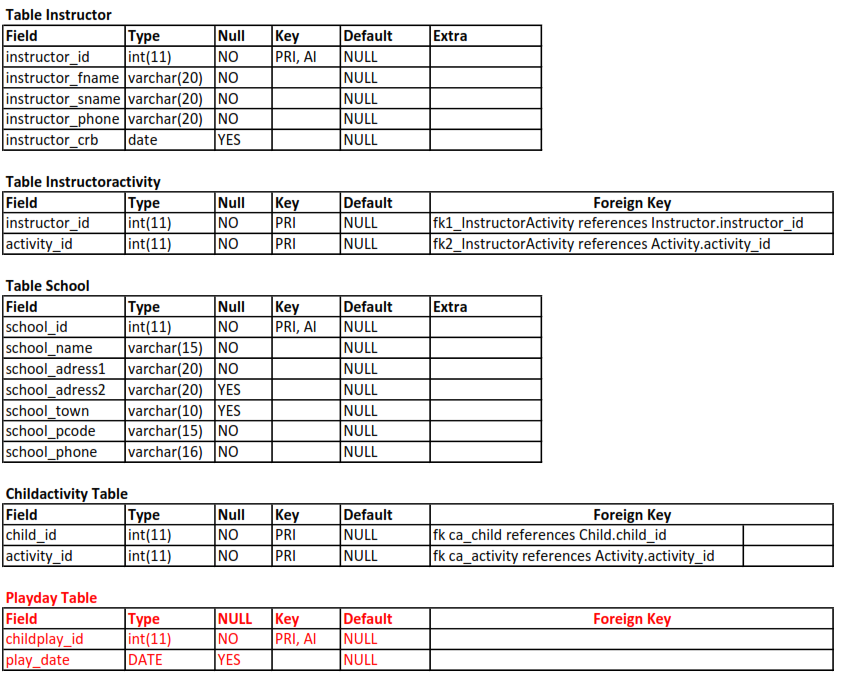


Figure .A Extended Table Description

### 5. References

1. <https://www.lifewire.com/what-is-sql-1019769>
2. <https://www.lifewire.com/what-is-sql-1019769>
3. <https://www.lifewire.com/joining-multiple-tables-sql-inner-join-1019774>
4. <https://www.lifewire.com/referential-integrity-definition-1019181>
5. <https://dev.mysql.com/doc/refman/8.0/en/create-table-foreign-keys.html#foreign-keys-referential-actions>
6. <https://database.guide/what-is-an-orphaned-record/>