# Overview

You are given a simple SQLITE3 database for an attendance system with a small set of data.

An employee can perform one of two actions: CheckIn or CheckOut. When an action is performed, it is stored at 2 levels:

* An attendance entry is created for the day if no actions were previously performed on that day
* An attendance action entry is created recording the time for this specific action and linked to the day’s attendance record

For example, if EMP01 checks in at midnight of 2020-04-01, two records are created:

* A record in the **Attendance** table with date 2020-04-01 and employee **EMP01**
* A record in the **AttendanceActions** table referencing the previous attendance record with time 2020-04-01 12:00 AM

The time stamps are stored in local server time, which is in the Cairo timezone (UTC + 2). This is not a good practice, but it’s based on a real-world case.

# Requirements

## 01 Check day attendance for employee

Write a function that accepts an employee code (e.g. EMP01) and a date (e.g. 2020-04-01) and reports whether the employee has attended that day and how long.

### Examples

get\_attendance(‘emp01’, ‘2020-04-01’) on the sample data should return the following dictionary:

{  
 “attended”: true,  
 “duration”: “12:00”  
}

The employee attended from 12:00 AM to 12:00 PM, which is exactly 12 hours.

For 2020-04-02, the employee attended from 12:05 AM to 11:50 AM (11:45 night shift), then checked in again for his next shift at 11:50 PM (so 10 minutes until midnight the next day) for a total of 11:55:

{  
 “attended”: true,  
 “duration”: “11:55”  
}

For 2020-04-03, the employee checked out at 12:05 PM for a duration of 12:05 (12 hours 5 minutes) calculated from midnight since his last check-in was the previous day.

## 02 Retrieve attendance history for employee

Write a function that accepts an employee code (e.g. EMP01) and returns the attendance history for that employee. An attendance history is a list of days, and within each day, a list of attendance actions.

**This function should return action times in UTC ISO format** so that its clients can translate back from UTC to the local time zone (e.g. it would be exposed as an API that can be used by people from different time zones). Remember that the sample data is in the Cairo time zone (UTC + 2).

The structure of output should look like this (JSON representation of a dict):

{  
 “days”: [  
 “date”: “2020-04-03”  
 “actions”: [  
 { “action”: “CheckOut”, “time”: “2020-04-01T10:05:00.000000+00:00” }  
 ]  
 ]  
}

Note that UTC translation may create day entries for which there’s no **Attendance** record (e.g. the check-in at 2020-04-01 12:00 AM Cairo time is actually 2020-03-31 10:00 PM UTC)