Pràctica 2: Introducció a SQL

Bases de Dades — iTIC

Marta Tarrés-Puertas

22 de març de 2017

Índex

1 Organització

1.1 Lliurament

Cal lliurar la pràctica en un tarfile a través d'Atenea en la data fixada.

2 Tasques a realitzar

Un cop realitzats els previs *Instal·lació entorn SQLite3 i creació de la primera BD*, realitzeu les següents tasques. Per cadascuna de les expressions SQL utilitzades, es demana que adjunteu addicionalment els resultats d'execució d'aquestes.

2.1 El pàrquing

Creeu l'esquema de la Base de Dades que creieu necessària pel gestor de pàrquing, que pugui permetre la gestió dels tiquets de sortida. Inseriu-hi dades, realitzeu consultes (places ocupades, places lliures, cotxe aparcat al pàrquing, cotxes per colors, cotxes per models, cotxes més d'un dia al pàrquing) i permeteu-ne la sortida de vehicles del pàrquing. Es tindran en compte els aspectes següents:

- Diferència entre definir un atribut com primary key, i definir-lo com a primary key not null

- Chequeig nulls i repetits en la definició d'atributs com a primary key, i en la definició de primary key not null
- Gestió de la data del sistema en la entrada de cotxes
- Inserció d'informació en la taula de diferent tipus al definit a l'esquema / relació
- Còmput del cost d'estada
- Gestió de places lliures

A continuació haureu de gestionar taules relacionades, i caldrà que tingueu en compte els aspectes següents:

- Com s'activa la integritat refencial en SQLite?
- Chequegeu-ne el comportament per veure si és l'esperat: afegiu valors pels atributs que no existeixin en el valor de les claus que referencien
- Quin és el mode d'actualització per defecte en cas de borrat/modificació d'atributs amb claus referenciades en SQLite?
- Es pot canviar el mode anterior? Com?

2.2 La mini-xarxa social

Utilitzant l'esquema de BD de la mini-xarxa social que es detalla a continuació,

- usuaris(email,nom,cognom,poblacio,dataNaixement,pwd)
- amistats(email1,email2,estat)

Se us demana:
* Proporcionar el diagrama E/R que correspon a aquest model relacional
* Escriure les expressions SQL per cadascuna de les següents consultes.

1. Obtenir les dades dels usuaris (excepte pwd) que viuen a Manresa
2. Obtenir l'email dels usuaris amb cognom "Albets"
3. Visualitzar els amics (nom i cognom) de l'usuari "Pere", "Garcia"(estat=Acceptada)
4. Obtenir els amics de l'usaris "Pere""Garcia"que no són amics de l'usuari "Jordi""Alba"
5. Obtenir el nombre total de peticions d'amistat rebutjades
6. Obtenir les dades (noms,cognoms) d'amics que viuen a Manresa
7. Obtenir, per cada usuari, el nombre de peticions rebutjades
8. Obtenir els usuaris que no són amics de "Ana", "Vilella"

2.3 Els empleats

Utilitzant l'esquema de BD que es mostra a continuació,

- empleat(id-empleat,carrer,ciutat)
- feina(id-empleat, id-empresa,salari)
- empresa(id-empresa, ciutat)
- manager(id-empleat,id-empleat-coordinador)

Se us demana:
* Proporcionar el diagrama E/R que correspon a aquest model relacional
* Escriure les expressions SQL per cadascuna de les següents consultes.

1. Obtenir els identificadors i ciutat de residència dels empleats que treballen per l'empresa "Bank Newton"
2. Obtenir totes les dades dels empeleats que treballen per "Bank Newtonï guanyen més de 10000
3. Obtenir els identificadors dels treballadors que no treballen a "Bank Newton"
4. Trobar tots els treballadors que guanyen més que cada empleat de "Bank Newton"
5. Troba la empresa que té més empleats
6. Modifica la ciutat de residència de l'empleat 22 a 'Barcelona'
7. Apuja el sou de tots els empleats coordinadors un 10
8. Troba el nom de tots els empleats que viuen a la mateixa ciutat on treballen
9. Troba tots els empleats que viuen a la mateixa ciutat que els seus coordinadors
10. Elimina a 'feina' totes les tuples corresponents a empleats que treballin a "Bank Newton"

2.4 Els sensors

Utilitzant l'esquema de BD proporcionat, *sensors.sql*, llegiu l'enunciat del problema a resoldre i escriviu les expressions SQL que es detallen a continuació. Nota: podeu canviar les dades a inserir en l'esquema.

In this problem set, you will write a series of queries using the SELECT statement over a table called expt table of light and temperature readings collected from three wireless nodes with special sensing hardware. The data represents a day's worth of light and temperature readings, collected every 30 seconds from each sensor node. Because these sensor that were used are lossy, there are some times when sensor readings are missing. The schema of the sensor data is as follows:

SENSORS(**id**, result time, epoch, nodeid, light, temp, voltage)

Where the fields are as follows:

- *result time: timestamp*: The time the record was inserted into the database.

- *epoch: int*: The sensornode sequence number of the record. Different results from different sensors with the same epoch number should arrive in the database at about the same time. If they have different times, it can indicate a failure in the software that runs on the sensor – one of the queries below asks you to look for such failures.
- *nodeid: int*: The id of the sensor node that produced this reading. There are three sensors in this data set, numbered 1, 2, and 3.
- *light: int*: The light reading from the sensor, in raw units relative to the minimum and maximum brightness the sensor can perceive. This is a 10bit sensor, with a value of 0 corresponding to darkness 1024 representing maximum brightness. The calib light table maps raw units into Lux, which is a commonly used measure of brightness.
- *temp: int*: The temperature reading from the sensor, in raw units. The calib temp table maps raw units into degrees Celsius.
- *voltage: int*: The voltage on this device, in raw units.

The two tables, calib light and calib temp each have two fields :

- *raw* : integer
- *calib* : integer

Where *raw* is the raw sensor value and *calib* is the calibrated value, in Lux (in the case of *calib light*) or degrees Celsius (in the case of *calib temp.*)

1. Write a query (using the SELECT statement) that will compute times and ids when any sensor's light reading was above 550. Show both the query and the first few lines of the result.
2. Write a query that will compute the average light reading at sensor 1 between 6 PM and 9 PM (inclusive of 6:00:00 PM and 9:00:00 PM). Show the query and the result.
3. Write a single query that computes the average temperature and light reading at every sensor between 6 PM and 9 PM, but exclude any sensors whose maximum voltage was greater than 418 during that time period. Show both the query and the result.
4. Write a query that computes the average calibrated temperature readings from sensor 2 during each hour, inclusive, between 6 PM and 9 PM (i.e., your answer should consist of 4 rows of calibrated temperatures.)
5. Write a query that computes all the epochs during which the results from sensors 1 and 2 arrived more than 1 second apart. Show the query and the result.
6. Write a query that determines epochs during which one or two of the sensors did not return results. Show your query and the first few results, sorted in by epoch number. You may wish to use a nested query – that is, a SELECT statement within the FROM clause of another SELECT statement.
7. Write a query that produces a temperature reading for each of the three sensors during any epoch in which any sensor produced a reading. If a sensor is missing a value during a given epoch, your result should report the value of this sensor as the most recent previously

reported value. If there is no such value (e.g., the first value for a particular sensor is missing), you should return the special value 'null'. You may wish to read about the CASE and OUTER JOIN SQL statements.

8. Write a query that determines epochs during which all three sensors did not return any results. Note that this is a deceptively hard query to write – you may need to make some assumptions about the frequency of missing epochs.

2.5 SQL hostatjat

Després de realitzar els previs *Pràctica 3: SQLite in Python*, trieu un dels esquemes BD anteriors i dissenyeu i implementeu un script python que gestioni la informació de dit esquema via un menú d'opcions i/o intèrpret.

Heu de permetre gestionar:

- Llistats de la informació de la BD
- Afegir dades
- Eliminar dades
- Modificar dades
- Consultes de la informació via consultes parametritzades
- Lectura de dades proporcionades en fitxer de text i inserció a la base de dades, i el procés invers (volcat de la informació de taula/es a un/s fitxer/s de text)

Consideracions a tenir en compte:

- En el cas de treballar amb més d'una taula, recordeu a activar des de Python la integritat referencial.
- El programa ha de ser robust. Gestioneu els casos d'error possibles via tractament d'excepcions.

3 Bibliografia

- `https://www.sqlite.org/docs.html`
- `http://www.sqlite.org/cli.html`
- Previs Pràctica 2, Material disponible a a Atenea, *Pràctica 2: Instal·lació de SQLite3 i creació de la primera BD* i *Pràctica 2: SQLite in Python*.