Subject: Doubts on a sqlite query
Posted by forlano on Sat, 29 Dec 2012 07:53:47 GMT
View Forum Message <> Reply to Message

Hello,

I have two tables, TEAMS with 120 records and PLAYERS with 600 records. Each team contains 5 players. After a sort I need to update a field of TEAMS and another of PLAYERS. Both fields are called N. To perform this task I use the following procedure

```
void vegadb::SetPairingNumbers(int n, int arr_N[], Vector<int> ids)
{  int i, id, idp, np=1;
 Sql sqlteam(sqliteVT);
 Sql sqlplayer(sqliteVT);
 Sql sqlp(sqliteVT);

  for(i=0; i<n; i++) {
      id = arr_N[i];
     sqlteam.Execute("update TEAMS set N=? where ID=?", i+1, ids[i]);
           sqlplayer.Execute("SELECT ID FROM PLAYERS WHERE TEAM_ID=? ORDER BY
BOARD ASC", id); //retrieve players of team id by boards
     while (sqlplayer.Fetch()) {
   idp = (int) (sqlplayer[0]);
         sqlp.Execute("update PLAYERS set N=? where ID=?", np++, idp); //set pairing number to
the player idp
     }
 }
}
```

It works as expected. The problem is that it requires about 25 seconds on my notebook (intel celeron 1.70 GHz, 1Gb ram). It is not a monster machine but the number of records to be processed is not huge. Because I have not experience with sqlite I would like to know if this delay time is normal or maybe there is a faster way to write the query.

Thanks,
Luigi

Subject: Re: Doubts on a sqlite query
Posted by dolik.rce on Sat, 29 Dec 2012 09:35:06 GMT
View Forum Message <> Reply to Message

Hi Luigi,

The 25 seconds for few simple queries sounds awfully wrong. Have you tried to figure out what exactly takes so long there? Putting TIMING macros to each of the Execute() calls could tell you something:  for(i=0; i<n; i++) {
      id = arr_N[i];

```
    { TIMING("team-update"); sqlteam.Execute("update TEAMS set N=? where ID=?", i+1, ids[i]); }
          { TIMING("player-select"); sqlplayer.Execute("SELECT ID FROM PLAYERS WHERE
TEAM_ID=? ORDER BY BOARD ASC", id); }//retrieve players of team id by boards
    while (sqlplayer.Fetch()) {
  idp = (int) (sqlplayer[0]);
        TIMING("player-update"); sqlp.Execute("update PLAYERS set N=? where ID=?", np++,
idp); //set pairing number to the player idp
    }
 }
```

One trick that might make it faster is to execute it all in single query, using a long case construct. The query would look something like this:update PLAYERS
set N=case ID
  when 1 then 23
  when 2 then 42
  ...
  when 600 then 123
It'll be a long (and ugly  ) query, but it could be faster then 600 small ones. Similar thing can be done for teams too.

Also, are you using indexes on your tables? They can make a lot of difference.

Best regards,
Honza

---

Subject: Re: Doubts on a sqlite query
Posted by forlano on Sat, 29 Dec 2012 12:21:37 GMT
View Forum Message <> Reply to Message

Hi Honza,

thank you very much for the anwer. These were the guilty tables

```
TABLE_(TEAMS)
 INT     (ID) PRIMARY_KEY AUTO_INCREMENT
 INT_    (N)
 STRING_ (NAMET, 30)
 STRING_ (ORIG, 30)
 STRING_ (REGION, 30)
 STRING_ (FED, 3)
 INT_    (RTG)
 INT_    (STATUS)
END_TABLE

TABLE_(PLAYERS)
 INT     (ID) PRIMARY_KEY AUTO_INCREMENT
```

```
 INT_    (TEAM_ID) INDEX
 INT_    (BOARD)
 INT     (N)
 STRING  (NAME, 30)
 STRING  (FED, 30)
 STRING_ (BDAY, 10)
 STRING_ (SEX, 1)
 STRING_ (CAT, 5)
 INT_    (IDFIDE)
 INT_    (RTGFIDE)
 STRING_ (IDNAT,10)
 INT_    (RTGNAT)
 INT_    (K)
END_TABLE
```

I have now added an index to TEAM_ID. I do not know if the index is created while the database is populated or even later. I hope it has been created with this code when I opened the existing DB

```
void vegadb::SetDatabase(String dbname)
{
    if (sqliteVT.IsOpen()) sqliteVT.Close();
    if (!FileExists(dbname)) {
  if(!sqliteVT.Open( dbname )) {
   Exclamation("Can't create or open database file");
   return;
  }
  SQL = sqliteVT;
//  sqliteVT.SetTrace();
  SqlSchema sch(SQLITE3);
  All_Tables(sch);
  Sqlite3PerformScript(sch.Upgrade());
  Sqlite3PerformScript(sch.Attributes());
 }
 else {
  if(!sqliteVT.Open( dbname )) {
   Exclamation("Can't create or open database file");
   return;
  }
  SQL = sqliteVT;
//  sqliteVT.SetTrace();
  SqlSchema sch(SQLITE3);
 }
 SQL.ClearError();
}
```

Anyway I have not seen any improvement.
I used TIMING

TIMING players-update : 44.66 s  - 57.33 ms (44.66 s  / 779 ), min: 32.00 ms, max: 698.00 ms, nesting: 1 - 779
TIMING player-select  : 44.78 s  - 285.22 ms (44.78 s  / 157 ), min: 134.00 ms, max: 844.00 ms, nesting: 1 - 157
TIMING team-update    : 53.32 s  - 339.64 ms (53.32 s  / 157 ), min: 167.00 ms, max: 910.00 ms, nesting: 1 - 157

to discover I have understimated the delay  . The teams are 157 and players 779 but it looks wrong too. The players-update query seems to have problems. I'll try to understand what can be although at moment I do not see error.

Regards,
Luigi

---

Subject: Re: Doubts on a sqlite query
Posted by dolik.rce on Sat, 29 Dec 2012 14:03:40 GMT
View Forum Message <> Reply to Message

forlano wrote on Sat, 29 December 2012 13:21I have now added an index to TEAM_ID. I do not know if the index is created while the database is populated or even later. I hope it has been created with this code when I opened the existing DBI believe the sch.Upgrade() should create the index just fine. The index is normally created when it is added to the table and it is updated with each change to the contents of the table.

forlano wrote on Sat, 29 December 2012 13:21The players-update query seems to have problems. I'll try to understand what can be although at moment I do not see error.It actually seems to me that players-update is the best performing one out of those three  It takes only 57 ms per call. Could you try to do it in fewer queries using CASE as I described above? The code should look something like this (not tested): String teams = "update TEAMS set N = case ID";
```
 for(i=0; i<n; i++) {
  teams += Format(" when %i then %i", ids[i], i+1);
  {TIMING("player-select"); sqlplayer.Execute("SELECT ID FROM PLAYERS WHERE
TEAM_ID=? ORDER BY BOARD ASC", arr_N[i]);}
  String players = "update PLAYERS set N = case ID";
  while (sqlplayer.Fetch()) {
   idp = (int) (sqlplayer[0]);
   players += Format(" when %i then %i", idp, np++);
  }
  {TIMING("players-update"); sqlp.Execute(players); }
 }
```

{TIMING("teams"); sqlteam.Execute(teams);}
It should lower the query count to 157+157+1. I'm really curious if it cuts down the overall time significantly or not

Honza

---

Subject: Re: Doubts on a sqlite query
Posted by forlano on Sat, 29 Dec 2012 14:35:14 GMT
View Forum Message <> Reply to Message

Quote:It actually seems to me that players-update is the best performing one out of those three

This shows how deep is my understanding of the subject

Quote:It should lower the query count to 157+157+1. I'm really curious if it cuts down the overall time significantly or not

You are welcome. I've just copied and paste to get

TIMING teams : 0.00 ns - 0.00 ns ( 0.00 ns / 1 ), min: 0.00 ns, max: 0.00 ns, nesting: 1 - 1
TIMING players-update : 9.97 ms - 63.51 us (10.00 ms / 157 ), min: 0.00 ns, max: 1.00 ms, nesting: 1 - 157
TIMING player-select : 29.97 ms - 190.90 us (30.00 ms / 157 ), min: 0.00 ns, max: 1.00 ms, nesting: 1 - 157

Now it takes an instant instead of 46 seconds
I discovered there is another bottleneck in my code. I am going to remove with your excellent tecnique!

Thanks a lot and Happy New Year!!!
Luigi

Edit: just now I checked better and I discovered that the query do nothing ... there should be some syntax error in the query somewhere... investigating

---

Subject: Re: Doubts on a sqlite query
Posted by dolik.rce on Sat, 29 Dec 2012 16:45:45 GMT
View Forum Message <> Reply to Message

forlano wrote on Sat, 29 December 2012 15:35Edit: just now I checked better and I discovered that the query do nothing ... there should be some syntax error in the query somewhere... investigating
Exactly what I thought when I saw that the times are so much faster now, "That looks to good to be true" I probably made som mistake in the code or there is something I missed about sqlite syntax, I'm no expert either

You should probably check the queries. Some time ago in another thread, I posted a very simple sqlite console written in U++. You can use that or some other tools (e.g. sqlitebrowser or sqliteman) to inspect the contents of the database and to easily test the queries.

Honza

---

Subject: Re: Doubts on a sqlite query
Posted by forlano on Sat, 29 Dec 2012 20:27:47 GMT
View Forum Message <> Reply to Message

Good news. This works as expected (I got the hint here )

```
 String teams = "update TEAMS set N = case ";
 for(i=0; i<n; i++) {
  teams += Format(" when ID=%i then %i", ids[i], i+1 );
  {TIMING("player-select"); sqlplayer.Execute("SELECT ID FROM PLAYERS WHERE
TEAM_ID=? ORDER BY BOARD ASC", arr_N[i]); }
  String players = "update PLAYERS set N = case ";
  while (sqlplayer.Fetch()) {
   idp = (int) (sqlplayer[0]);
   players += Format(" when ID=%i then %i", idp, np++);
  }
  players += " else N end ";
  {TIMING("players-update"); sqlp.Execute(players); }
 }
 teams += " else N end ";
 {TIMING("teams");sqlteam.Execute(teams);}
```

with timing

```
TIMING teams          : 56.00 ms - 56.00 ms (56.00 ms / 1 ), min: 56.00 ms, max: 56.00 ms,
nesting: 1 - 1
TIMING players-update : 21.90 s  - 139.46 ms (21.90 s  / 157 ), min: 46.00 ms, max: 646.00 ms,
nesting: 1 - 157
TIMING player-select  : 177.97 ms -  1.13 ms (178.00 ms / 157 ), min:  0.00 ns, max:  2.00 ms,
nesting: 1 - 157
```

Much better but still too high. Anyway I saw the things can improve rearranging the query. I'll think about it.

Thanks again,
Luigi

---

Subject: Re: Doubts on a sqlite query
Posted by lectus on Sat, 29 Dec 2012 20:57:57 GMT
View Forum Message <> Reply to Message

Wow! I learn a lot in this forum.

I didn't know about TIMING macro. Very useful.

Also I didn't know about UPDATE using CASE.

Subject: Re: Doubts on a sqlite query
Posted by dolik.rce on Sat, 29 Dec 2012 21:40:18 GMT
View Forum Message <> Reply to Message

forlano wrote on Sat, 29 December 2012 21:27TIMING teams : 56.00 ms - 56.00 ms (56.00 ms / 1 ), min: 56.00 ms, max: 56.00 ms, nesting: 1 - 1
TIMING players-update : 21.90 s - 139.46 ms (21.90 s / 157 ), min: 46.00 ms, max: 646.00 ms, nesting: 1 - 157
TIMING player-select : 177.97 ms - 1.13 ms (178.00 ms / 157 ), min: 0.00 ns, max: 2.00 ms, nesting: 1 - 157

Much better but still too high. Anyway I saw the things can improve rearranging the query. I'll think about it.
Thats great, so the last problematic thing is the players update... Let's take it a step further and try to set all the players at once. This should work (unless I made some stupid mistake again ): String teams = "update TEAMS set N = case ";
String players = "update PLAYERS set N = case ";
for(i=0; i<n; i++) {
teams += Format(" when ID=%i then %i", ids[i], i+1 );
{TIMING("player-select"); sqlplayer.Execute("SELECT ID FROM PLAYERS WHERE TEAM_ID=? ORDER BY BOARD ASC", arr_N[i]); }
while (sqlplayer.Fetch()) {
idp = (int) (sqlplayer[0]);
players += Format(" when ID=%i then %i", idp, np++);
}
}
players += " else N end ";
{TIMING("players-update"); sqlp.Execute(players); }
teams += " else N end ";
{TIMING("teams");sqlteam.Execute(teams);}

That should bring it to usable speeds, my guess is under half a second

Honza

Subject: Re: Doubts on a sqlite query
Posted by forlano on Sun, 30 Dec 2012 03:58:48 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Sat, 29 December 2012 22:40
That should bring it to usable speeds, my guess is under half a second

Honza

Thats is two times great!

TIMING teams : 55.00 ms - 55.00 ms (55.00 ms / 1 ), min: 55.00 ms, max: 55.00 ms, nesting: 1 - 1
TIMING players-update : 168.00 ms - 168.00 ms (168.00 ms / 1 ), min: 168.00 ms, max: 168.00 ms, nesting: 1 - 1
TIMING player-select : 123.98 ms - 789.65 us (124.00 ms / 157 ), min: 0.00 ns, max: 2.00 ms, nesting: 1 - 157

Two steps and each saved 25 seconds. Not bad!
Now I can devote my attention to the other horrible queries that still pollute my code.

Many, many thanks!
Luigi

Subject: Re: Doubts on a sqlite query
Posted by forlano on Sun, 30 Dec 2012 08:08:29 GMT
View Forum Message <> Reply to Message

Here is another achievement. The Honza's tecnique applied to insert. It is not trivial as the sqlite syntax is different by that of mysql (this costed me a hour of time to figure out the problem). See here for details :

Original query (26 seconds)

```
  for (i=0; i<pairs.GetCount(); i++) {
     for (int j=0; j<TD.MaxBoards+2; j++) {t1[j]=0; t2[j]=0;}
     idw = pairs[i].idw;
     idb = pairs[i].idb;
     played =1; //regular game (default)
     if (idb==0 || idw==0) played = 0; //byed game
   {RTIMING("save-team"); SQL.Execute("INSERT INTO TEAMROUND
(ROUND,BOARD,NW,NB, PLAYED) VALUES(?,?,?,?, ?)", round, i+1, idw, idb, played); }
     Team& team1 = VTeam.Get( idw );
     Team& team2 = VTeam.Get( idb );
     if (idw) team1.GetPlayerTeam(t1); //get the white players
     if (idb) team2.GetPlayerTeam(t2); //get the white players
```

```
      int idTEAMROUND = SQL.GetInsertedId();
      //save the player pairing
      for (b=1; b<=TD.NBoards; b++) {
          gameplayed =1; //regular game (default)
          if (t2[b]==0 || t1[b]==0) gameplayed = 0; //byed game
       {RTIMING("save-player"); SQL.Execute("INSERT INTO PLAYERROUND (TEAMROUND_ID,
ROUND,BOARD,NW,NB,PLAYED) VALUES(?,?,?,?,?, ?)",
           idTEAMROUND, round, b, t1[b], t2[b], gameplayed); }
      }
  }
```

New version (5 seconds)

```
  String teamround = "INSERT INTO PLAYERROUND (TEAMROUND_ID,
ROUND,BOARD,NW,NB,PLAYED) select 0 as TEAMROUND_ID, 0 as ROUND, 0 as BOARD, 0
as NW, 0 as NB,0 as PLAYED ";
  int npair = pairs.GetCount();
  for (i=0; i<pairs.GetCount(); i++) {
      for (int j=0; j<TD.MaxBoards+2; j++) {t1[j]=0; t2[j]=0;}
      idw = pairs[i].idw;
      idb = pairs[i].idb;
      played =1; //regular game (default)
      if (idb==0 || idw==0) played = 0; //byed game
   SQL.Execute("INSERT INTO TEAMROUND (ROUND,BOARD,NW,NB, PLAYED)
VALUES(?,?,?,?, ?)", round, i+1, idw, idb, played);
      Team& team1 = VTeam.Get( idw );
      Team& team2 = VTeam.Get( idb );
      if (idw) team1.GetPlayerTeam(t1); //get the white players
      if (idb) team2.GetPlayerTeam(t2); //get the white players
      int idTEAMROUND = SQL.GetInsertedId();
      //save the player pairing
      for (b=1; b<=TD.NBoards; b++) {
          gameplayed =1; //regular game (default)
          if (t2[b]==0 || t1[b]==0) gameplayed = 0; //byed game
   teamround += Format("UNION SELECT %d,%d,%d,%d,%d,%d ", idTEAMROUND, round, b,
t1[b], t2[b], gameplayed);
      }
  }
  SQL.Execute(teamround);
```

Perhaps the previous ugly query stored in teamround string can be simplified with the U++ syntax... if one knows how to do.

Regards,
Luigi

Subject: Re: Doubts on a sqlite query
Posted by dolik.rce on Sun, 30 Dec 2012 08:44:53 GMT
View Forum Message <> Reply to Message

Interesting, I didn't know that you have to do such tricks for multirow inserts in sqlite

For the last piece of code you posted: You probably want the inserting into TEAMROUND to be done in single query too. You now know how, so it should be easy. Also about the "U++ syntax" for sql: It looks better, it takes care of type control, escaping etc. so it is safer, but sometimes it is a pain to do complicated queries in it... I think for example multirow insert would be quite hard to achieve using SqlInsert.

Anyway, the moral of this story is: Sqlite is fast, but only for read queries (selects). The write queries (insert, update) have a big overhead, so many small queries take a long time. Therefore, concatenating them into single long query can save orders of magnitude of time.

This is to some degree true for any database, e.g. for MySql you want to minimize the number of queries too, because connecting to the server can be costly operation. But I admit I didn't know that this applies to sqlite and that it is so bad that couple queries can render the application unusable

Honza

Subject: Re: Doubts on a sqlite query
Posted by forlano on Sun, 30 Dec 2012 12:16:59 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Sun, 30 December 2012 09:44
For the last piece of code you posted: You probably want the inserting into TEAMROUND to be done in single query too. You now know how, so it should be easy.

I tried but I am afraid this time it can't be done. In fact I am iserting in two tables (TEAMROUND and PLAYERROUND) and in the second one I need the 'id' of the record just inserted in the first table that is unknown before of query (or maybe yes ). So, after the query I get it by
int idTEAMROUND = SQL.GetInsertedId();
and then use it later in the final query.

Regards,
Luigi

Subject: Re: Doubts on a sqlite query
Posted by dolik.rce on Sun, 30 Dec 2012 13:22:14 GMT
View Forum Message <> Reply to Message

Oups, I overlooked that

You could in theory get around that limitation too, but it depends on the exact logic of your application, which I don't know  The basic idea would be to use some other unique combination of columns (ROUND + BOARD perhaps?) to find the correct row instead of the autoincremented ID. If there is such unique combination of columns, you could then construct the query all at once and don't care about the IDs at all.

Honza

---

Subject: Re: Doubts on a sqlite query
Posted by forlano on Sun, 30 Dec 2012 14:20:00 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Sun, 30 December 2012 14:22Oups, I overlooked that

You could in theory get around that limitation too, but it depends on the exact logic of your application, which I don't know  The basic idea would be to use some other unique combination of columns (ROUND + BOARD perhaps?) to find the correct row instead of the autoincremented ID. If there is such unique combination of columns, you could then construct the query all at once and don't care about the IDs at all.

Honza

That is a very good idea! I must study carefull if it backfires (the board can be changed...).

Luigi

---