octomon - Segnalazione #7

Database MySQL nuovo Octomon e integrazione con il vecchio...

08/28/2013 06:15 PM - Mark Caglienzi

Status:	Chiuso	**Start date:**	08/28/2013
Priority:	Normale	**Due date:**	
Assignee:	Christopher R. Gabriel	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			
Resolution:			

Description

Dunque, premetto che non sono espertissimo di database.
La situazione attuale è questa:
1. Il vecchio DB usa MyISAM come storage engine (che pare fosse il default anni fa)
2. Django di default usa InnoDB, fra poche righe sarà chiaro il motivo (in ogni caso si può dirgli tranquillamente di usare MyISAM, è una riga di settings.py)

Differenze fra i due storage engine? Questo è quello che ho dedotto leggendo la documentazione di MySQL (un sysadmin o un dbadmin potrebbero bruciarmi con lo sguardo..):

MyISAM
https://dev.mysql.com/doc/refman/5.7/en/myisam-storage-engine.html
Per farla breve, non supporta le transazioni né i vincoli di ForeignKey

InnoDB:
https://dev.mysql.com/doc/refman/5.7/en/innodb-storage-engine.html
Supporta sia le transazioni, che i vincoli di ForeignKey.

A quando pare le FK con le tabelle MyISAM sono semplicemente dei campi int, e MySQL non si occupa del controllo del vincolo dato che non c'è supporto a livello di storage engine.

La documentazione di MySQL dice che è possibile migrare da MyISAM a InnoDB, ma si va un po' troppo sul tecnico per le mie conoscenze: https://dev.mysql.com/doc/refman/5.7/en/converting-tables-to-innodb.html

Probabilmente la mossa più corretta sarebbe quella di migrare il vecchio db a InnoDB e lasciare Django col default.
Nell'ambiente di test (MySQL in locale con il dump del db di produzione) ho avuto problemi col primo esperimento perché i vincoli intertabella non funzionano se gli storage engine sono diversi (pagina di errore di Django).

Però mi viene qualche dubbio, in primis questo:
Dato che MyISAM è "di manica larga", a differenza di InnoDB che "fa il database anziché fare finta", cosa succede durante la migrazione in caso di vincoli non rispettati nelle tabelle MyISAM?

History

#1 - 08/29/2013 11:26 AM - Mark Caglienzi

I test proseguono, situazione al commit 3ead8b7e:

- Modificato settings.py.devel con la rimozione di SQLite, MySQL è stato rinominato in 'default'.
- Rimossi i file db_routers.py e i relativi settaggi in settings.py.devel (dato che ora il database è unico).
- Aggiunto un modello SchoolsManagers agganciato alla tabella school_tg_user, settato come through per il m2m School.managed.

Workflow:

- Porting di tutte le tabelle del db MysQL a InnoDB: 'ALTER TABLE nome_tabella ENGINE=InnoDB;' (per la tabella components il mio PC (Core i7 + 8GB di RAM) ha impiegato più di 2 ore, le altre sono robe di una manciata di secondi).
- ./manage.py syncdb && ./manage.py update_stats && ./manage.py create_test_users (così vengono create le tabelle di Django in MySQL, poi si popolano le tabelle delle statistiche, e vengono creati una trentina di utenti 'pippo##' per averli disponibili durante i test).

Riguardo alla storia InnoDB/MyISAM tutte le tabelle in locale sono state trasformate senza nessun errore, vedremo in futuro se usciranno problemi.

#2 - 08/29/2013 07:27 PM - Mark Caglienzi

Situazione al commit ec39a0a4:

- Modificata la tabella ticket perché fosse coerente con Django: ALTER TABLE `octomon`.`ticket` CHANGE COLUMN `closed_by` `closed_by_id` INT(11) NULL DEFAULT NULL;

Dato che Django quando crea una ForeignKey aggiunge _id al nome del campo, e dato che Ticket non sembra attualmente usato in produzione, mi è sembrato più pulito agire a livello di MySQL piuttosto che evitare che Django aggiungesse _id (e fra l'altro la colonna relativa a chi ha aperto il ticket è correttamente chiamata opened_by_id).

- Modificati i modelli, le view e i template dappertutto in modo che supportassero correttamente il modello User di Django.
- Aggiunti alcuni controlli sull'utente a livello di view e template [ma di questo ne parlo nel ticket #8, per mantenere le cose divise]

#3 - 08/30/2013 11:18 AM - Mark Caglienzi

Da ulteriore ispezione è uscito che il porting a InnoDB, come era lecito aspettarsi, non ha aggiunto i vincoli di chiave esterna, dato che MyISAM non li supporta e MySQL non li può conoscere indipendentemente da Django. E ovviamente ./manage.py syncdb non aggiunge i vincoli, dato che di default non impartisce query ALTER TABLE.

A conferma di questo prendo ad esempio la tabella school:

```
mysql> show create table school;
school | CREATE TABLE `school` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(200) DEFAULT NULL,
  `area_id` int(11) DEFAULT NULL,
  `address` varchar(300) DEFAULT NULL,
  `phone_number` varchar(200) DEFAULT NULL,
  `email_address` varchar(200) DEFAULT NULL,
  `lat` double DEFAULT NULL,
  `lon` double DEFAULT NULL,
  `last_timestamp` double DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `school_area_id_exists` (`area_id`)
) ENGINE=InnoDB AUTO_INCREMENT=81 DEFAULT CHARSET=utf8
```

Mentre manage.py sql specifica anche l'ALTER TABLE per la chiave da School ad Area:

```
$ ./manage.py sql schools
BEGIN;
CREATE TABLE `school` (
    `id` integer AUTO_INCREMENT NOT NULL PRIMARY KEY,
    `name` varchar(200) NOT NULL,
    `area_id` integer,
    `address` varchar(300) NOT NULL,
    `phone_number` varchar(200) NOT NULL,
    `email_address` varchar(200) NOT NULL,
    `lat` double precision,
    `lon` double precision,
    `last_timestamp` double precision
)
;
ALTER TABLE `school` ADD CONSTRAINT `area_id_refs_id_73fe4920` FOREIGN KEY (`area_id`) REFERENCES `area` (`id`
);
COMMIT;
```

Domanda: fare la migrazione a InnoDB senza forzare i vincoli di chiave esterna forse ha poco senso. Però è proprio qui che possono uscire i problemi secondo me, dato che fare tutti gli ALTER TABLE per aggiungere i vincoli a un database che gira da anni senza averli, probabilmente può portare a galla situazioni di chiavi non correttamente settate (è un po' come buttare la polvere sotto il tappeto per anni, e poi un bel giorno alzare il tappeto). Ti sei già trovato in situazioni di porting/migrazione simili?

#4 - 09/04/2013 04:39 PM - Christopher R. Gabriel

- Status changed from Nuovo to Chiuso

Chiudo, abbiamo risolto e attivita' completata.