Copiosis DB Setup & Examples

This is the documentation for creating the Copiosis database in SQL Server Management Studio. Additionally, there is some optional instructions for manually adding some data into various tables. In this document you will also find example queries and explanations about certain database fields.

DB Setup:

1. Open Management Studio
2. Right click on the ”Databases” text in the left-hand pane. Choose “New Database…”
3. Name your database “Copiosis” and leave all the other options the same.
4. Click File > Open > File
5. Navigate to the Copiosis.data/Scripts/Copiosis-Setup.sql file
6. After opening, make sure the select box to the left of the “Execute” button has Copiosis selected. (connect to your instance if prompted)
7. Click Execute.

Add some data (optional):

1. In the left-hand pane, expand the Copiosis DB, and then expand the “Tables” section.
2. Right click on the Location table, and select the “Edit Top 200 Rows” option.
3. Enter data to your heart’s content. Due to FOREIGN KEY dependencies, you’ll probably have to enter data in the following order:
    a. Location
    b. User
    c. Item Class
    d. Product
    e. Transaction

Example Queries:

Get all the transactions a user has been involved in and order them by status (Completed, Pending, Rejected).
Warning: Pending YOUR input vs. Pending OTHERS input would not necessarily be in order.

```
SELECT transactionID, providerID, prov.firstName as provFirstName,
prov.lastName as provLastName, receiverID, rec.firstName as
recFirstName, rec.lastName as recLastName, [transaction].productID,
prod.name, [transaction].productDesc, [transaction].status,
dateAdded, createdBy, dateClosed
FROM [transaction]
JOIN [user] AS prov ON prov.userID = providerID
JOIN [user] AS rec ON rec.userID = receiverID
JOIN product as prod ON prod.productID = [transaction].productID
WHERE (providerID = *USERID* OR receiverID = *USERID*)
ORDER BY [transaction].status;
```

OR
Completed Transactions:

```
WHERE (providerID = *USERID* OR receiverID = *USERID*) AND
[transaction].status = 'Completed' OR [transaction].status =
'Rejected';
```

Pending OTHERS input:

```
WHERE (providerID = *USERID* OR receiverID = *USERID*) AND
[transaction].status = 'Pending' AND createdBy = *USERID*;
```

Pending YOUR input:

```
WHERE (providerID = *USERID* OR receiverID = *USERID*) AND
[transaction].status = 'Pending' AND createdBy != *USERID*;
```

Get a user's NBR:

```
SELECT nbr FROM [user] WHERE userID = *USERID*
```

View Specific Transaction:

```
SELECT * FROM [transaction] WHERE transactionID = *TRANSID*
```

Select active users in the same neighborhood experiment as the end-user:

```
SELECT * FROM [user] WHERE status = 1 AND locationID =
*END_USER_LOCATION_ID*
```

Notable DB Fields:

Location Table

neighborhood	Ex: “Kenton”
signupKey	Ex: “kentonSecretKey”

User Table

status	1 = active, 0 = inactive
locationID	Will be assigned at creation based on the KEY they enter when signing up. Mostly unused for our project, but is a way for Copiosis to support multiple inter-connected neighborhoods in the future.

Product Table

deletedDate	We can use this to generate a list of products that were active at the time a transaction took place, even if the product was deleted in the time between when the transaction took place and the info was entered into Copiosis.

Transaction Table

productDesc	We are storing the product description here in the transaction log, even though we have a reference to the productID with its description in the product table, because there are opportunities for fraud to happen otherwise. We want to know what the product description was at the time the transaction was confirmed.
status	Completed, Pending, Rejected
dateAdded	Can be used to “highlight” new activity since a user’s last login.
createdBy	Tells us who added this transaction to Copiosis (the provider or the receiver). We can then use this info to know who the Pending transaction is waiting for input from.
dateClosed	The date when everything is confirmed or rejected by both parties.