Kapitel 5
Die Facebook Query Language (FQL)

In diesem Kapitel erhalten Sie einen Überblick über die Abfragesprache Facebook Query Language (FQL) und die verfügbaren FQL-Tabellen.

Die Facebook Query Language ist eine an die weithin bekannte Datenbank-Abfragesprache SQL (Structured Query Language) angelehnte Schnittstelle, die ähnlich wie die Graph API Zugriff auf die Objekte des sozialen Graphen ermöglicht. Als Vorgänger der Graph API sind Zugriffe mittels FQL heute nur mehr in besonderen Fällen notwendig und empfehlenswert.

Die *Facebook Query Language* (FQL) wurde von Facebook seit dem Start der Anwendungsplattform im Jahr 2007 als alternative Möglichkeit zum direkten Zugriff auf Objektdaten der Plattform positioniert. Bevor 2010 die Graph API umfassenden Zugriff auf die Objekte des sozialen Graphen erlaubte, mussten Entwickler häufig zwischen der damaligen HTTP-REST-API und FQL wechseln, um die gewünschten Daten abfragen zu können. Beide Technologien erlaubten leider jeweils nur Zugriff auf einen Teil der Objektdaten.

Mit der Graph API hat Facebook große Anstrengungen unternommen, die Abbildung des sozialen Graphen in einer einzigen API zu vereinheitlichen. Heute können *fast alle Anwendungsfälle* mit der Graph API abgebildet werden, lediglich einige wenige Spezialfälle erfordern noch den Einsatz von FQL.

FQL orientiert sich an der bekannten Datenbank-Abfragesprache SQL (*Structured Query Language*) und folgt demselben Aufbau wie diese gebräuchliche Sprache:

```
SELECT Feldname,... FROM Tabellenname WHERE Bedingung,...
```

FQL-Abfragen sehen SQL-Abfragen zwar ähnlich, sind jedoch in vielen Punkten wesentlich unflexibler als herkömmliche Datenbank-Queries:

- Die `WHERE`-Klausel darf nur Bedingungen für Felder enthalten, die mindestens ein von Facebook explizit indiziertes Feld besitzen.
- Die `FROM`-Klausel einer FQL-Abfrage kann immer nur eine Tabelle enthalten – Verknüpfungen mehrerer Tabellen mittels Inner oder Outer Join sind grundsätzlich nicht vorgesehen.

- Verschachtelte Abfragen (*Sub Selects*) sind in der `WHERE`-Klausel prinzipiell mit dem Schlüsselwort `IN` möglich, können aber, anders als bei SQL, nicht Variablen der äußeren Abfrage referenzieren.
- FQL kennt keine Möglichkeit zur Gruppierung von Ergebnissen ähnlich der `GROUPBY`-Klausel von SQL.
- Mathematische und logische Operationen sind in FQL grundsätzlich vorhanden, in der offiziellen Dokumentation leider aber nur unzureichend beschrieben.
- FQL unterstützt die Klauseln `ORDERBY` zum Sortieren und `LIMIT` zum Festlegen der Anzahl der Ergebniszeilen.
- FQL ermöglicht *ausschließlich Lesezugriffe*: `INSERT`-, `UPDATE`- oder `DELETE`-Befehle sind nicht mittels FQL möglich und können nur wie gewohnt über die Graph API durchgeführt werden.
- Ähnlich der Adressierung von Objekten in der Graph API dient das Schlüsselwort `me()` in FQL dazu, den aktuellen Benutzer anhand des verwendeten Access Tokens zu identifizieren.
- Die Verwendung eindeutiger Benutzer- oder Seitennamen wird – anders als beim Adressieren von Objekten in der Graph API – nicht von FQL unterstützt.

5.1 FQL-Zugriffe über die Graph API

Mit der angekündigten Einstellung der HTTP-REST-API müssen Entwickler FQL-Abfragen nunmehr über die Graph API durchführen. Dazu ist ein GET-Zugriff auf den speziellen API-Endpunkt `/fql` notwendig, dem neben dem optionalen Access Token die FQL-Abfrage im Parameter `q` übergeben wird:

```
https://graph.facebook.com/fql?q=SELECT+uid2+FROM+friend+WHERE+uid1=me()&
access_token=...
```

Wichtig: Bei der Übergabe der FQL-Abfrage müssen Sie unbedingt auf eine korrekte URL-Codierung der Query achten. In PHP kann dies am einfachsten mit der Funktion `urlencode()` erfolgen.

Parameter	Beschreibung	Typ	Pflichtfeld
`q`	Auszuführende FQL-Abfrage. Die Abfrage muss dabei URL-codiert übergeben werden.	`string`	ja
`access_token`	Access Token zur Durchführung der Abfrage	`string`	nein

Tabelle 5.1 Parameter zum Ausführen von FQL über die Graph API

Zum Testen von FQL-Abfragen kann das in Abschnitt 3.1.9, »Der Graph API Explorer«, vorgestellte Tool verwendet werden. Dabei kann die FQL-Abfrage ohne weitere URL-Codierung im Parameter `q` an den API-Endpunkt angehängt werden, da die Codierung automatisch vom Graph API Explorer übernommen wird:

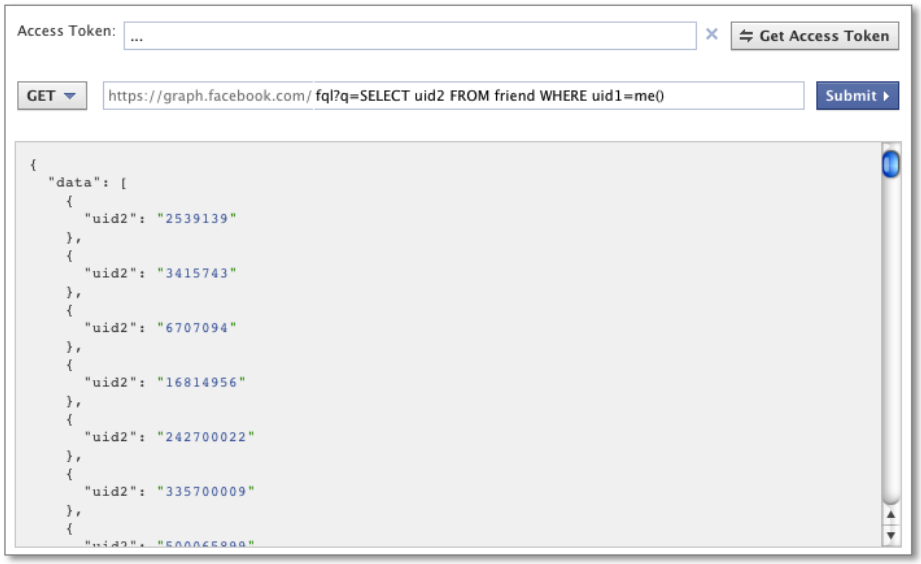

Abbildung 5.1 Durchführen einer FQL-Abfrage über den Graph API Explorer

Das Ergebnis von FQL-Abfragen wird – wie von der Graph API gewohnt – in JSON-Codierung zurückgegeben.

Das folgende Beispiel zeigt die Durchführung einer einfachen FQL-Abfrage über die Graph API. Der Code muss am Webserver unter dem Dateinamen *6-fql-graphapi.php* abgelegt werden:

```
<?
include_once("tools.php");
define('APP_ID', '214728715257742');
define('APP_SECRET', '...');
define('SITE_URL', 'http://apps.mycompany.com');

$login_url = "http://www.facebook.com/dialog/oauth?client_id=".
   APP_ID."&redirect_uri=".SITE_URL."/6-fql-graphapi.php";

if (empty($_REQUEST["code"])) {
   header("Location: ".$login_url);
   exit;
```

```
} else {
   $url =
      "https://graph.facebook.com/oauth/access_token?client_id=".
      APP_ID."&redirect_uri=".urlencode(SITE_URL.
             "/6-fql-graphapi.php")."&client_secret=".APP_SECRET.
             "&code=".$_REQUEST["code"];

      $access_token = curl($url);
      $access_token =
         substr($access_token,0,strpos($access_token,"&"));
      $fql = "SELECT uid,name,pic FROM user WHERE uid=me()";
      $url = "https://graph.facebook.com/fql?q=".
             urlencode($fql)."&".$access_token;
      $result = json_decode(curl($url));
      print "FQL-Query: ".$fql."<br/><br/>";
      print "FQL-Ergebnis:<br/><pre>";
      print_r($result);
}
?>
```

Listing 5.1 Ausführung einer FQL-Abfrage über die Graph API

Und so funktioniert dieses Beispiel:

- Da für viele FQL-Zugriffe analog zur Graph API ein Access Token erforderlich ist, wird im ersten Schritt geprüft, ob der Parameter `code` gesetzt ist – beim ersten Aufruf des Beispiels ist dies nicht der Fall, weswegen auf den OAuth-Dialog von Facebook umgeleitet wird. Nach erfolgreicher Autorisierung erfolgt der Callback mit gesetztem Parameter `code` auf das Skript.
- Mittels `code` kann nun das Access Token wie gewohnt bezogen werden.
- Die durchzuführende FQL-Abfrage wird in der Variablen `$fql` vorbereitet und an den API-Endpunkt `/fql?q=...` übergeben.
- Dabei muss die FQL-Abfrage im Parameter `q` mittels der PHP-Funktion `urlencode()` umgewandelt werden.
- Das Ergebnis der FQL-Abfrage wird wie üblich als JSON-Array zurückgegeben und kann mit der Funktion `json_decode()` in eine PHP-Struktur umgewandelt werden.

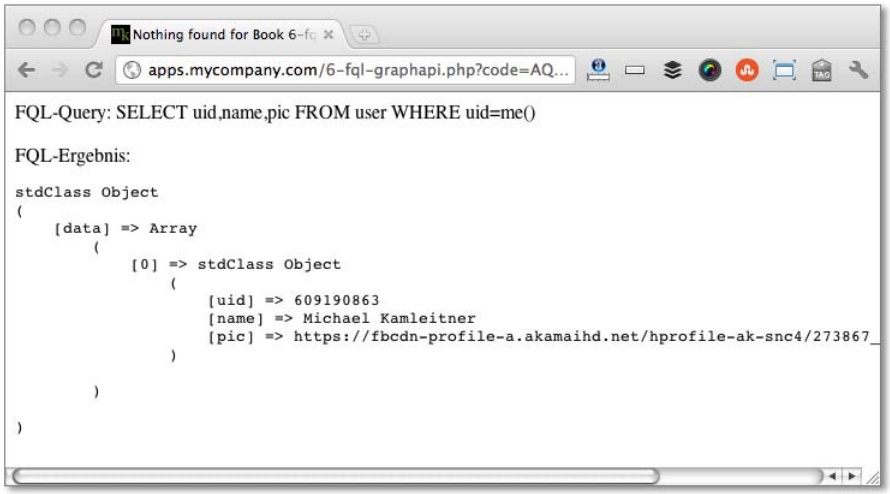

Abbildung 5.2 Anzeige des Ergebnisses einer einfachen FQL-Abfrage

5.2 FQL-Tabellen

Der folgende Abschnitt gibt einen vollständigen Überblick über die verfügbaren FQL-Tabellen und deren Felder. Eine Besonderheit kommt dabei der Feldeigenschaft INDIZIERT zu, denn die `WHERE`-Bedingungen einer FQL-Abfrage muss immer mindestens ein indiziertes Feld enthalten! Wenn Sie die Angabe eines indizierten Feldes vergessen, liefert die Abfrage eine entsprechende Fehlermeldung zurück.

Eine weitere Besonderheit von FQL betrifft Datums- bzw. Zeitfelder – diese liefern – anders als in der Graph API – keinen Zeitstempel im ISO-8601-Format, sondern die unter UNIX-Betriebssystemen übliche Notation als Differenz zwischen dem 1. Januar 1970, 00:00:00 Uhr, in Sekunden zurück.

Da Facebook, ähnlich wie bei der Graph API, laufend Anpassungen und Erweiterungen an den FQL-Tabellen vornimmt, ist ein Blick auf die offizielle Online-Dokumentation jedenfalls empfehlenswert.

Wichtiger Link:

- *https://developers.facebook.com/docs/reference/fql/* – offizielle Dokumentation zu den FQL-Tabellen

5.2.1 Albumtabelle

Die Tabelle `album` entspricht dem in Abschnitt 3.2.4, »Das Albumobjekt«, beschriebenen Objekt der Graph API. Hinsichtlich der Zugriffsberechtigungen gelten analog die gleichen Anforderungen an das Access Token wie beim Zugriff über die Graph API.

Beispiel – Abfrage der eigenen Alben des aktuellen Benutzers:

```
SELECT aid, object_id, name FROM album WHERE owner=me()
```

Beispiel – Abfrage eines bestimmten Albums anhand seiner ID:

```
SELECT description, link, name FROM album
WHERE object_id=10150397398555864
```

Feldname	Beschreibung	Typ	Indiziert
aid	Interne Facebook-ID des Albums – die Verwendung dieser ID stammt aus der Zeit vor Einführung der Graph API.	string	ja
object_id	Objekt-ID des Albums – entspricht jener ID, über die das Objekt üblicherweise mit der Graph API adressiert werden kann.	int	ja
owner	Besitzer des Albums (Benutzer oder Seite)	int	ja
cover_pid	Interne Facebook-ID des Cover-Fotos – die Verwendung dieser ID stammt aus der Zeit vor Einführung der Graph API.	string	nein
cover_object_id	Objekt-ID des Cover-Fotos – entspricht jener ID, über die das Objekt üblicherweise mit der Graph API adressiert werden kann.	int	nein
name	Titel des Albums	string	nein
created	Zeitstempel der Erstellung des Albums	UNIX-Zeitstempel	nein
modified	Zeitstempel der letzten Modifikation des Albums	UNIX-Zeitstempel	nein
modified_major	Zeitstempel der letzten signifikanten Modifikation des Albums (z. B. das Hochladen eines neuen Fotos)	UNIX-Zeitstempel	nein
description	Beschreibung des Albums	string	nein
location	Ort des Albums	string	nein

Tabelle 5.2 Felder der Albumtabelle

Feldname	Beschreibung	Typ	Indiziert
size	Anzahl der Fotos und Videos in diesem Album	int	nein
link	URL zum Album	string	nein
visible	Privatsphäre-Einstellung des Albums (friends, friends-of-friends, networks, everyone, custom)	string	nein
edit_link	URL zur Bearbeitungsseite des Albums (aufrufbar nur durch den Besitzer des Albums)	string	nein
type	Typ des Albums (profile, mobile, wall, normal)	string	nein
can_upload	Zeigt an, ob der aktuelle Benutzer Fotos in das Album hochladen kann.	boolean	nein
photo_count	Anzahl der Fotos in diesem Album	integer	nein
video_count	Anzahl der Videos in diesem Album	integer	nein

Tabelle 5.2 Felder der Albumtabelle (Forts.)

5.2.2 Application-Tabelle

Die Tabelle `application` entspricht dem in Abschnitt 3.2.5, »Das Anwendungsobjekt«, beschriebenen Graph-API-Objekt. Da mittels FQL ausschließlich öffentliche Informationen über Anwendungen ermittelt werden können, erfordert der Zugriff kein Access Token.

Beispiel – Auslesen einer bestimmten Anwendung über die bekannte CANVAS URL:

```
SELECT app_id, api_key, canvas_name FROM application
WHERE canvas_name='meinklub'
```

Feldname	Beschreibung	Typ	Indiziert
app_id	Objekt-ID der Anwendung	int	ja
api_key	öffentlicher API-Key der Anwendung	string	ja

Tabelle 5.3 Felder der Application-Tabelle

Feldname	Beschreibung	Typ	Indiziert
canvas_name	Entspricht der Option Canvas URL in den Einstellungen der Anwendung.	string	ja
display_name	Name der Anwendung	string	nein
icon_url	URL zum Icon der Anwendung	string	nein
logo_url	URL zum Logo der Anwendung	string	nein
company_name	Name des Herstellers der Anwendung	string	nein
developers	Liste der Benutzer, die als Entwickler der Anwendung eingetragen sind. Dieses Feld wird von Facebook mittlerweile immer leer zurückgeliefert, da die Entwickler der Anwendung nicht mehr veröffentlicht werden!	string	nein
description	Beschreibung der Anwendung	string	nein
daily_active_users	Anzahl der aktiven Benutzer am letzten Tag	integer	nein
weekly_active_users	Anzahl der aktiven Benutzer in der letzten Woche	integer	nein
monthly_active_users	Anzahl der aktiven Benutzer im letzten Monat	integer	nein
category	Kategorie der Anwendung	string	nein
subcategory	Unterkategorie der Anwendung	string	nein
is_facebook_app	Zeigt an, ob die Anwendung von Facebook selbst oder einem Drittthersteller entwickelt wurde.	boolean	nein
restriction_info	demographische Einschränkungen der Anwendung	JSON-Objekt	nein

Tabelle 5.3 Felder der Application-Tabelle (Forts.)

5.2.3 Apprequest-Tabelle

Applikationsanfragen werden, wie in Abschnitt 3.2.1, »Das User-Objekt«, beschrieben, in `apprequest`-Objekten im sozialen Graphen von Facebook gespeichert. Diese Objekte können ebenfalls über die FQL-Tabelle `apprequest` gelesen werden. Zum

Lesen der Applikationsanfragen eines Benutzers ist immer das Access Token dieser Person erforderlich.

Beispiel – Lesen aller Anfragen einer Anwendung an den aktuellen Benutzer:

```
SELECT recipient_uid, request_id, app_id FROM apprequest
WHERE recipient_uid = me() AND app_id = ...
```


Hinweis: Da immer nur die Applikationsanfragen der eigenen Anwendung ausgelesen werden dürfen, müssen die Felder `recipient_uid` und `app_id` immer gemeinsam in der `WHERE`-Klausel abgefragt werden!

Beispiel – Lesen einer bestimmten Applikationsanfrage:

```
SELECT request_id, app_id FROM apprequest
WHERE request_id = ...
```

Feldname	Beschreibung	Typ	Indiziert
`request_id`	Objekt-ID der Applikationsanfrage	`int`	ja
`app_id`	Objekt-ID der Anwendung, von der die Anfrage erzeugt wurde	`int`	ja (nur gemeinsam mit `recipient_uid`)
`recipient_uid`	ID des Benutzers, an den die Anfrage verschickt wurde	`int`	ja (nur gemeinsam mit `app_id`)
`sender_uid`	ID des Benutzers, der die Anfrage verschickt hat	`int`	nein
`message`	Nachricht, mit der die Anfrage verschickt wurde	`string`	nein
`data`	optional mit der Anfrage verknüpftes JSON-Objekt	`string`	nein
`created_time`	Zeitstempel der Anfrage	UNIX-Zeitstempel	nein

Tabelle 5.4 Felder der Apprequest-Tabelle

5.2.4 Checkin-Tabelle

Die Tabelle `checkin` entspricht dem in Abschnitt 3.2.14, »Das Checkin-Objekt«, beschriebenen Objekt der Graph API. Hinsichtlich der Zugriffsberechtigungen gelten analog die gleichen Anforderungen an das Access Token wie beim Zugriff über die Graph API.

Beispiel – Lesen der neuesten Checkins einer bestimmten Seite bzw. eines bestimmten Ortes:

```
SELECT checkin_id, author_uid, message, timestamp
FROM checkin WHERE page_id = 163483520335945
ORDER BY timestamp DESC
```

Beispiel – Lesen der neuesten Checkins des aktuellen Benutzers:

```
SELECT checkin_id, page_id, message FROM checkin
WHERE author_uid= me()
ORDER BY timestamp DESC
```

Feldname	Beschreibung	Typ	Indiziert
`checkin_id`	Objekt-ID des Checkins	`int`	ja
`author_uid`	ID des Benutzers, der den Checkin veröffentlicht hat	`int`	ja
`page_id`	ID der Seite, die den Ort des Checkins repräsentiert	`int`	ja
`app_id`	ID der Anwendung, über die der Checkin veröffentlicht wurde	`int`	nein
`post_id`	ID des Postings, das mit dem Checkin veröffentlicht wurde	`int`	nein
`coords`	Koordinaten des Checkins	Array mit den Feldern `latitude` und `longitude`	nein
`timestamp`	Zeitstempel des Checkins	UNIX-Zeitstempel	nein

Tabelle 5.5 Felder der Checkin-Tabelle

Feldname	Beschreibung	Typ	Indiziert
tagged_uids	weitere Benutzer, die in diesem Checkin markiert wurden	Array, bestehend aus Benutzer-IDs	nein
message	optionale Nachricht zum Checkin	string	nein

Tabelle 5.5 Felder der Checkin-Tabelle (Forts.)

5.2.5 Comment-Tabellen

Die Tabelle comment entspricht grundsätzlich dem in Abschnitt 3.2.15, »Das Kommentarobjekt«, beschriebenen Objekt der Graph API. Hinsichtlich der Zugriffsberechtigungen gelten analog die gleichen Anforderungen an das Access Token wie beim Zugriff über die Graph API.

Beispiel – Lesen der Kommentare zu einem Wall-Posting:

```
SELECT post_id, post_fbid, fromid, object_id, text, time
FROM comment WHERE object_id = 177372615695106
```

Die Tabelle comments erlaubt auch den Zugriff auf Kommentare, die über das Social Plugin fb:comments veröffentlicht wurden. fb:comments erlaubt die Einbindung eines standardisierten Kommentarmoduls auf beliebigen Webseiten und -apps und wird in Abschnitt 6.9, »Die Kommentarbox«, ausführlich beschrieben. Das Kommentarmodul wird grundsätzlich über die URL der Webseite, auf der es eingesetzt wird, identifiziert. Um die Kommentare eines bestimmten Kommentarmoduls einzulesen, muss daher mithilfe der Tabelle link_stat und einer Unterabfrage die Objekt-ID der Webseite, auf der das Kommentarmodul eingebunden wurde, herausgefunden werden:

```
SELECT post_fbid, fromid, object_id, text, time
FROM comment WHERE object_id in
 (SELECT comments_fbid FROM link_stat
  WHERE url =
  'http://developers.facebook.com/docs/reference/fql/comment/')
```

Feldname	Beschreibung	Typ	Indiziert
xid	externe ID zur Identifikation des Kommentarmoduls	string	ja

Tabelle 5.6 Felder der Comment-Tabelle

Feldname	Beschreibung	Typ	Indiziert
object_id	ID des Objekts (Foto, Wall-Posting etc.), das kommentiert wurde	string	ja
post_id	ID des Kommentars	string	ja
fromid	ID des Benutzers, der den Kommentar veröffentlicht hat	int	nein
time	Zeitstempel des Kommentars	UNIX-Zeitstempel	nein
text	Text des Kommentars	string	nein
id	eindeutige Kommentar-ID innerhalb einer xid	int	nein
username	Name des Benutzers, wenn dieser nicht über einen Facebook-Account kommentiert hat (das Kommentar-Plugin unterstützt Benutzer-Logins via Yahoo!, AOL und Hotmail)	string	nein
reply_xid	Ziel-ID für ein Wall-Posting, das vom Benutzer veröffentlicht wurde	string	nein
post_fbid	Objekt-ID des Kommentars	string	nein
app_id	ID der Anwendung, die mit dem Kommentar verknüpft ist	int	nein
likes	Anzahl der Gefällt mir des Kommentars	int	nein
comments	weitere Kommentare, die als Antwort zu diesem Kommentar veröffentlicht wurden	Array an Objekten mit den Feldern id, from, message und created_time	nein
user_likes	Zeigt, ob der aktuelle Benutzer bei diesem Kommentar auf Gefällt mir geklickt hat.	boolean	nein
is_private	Zeigt, ob der Kommentar privat ist.	boolean	nein

Tabelle 5.6 Felder der Comment-Tabelle (Forts.)

Hinweis: In früheren Versionen des Kommentarmoduls wurde die Verknüpfung zwischen einbindender Webseite und Modul nicht durch die URL, sondern durch die Anwendungs-ID in Kombination mit einem manuell gesetzten Parameter `xid` hergestellt. Die Felder der Tabelle `comments` enthalten diese `xid` aus Kompatibilitätsgründen nach wie vor.

Comments-Info-Tabelle

In diesem Zusammenhang muss aus Gründen der Vollständigkeit auch die Tabelle `comments_info` erwähnt werden, aus der die mittlerweile eingestellten `xid`-basierten Kommentarmodule einer bestimmten Anwendungs-ID abgefragt werden können:

Feldname	Beschreibung	Typ	Indiziert
`app_id`	ID der Anwendung	`string`	ja
`xid`	externe ID zur Identifikation des Kommentarmoduls	`string`	nein
`count`	Anzahl der Kommentare	`int`	nein
`updated_time`	Zeitstempel der letzten Modifikation des Albums	UNIX-Zeitstempel	nein

Tabelle 5.7 Felder der Comments-Info-Tabelle

5.2.6 Connection-Tabelle

Die Tabelle `connection` enthält alle Verbindungen des aktuellen Benutzers (Freundschaft) zu anderen Benutzern und Seiten (Fan). Um diese Verbindungen einzulesen, ist ein Access Token mit der erweiterten Berechtigung `read_stream` notwendig. Die Tabelle `connection` ist immer nur für den aktuellen Benutzer verfügbar und muss daher folgendermaßen abgefragt werden:

```
SELECT target_id,target_type FROM connection WHERE source_id=me()
```

Feldname	Beschreibung	Typ	Indiziert
`source_id`	ID des Benutzers, von dem die Verbindung ausgeht	`int`	ja
`target_id`	ID des Benutzers oder der Seite, mit der der Benutzer verknüpft ist	`int`	nein

Tabelle 5.8 Felder der Connection-Tabelle

Feldname	Beschreibung	Typ	Indiziert
target_type	Zeigt an, ob das verknüpfte Objekt ein Benutzer (user) oder eine Seite (page) ist.	string	nein
is_following	Zeigt an, ob der Benutzer, von dem die Verknüpfung ausgeht, Pinnwand-Beiträge des verknüpften Benutzers oder der verknüpften Seite ausgeblendet hat (false) oder nicht (true).	boolean	nein

Tabelle 5.8 Felder der Connection-Tabelle (Forts.)

5.2.7 Cookies-Tabelle

Laut offizieller Dokumentation enthält die Tabelle cookies eine Liste aller aktuell gesetzten Browser-Cookies des aktuellen Benutzers. Zum Zugriff wird ein Access Token benötigt, womit lediglich die Cookies des eigenen Benutzers innerhalb der Anwendung, für die das Token ausgestellt wurde, abgefragt werden können:

```
SELECT uid, name, value, expires, path
FROM cookies WHERE uid = me()
```

Hinweis: Die Tabelle cookies hat während der Arbeiten an diesem Buch ausschließlich leere Ergebnisse zurückgeliefert, was nahelegt, dass diese Funktion nur in Verbindung mit der mittlerweile eingestellten HTTP-REST-API funktioniert hat. In den aktuellen Versionen des PHP SDKs und seit der Einführung von OAuth 2.0 dürfte dieser Tabelle daher keine Bedeutung mehr zukommen.

Feldname	Beschreibung	Typ	Indiziert
uid	ID des Benutzer, dem das Cookie zugeordnet ist	int	ja
name	Name des Cookies	string	nein
value	Inhalt des Cookies	string	nein
expires	Zeitstempel, zu dem die Gültigkeit des Cookies ausläuft	boolean	nein
path	URL-Pfad relativ zur Canvas URL, mit der das Cookie verknüpft werden soll	string	nein

Tabelle 5.9 Felder der Cookies-Tabelle

5.2.8 Developer-Tabelle

Die Tabelle `developer` enthält alle Anwendungen eines Benutzers, bei denen dieser als Administrator, Entwickler, Tester oder Insights-Benutzer eingetragen ist. Der Zugriff ist nur auf die Anwendungen des aktuellen Benutzers möglich und erfordert ein beliebiges Access Token:


```
SELECT application_id FROM developer WHERE developer_id = me()
```

Feldname	Beschreibung	Typ	Indiziert
developer_id	ID des Benutzers, der der Anwendung als Administrator, Entwickler, Tester oder Insight-Benutzer zugeordnet ist	int	ja
application_id	ID der Anwendung	int	nein

Tabelle 5.10 Felder der Developer-Tabelle

5.2.9 Domain-Tabellen

Die Tabelle `domain` entspricht grundsätzlich dem in Abschnitt 3.2.6, »Das Domain-Objekt«, beschriebenen Objekt der Graph API und erlaubt das Auslesen der Zuordnung von Domain-Namen zu ihren Objekt-IDs im sozialen Graphen.

Beispiel – Auslesen der Objekt-ID einer des Namens nach bekannten Domain

```
SELECT domain_id FROM domain
WHERE domain_name='www.facebook.com'
```

Beispiel – Auslesen des Domain-Namens einer der ID nach bekannten Domain:

```
SELECT domain_name FROM domain
WHERE domain_id= 369296215699
```

Feldname	Beschreibung	Typ	Indiziert
domain_id	Objekt-ID der Domin	int	ja
domain_name	Domain-Name	string	ja

Tabelle 5.11 Felder der Domain-Tabelle

Domain-Administrator-Tabelle

Neben der `domain`-Tabelle wird in der Tabelle `domain_admin` gespeichert, ob eine Domain von einem Facebook-Benutzer beansprucht (*claim*) wurde. Dies geschieht üblicherweise über Facebook Insights, das Statistik-Tool von Facebook, um Einblick in Nutzungsstatistiken einer Domain zu erhalten. Domains können von Benutzern, Seiten und Anwendungen bzw. deren Administratoren beansprucht werden.

Feldname	Beschreibung	Typ	Indiziert
`owner_id`	ID des beanspruchenden Benutzers bzw. der beanspruchenden Seite oder Anwendung	`int`	ja
`domain_id`	Objekt-ID der Domain	`int`	ja

Tabelle 5.12 Felder der Domain-Administrator-Tabelle

Zum Beanspruchen einer Domain ist es notwendig, durch das Setzen eines speziellen Open Graph Tags `fb:admins` auf der eigenen Webseite die Inhaberschaft der Domain nachzuweisen. Ab diesem Zeitpunkt wird die erfolgreiche Beanspruchung der Domain auch in der FQL-Tabelle `domain_admins` repräsentiert.

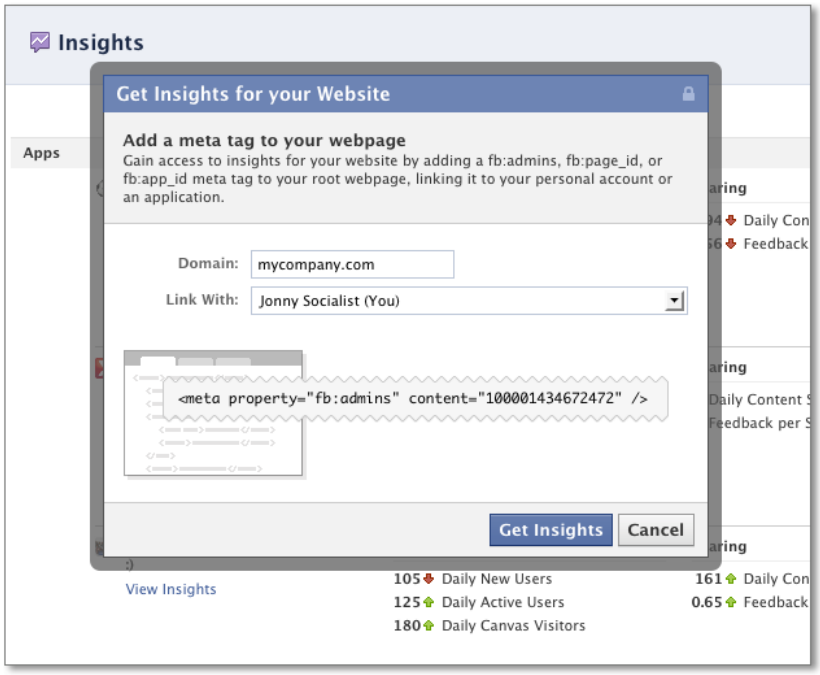

Abbildung 5.3 Beanspruchen einer Domain in Facebook Insights

5.2.10 Event-Tabellen

Die Tabelle `event` entspricht grundsätzlich dem in Abschnitt 3.2.9, »Das Veranstaltungsobjekt«, beschriebenen Objekt der Graph API. Hinsichtlich der Zugriffsberechtigungen gelten analog die gleichen Anforderungen an das Access Token wie beim Zugriff über die Graph API.

Beispiel – Auslesen eines bestimmten Events:

```
SELECT name, venue, location, start_time FROM event
WHERE eid = 209798352393506
```

Feldname	Beschreibung	Typ	Indiziert
`eid`	numerische ID der Veranstaltung	`int`	ja
`name`	Name der Veranstaltung	`string`	nein
`tagline`	der Untertitel bzw. die Zusammenfassung der Veranstaltung	`string`	nein
`nid`	numerische ID des Benutzernetzwerks, zu dem die Veranstaltung gehört	`int`	nein
`pic_small`	URL zum kleinen Profilbild der Veranstaltung (maximal 50 x 150 Pixel)	`string`	nein
`pic_big`	URL zum großen Profilbild der Veranstaltung (maximal 200 x 600 Pixel)	`string`	nein
`pic_square`	URL zum quadratischen Profilbild der Veranstaltung (50 x 50 Pixel)	`string`	nein
`pic`	URL zum mittelgroßen Profilbild der Veranstaltung (maximal 100 x 300 Pixel)	`string`	nein
`host`	Name des Veranstalters	`string`	nein
`description`	Beschreibung der Veranstaltung	`string`	nein

Tabelle 5.13 Felder der Event-Tabelle

Feldname	Beschreibung	Typ	Indiziert
event_type	Kategorie der Veranstaltung (diese Event-Einstellung wird von Facebook nicht mehr bereitgestellt und steht nur bei historischen Events zur Verfügung)	string	nein
event_subtype	Unterkategorie der Veranstaltung (diese Event-Einstellung wird von Facebook nicht mehr bereitgestellt und steht nur bei historischen Events zur Verfügung)	string	nein
start_time	Zeitstempel des Beginns der Veranstaltung	UNIX-Zeitstempel	nein
end_time	Zeitstempel des Endes der Veranstaltung	UNIX-Zeitstempel	nein
creator	Benutzer- oder Seiten-ID des Erstellers der Veranstaltung	int	nein
update_time	Zeitstempel der letzten Aktualisierung der Veranstaltung	UNIX-Zeitstempel	nein
venue	Ort der Veranstaltung	Array mit den Feldern street, city, state, zip, country, latitude und longitude	nein
location	Name des Ortes der Veranstaltung	string	nein
privacy	Privatsphäre-Einstellung der Veranstaltung (OPEN, CLOSED oder SECRET)	string	nein
hide_guest_list	Zeigt an, ob die Gästeliste versteckt werden soll.	boolean	nein
can_invite_friends	Zeigt an, ob der Benutzer, der die FQL-Anfrage stellt, Freunde zu der Veranstaltung einladen darf.	boolean	nein

Tabelle 5.13 Felder der Event-Tabelle (Forts.)

Event-Member-Tabelle

Die Tabelle `event_member` enthält die Liste der Teilnehmer einer Veranstaltung. Je nach Art des Zugriffs ist ein Access Token analog, wie in Abschnitt 3.2.9, »Das Veranstaltungsobjekt«, beschrieben, notwendig.

Beispiel – Lesen aller Teilnehmer einer bestimmten Veranstaltung:

```
SELECT eid, uid, rsvp_status FROM event_member
WHERE eid = 209798352393506;
```

Beispiel – Lesen aller Veranstaltungen des aktuellen Benutzers:

```
SELECT eid, uid, rsvp_status FROM event_member WHERE uid=me()
```

Feldname	Beschreibung	Typ	Indiziert
`eid`	numerische ID der Veranstaltung	`int`	ja
`uid`	Benutzer-ID des Teilnehmers	`int`	nein
`rsvp_status`	Teilnahmestatus des Benutzers (`attending`, `unsure`, `declined` oder `not_replied`)	`string`	nein
`start_time`	UNIX-Zeitstempel des Beginns der Veranstaltung	UNIX-Zeitstempel	nein

Tabelle 5.14 Felder der Event-Member-Tabelle

5.2.11 Family-Tabelle

Die FQL-Tabelle `family` erlaubt den Zugriff auf die Familienmitglieder eines Benutzers. Die Tabelle entspricht der `family`-Verknüpfung des User-Objekts, wie in Abschnitt 3.2.1, »Das User-Objekt«, beschrieben. Das verwendete Access Token muss zum Lesen der Familienmitglieder die Berechtigung `user_relationships` (aktueller Benutzer) bzw. `friends_relationships` (Freunde des aktuellen Benutzers) enthalten.

Beispiel – Lesen aller Familienmitglieder des aktuellen Benutzers:

```
SELECT uid, name, birthday, relationship FROM family
WHERE profile_id = me()
```

Feldname	Beschreibung	Typ	Indiziert
`profile_id`	numerische ID des abgefragten Benutzers	`int`	ja
`uid`	Numerische ID des verknüpften Familienmitglieds. Nur gesetzt, wenn dieses die Verknüpfung bestätigt hat.	`int`	nein
`name`	Name des Familienmitglieds	`string`	nein
`birthday`	Geburtstag des Familienmitglieds. Nur gesetzt, wenn dieses die Verknüpfung *nicht* bestätigt hat.	`string`	nein
`relationship`	Art der Familienbeziehung (`sister`, `brother` etc.)	`string`	nein

Tabelle 5.15 Felder der Family-Tabelle

5.2.12 Friend-Tabelle

Die FQL-Tabelle `friend` bildet die Freundschaftsbeziehungen im sozialen Graphen von Facebook ab und entspricht damit der `friends`-Beziehung, wie in Abschnitt 3.2.1, »Das User-Objekt«, beschrieben.

Beispiel: Lesen der vollständigen Freundesliste des aktuellen Benutzers:

```
SELECT uid2 FROM friend WHERE uid1 = me()
```

Dabei können die Spalten `uid1` und `uid2` beliebig vertauscht werden, da die FQL-Tabellen die Freundschaftsbeziehungen bidirektional abbilden.

Hinweis: Auch hier gilt – die vollständige Liste an Freunden kann nur für den eigenen, aktuellen Benutzer ausgelesen werden, nicht aber für die Freunde des aktuellen Benutzers!

Beispiel: Über FQL kann auch die Freundschaftsbeziehung zweier Benutzer geprüft werden:

```
SELECT uid1, uid2 FROM friend
WHERE uid1 = 532617296 AND uid2 = 633616172
```

Sind die beiden Benutzer nicht befreundet, wird als Ergebnis ein leeres Array zurückgeliefert.

Hinweis: Die Freundschaftsbeziehung kann für alle Freunde des aktuellen Benutzers geprüft werden!

Feldname	Beschreibung	Typ	Indiziert
uid1	numerische ID des ersten Benutzers in der Freundschaftsbeziehung	int	ja
uid2	numerische ID des zweiten Benutzers in der Freundschaftsbeziehung	int	ja

Tabelle 5.16 Felder der Friend-Tabelle

5.2.13 Friend-Request-Tabelle

Die FQL-Tabelle `friend_request` bildet alle ausstehenden Freundschaftsanfragen des aktuellen Benutzers ab. Sie entspricht der `friendrequests`-Verbindung, wie in Abschnitt 3.2.1, »Das User-Objekt«, beschrieben, und erfordert wie diese ein Access Token mit der erweiterten Berechtigung `read_requests`.

Beispiel – Lesen aller ausstehenden eingehenden Freundschaftsanfragen des aktuellen Benutzers:

```
SELECT uid_from, time, message
FROM friend_request WHERE uid_to = me()
```

Beispiel – Lesen einer bestimmten ausgehenden Freundschaftsanfrage des aktuellen Benutzers:

```
SELECT time, message FROM friend_request
WHERE uid_from = me() AND uid_to = 633616172
```

Hinweis: Das Auslesen aller ausgehenden Freundschaftsanfragen des aktuellen Benutzers wird von FQL nicht unterstützt, da das Feld `uid_from` nicht indiziert ist, sondern nur in Kombination mit `uid_to` abgefragt werden kann!

Feldname	Beschreibung	Typ	Indiziert
uid_to	numerische ID des Benutzers, an den die Anfrage gesendet wurde	int	ja
uid_from	numerische ID des Benutzers, der die Anfrage gesendet hat	int	nein (Abfrage nur in Kombination mit uid_from möglich)
time	UNIX-Zeitstempel des Beginns der Veranstaltung	UNIX-Zeitstempel	nein
message	persönliche Nachricht zur Anfrage	string	nein
unread	Zeigt an, ob die Anfrage bereits gelesen wurde. Wird nur zurückgeliefert, wenn uid_to dem aktuellen Benutzer entspricht.	boolean	nein

Tabelle 5.17 Felder der Friend-Request-Tabelle

5.2.14 Friendlist-Tabellen

Die FQL-Tabelle `friendlist` enthält die Freundeslisten des aktuellen Benutzers und entspricht damit der `friendlists`-Verbindung, wie in Abschnitt 3.2.1, »Das User-Objekt«, beschrieben. Zum Lesen der Freundeslisten ist ein Access Token mit der erweiterten Berechtigung `read_friendlists` notwendig.

Beispiel – Lesen der Freundeslisten des aktuellen Benutzers:

```
SELECT flid, owner, name FROM friendlist WHERE owner=me()
```

Feldname	Beschreibung	Typ	Indiziert
owner	numerische ID des Benutzers, der die Freundesliste besitzt	int	ja
flid	numerische ID der Freundesliste	int	nein
name	Name der Freundesliste	string	nein

Tabelle 5.18 Felder der Friendlist-Tabelle

Friendlist-Member-Tabelle

Die Mitglieder einer Freundesliste können aus der Tabelle `friendlist_member` ausgelesen werden. Auch hierzu ist die Berechtigung `read_friendlists` notwendig.

Beispiel – Lesen der Mitglieder einer bestimmten Freundesliste:

```
SELECT uid, flid FROM friendlist_member
WHERE flid = 10150456156070864
```

Beispiel – Lesen aller Mitglieder aus allen Freundeslisten des aktuellen Benutzers unter Verwendung einer verschachtelten Abfrage:

```
SELECT uid, flid FROM friendlist_member
WHERE flid IN (SELECT flid FROM friendlist WHERE owner=me())
```

Feldname	Beschreibung	Typ	Indiziert
`flid`	numerische ID der Freundesliste	`int`	ja
`uid`	numerische ID des Benutzers, der Mitglied der Liste ist	`int`	nein

Tabelle 5.19 Felder der Friendlist-Member-Tabelle

5.2.15 Group-Tabellen

Die FQL-Tabelle `group` bildet Gruppenobjekte, wie in Abschnitt 3.2.3, »Das Gruppenobjekt«, beschrieben, ab. Während öffentliche Gruppen mit einem beliebigen Access Token ausgelesen werden können, sind für private Gruppen die `user_groups`- bzw. `friends_groups`-Berechtigung und das Access Token eines Gruppenmitglieds notwendig.

Beispiel – Lesen einer öffentlichen Gruppe:

```
SELECT gid, name, description, pic_small
FROM group WHERE gid = 146797922030397
```

Hinweis: Die Gruppenfunktion von Facebook wurde im Oktober 2010 grundlegend verändert. Felder, die in der folgenden Übersicht mit dem Vermerk unbenutzt ab Version 1 versehen sind, werden für Gruppen, die nach Oktober 2010 erstellt wurden, nicht mehr verwendet.

Feldname	Beschreibung	Typ	Indiziert
gid	numerische ID der Gruppe	int	ja
name	Name der Gruppe	string	nein
nid	numerische ID des Benutzernetzwerks, zu dem die Gruppe gehört	int	nein
pic_small	URL zum kleinen Profilbild der Gruppe (maximal 50 x 150 Pixel)	string	nein
pic_big	URL zum großen Profilbild der Gruppe (maximal 200 x 600 Pixel)	string	nein
pic	URL zum mittelgroßen Profilbild der Gruppe (maximal 100 x 300 Pixel)	string	nein
description	Beschreibung der Gruppe (unbenutzt ab Version 1)	string	nein
group_type	Kategorie der Gruppe (unbenutzt ab Version 1)	string	nein
group_subtype	Unterkategorie der Gruppe	string	nein
recent_news	Neuigkeiten-Feld der Gruppe	string	nein
creator	ID des Benutzers, der die Gruppe angelegt hat	int	nein
update_time	UNIX-Zeitstempel des letzten Updates der Gruppe	UNIX-Zeitstempel	nein
office	Ort/Büro der Gruppe (unbenutzt ab Version 1)	string	nein
website	Website der Gruppe (unbenutzt ab Version 1)	string	nein
venue	geographischer Ort der Gruppe (unbenutzt ab Version 1)	Objekt mit den Feldern street, city, state, country, zip, latitude und longitude	nein

Tabelle 5.20 Felder der Group-Tabelle

Feldname	Beschreibung	Typ	Indiziert
icon	URL zum Icon der Gruppe (16 x 16 Pixel)	string	nein
icon34	URL zum Icon der Gruppe (34 x 34 Pixel)	string	nein
icon68	URL zum Icon der Gruppe (68 x 68 Pixel)	string	nein
email	E-Mail-Adresse, die genutzt werden kann, um in die Gruppe zu posten	string	nein
version	Kennzeichen, ob die Gruppe vor Oktober 2010 (Wert 0) oder danach (Wert 1) erstellt wurde	int	nein

Tabelle 5.20 Felder der Group-Tabelle (Forts.)

Group-Member-Tabelle

Um die Mitglieder einer Gruppe zu ermitteln, kann die FQL-Tabelle `group_member` ausgelesen werden. Auch hier gilt: Die Mitglieder einer öffentlichen Gruppe können mit einem beliebigen Access Token, die Mitglieder einer privaten Gruppe mit einem Access Token mit `user_groups`- bzw. `friends_groups`-Berechtigung eines Mitglieds ausgelesen werden.

Beispiel – Lesen aller Mitglieder einer bestimmten Gruppe:

```
SELECT gid, uid FROMgroup_member
WHERE gid = 146797922030397
```

Beispiel – Lesen aller Mitgliedschaften des aktuellen Benutzers:

```
SELECT gid, uid FROM group_member WHERE uid=me()
```

Feldname	Beschreibung	Typ	Indiziert
uid	numerische ID des Mitglieds	int	ja
gid	numerische ID der Gruppe	int	ja
administrator	Zeigt an, ob das Mitglied Gruppenadministrator ist.	boolean	nein

Tabelle 5.21 Felder der Group-Member-Tabelle

Feldname	Beschreibung	Typ	Indiziert
positions	Berechtigungsstufe des Mitglieds – OWNER, ADMIN oder OFFICER	string	nein
unread	Anzahl der ungelesenen Nachrichten in der Gruppe (nur verfügbar, wenn das Feld uid in der WHERE-Klausel abgefragt wird)	int	nein
bookmark_order	Position der Gruppe in den Bookmarks des Benutzers (nur verfügbar, wenn das Feld uid in der WHERE-Klausel abgefragt wird)	int	nein

Tabelle 5.21 Felder der Group-Member-Tabelle (Forts.)

5.2.16 Insights-Tabelle

Facebook bietet zahlreiche Nutzungsstatistiken zu Objekten vom Typ page und application. Abgesehen von einigen öffentlich einsehbaren Metriken, ist zum Zugriff ein Anwendungs-Access-Token (bei anwendungsbezogenen Statistiken) oder ein Access Token mit der read_insights-Berechtigung (Anwendungs-, Seiten- oder domainbezogene Statistiken notwendig) eines Benutzers mit Administrationsrechten notwendig.

Beispiel – Abfrage der Metrik application_installed_users für den Zeitraum 1. Dezember bis 31. Dezember 2011:

```
SELECT metric, value FROM insights
WHERE object_id= 32788395891
AND metric='application_installed_users'
AND end_time=end_time_date('2011-12-31')
AND period=period('month')
```

Feldname	Beschreibung	Typ	Indiziert
object_id	numerische ID der Seite, Anwendung oder Domain	int	ja
metric	Bezeichnung der Metrik	string	ja
end_time	UNIX-Zeitstempel, der das Ende des Betrachtungszeitraums festlegt	UNIX-Zeitstempel	ja

Tabelle 5.22 Felder der Insights-Tabelle

Feldname	Beschreibung	Typ	Indiziert
`period`	Betrachtungszeitraum in Sekunden (1 Tag = 86.400, 1 Woche = 604.800, 1 Monat = 2.592.000, 0 = Lifetime)	`int`	ja
`value`	Wert der abgefragten Metrik im spezifizierten Zeitraum	`string`	nein

Tabelle 5.22 Felder der Insights-Tabelle (Forts.)

- Beim Abfragen der `insights`-Tabelle ist zu beachten, dass in der `WHERE`-Klausel immer die Felder `object_id`, `metric`, `period` und `end_time` vorkommen müssen.
- Es wird also immer der Wert einer bestimmten Metrik für ein bestimmtes Objekt (Anwendung, Seite oder Domain) in einem bestimmten Betrachtungszeitraum abgefragt.
- Der Betrachtungszeitraum ergibt sich aus der Angabe `end_time`, die, als UNIX-Zeitstempel codiert, das Ende des Zeitraums festlegt. `period` legt dabei fest, wie lang der Zeitraum – ausgehend von seinem Endpunkt – gewählt werden soll.
- `period` kann entweder in Sekunden angegeben werden oder mit der Hilfsfunktion `period()`, der der gewünschte Zeitraum als gut lesbarer String übergeben wird: `period('day')`, `period('week')`, `period('month')`oder `period('lifetime')`. *Hinweis:* Die verfügbaren Zeitperioden unterscheiden sich von Metrik zu Metrik.
- `end_time` kann entweder als UNIX-Zeitstempel (Anzahl der Sekunden seit 1. Januar 1970, 00:00 Uhr) angegeben werden oder über die Hilfsfunktion `end_time_date()`, der ein Datumsstring in der Form »YYYY-MM-DD« übergeben werden kann.

Facebook bietet mittlerweile über hundert Metriken für Anwendungen, Seiten oder Domains an. Aufzählung und Beschreibung der einzelnen Zahlen würden den Rahmen dieses Buches sprengen, deshalb verweise ich Sie an dieser Stelle auf die offizielle Dokumentation.

Wichtiger Link:

- *https://developers.facebook.com/docs/reference/fql/insights/* – offizielle Dokumentation zur FQL-Tabelle `insights`

5.2.17 »Like«-Tabelle

Wall-Postings, Videos, Fotos, Fotoalben, Notizen und Links, bei denen ein Benutzer auf Gefällt mir geklickt hat, werden in der FQL-Tabelle `like` abgebildet. Die Tabelle entspricht damit der `likes`-Verknüpfung der genannten Objekte in der Graph API.

Hinweis: Gefällt mir von Facebook-Seiten oder anderen Open-Graph-Objekten werden nicht in der FQL-Tabelle `like`, sondern in `page_fan` abgebildet!

Beispiel – Lesen aller Gefällt mir eines bestimmten Fotoalbums:

```
SELECT user_id FROM like WHERE object_id=10150409321060864
```

Beispiel – Lesen aller Gefällt mir des aktuellen Benutzers. Diese Art des Zugriffs erfordert die erweiterte Berechtigung `read_stream`!

```
SELECT user_id, object_id, object_type, post_id
FROM like WHERE user_id=me()
```

Feldname	Beschreibung	Typ	Indiziert
`object_id`	numerische ID des Objekts (Video, Foto, Link, Notiz, Album etc.)	`int`	ja
`post_id`	ID des zum Gefällt mir gehörenden Wall-Postings	`int`	ja
`user_id`	ID des Benutzers, der auf Gefällt mir geklickt hat	`int`	ja
`object_type`	Typ des Objekts (`photo`, `album`, `event`, `group`, `note`, `link`, `video`, `application`, `status`, `check-in`, `review`, `comment`, `post`)	`string`	nein

Tabelle 5.23 Felder der »Like«-Tabelle

5.2.18 Linktabelle

Die FQL-Tabelle `link` enthält die von einem Benutzer auf der Pinnwand veröffentlichten Links und entspricht damit dem in Abschnitt 3.2.11, »Das Linkobjekt«, beschriebenen Graph-Objekt. Analog dazu ist ein beliebiges Access Token zum Lesen von öffentlichen bzw. die `read_stream`-Berechtigung zum Lesen von privaten Links notwendig.

Beispiel – Lesen aller Links einer Facebook-Seite:

```
SELECT link_id, owner, title, summary, url, image_urls
FROM link WHERE owner=129842310398883
```

Beispiel – Lesen eines bestimmten Links:

```
SELECT link_id, owner, title, summary, url, image_urls
FROM link WHERE link_id=336134509732399
```

Feldname	Beschreibung	Typ	Indiziert
link_id	numerische ID des Links	int	ja
owner	ID des Benutzers, der Seite oder der Anwendung, die den Link veröffentlicht hat	int	ja
owner_comment	vom teilenden Benutzer bzw. der Seite/Anwendung verfasste Nachricht zum Link	string	nein
created_time	UNIX-Zeitstempel der Veröffentlichung des Links	UNIX-Zeitstempel	nein
title	Linktext bzw. Titel der geteilten URL	string	nein
summary	Beschreibungstext des Links	string	nein
url	URL des geteilten Links	string	nein
picture	URL zum Thumbnail-Bild des Links	string	nein
image_urls	URLs zu weiteren Bildern, die mit dem Artikel verknüpft sind	Array	nein

Tabelle 5.24 Felder der Linktabelle

5.2.19 Link-Stat-Tabelle

Der Tabelle link_stat kommt besondere Bedeutung zu, da die in ihr abgebildeten Daten derzeit nicht über die Graph API, sondern ausschließlich per FQL zugänglich sind. link_stat enthält Informationen zu sozialen Interaktionen, die auf verschiedenen Webseiten bzw. URLs vonstattengehen. Die abgefragten Webseiten können z. B. Fan-Seiten auf Facebook sein, aber auch jede externe Webseite. die mittels Social Plugins und Open Graph oder auch durch einfaches Sharing Teil des sozialen Graphen der Facebook-Plattform geworden sind.

Beispiel – Lesen der Linkstatistiken einer externen Seite:

```
SELECT url, normalized_url, share_count, like_count, comment_count,
total_count FROM link_stat WHERE url='facebook.com/diesocialisten'
```

Beispiel – gleichzeitiges Lesen der Linkstatistiken von mehreren externen Seiten mittels IN-Operator:

```
SELECT url, normalized_url, share_count, like_count, comment_count,
total_count FROM link_stat
WHERE url in ('http://die.socialisten.at/2011/10/last-fm-scrobbler-for-face-
book-open-graph-timeline/', 'http://die.socialisten.at/2011/08/facebook-fluid-
canvas-layout-f-apps-games/')
```

Beispiel – Lesen der Linkstatistiken einer Facebook-Seite:

```
SELECT url, normalized_url, share_count, like_count, comment_count,
total_count FROM link_stat
WHERE url='facebook.com/diesocialisten'
```

Feldname	Beschreibung	Typ	Indiziert
url	URL, auf die sich die Linkstatistiken beziehen	string	ja
normalized_url	normalisierte URL – vereinheitlicht unterschiedliche Schreibweisen, z. B. abschließende Schrägstriche etc.)	string	nein
share_count	Gibt an, wie oft diese URL auf Facebook geteilt wurde.	int	nein
like_count	Gibt an, wie oft Benutzer Gefällt mir auf dieser URL angeklickt haben.	int	nein
comment_count	Anzahl der Kommentare, die zu dieser URL auf Facebook veröffentlicht wurden	int	nein
total_count	Summe aus share_count, like_count und comment_count	int	nein
click_count	Gibt an, wie oft Benutzer auf Facebook auf die geteilte URL geklickt haben.	int	nein

Tabelle 5.25 Felder der Link-Stat-Tabelle

Feldname	Beschreibung	Typ	Indiziert
comments_fbid	Ist auf der URL das Facebook-Kommentar-Plugin integriert, enthält dieses Feld die ID, mit der die Kommentare aus der Tabelle comment gelesen werden können.	int	nein
commentsbox_count	Ist auf der URL das Facebook-Kommentar-Plugin integriert, enthält dieses Feld die Anzahl der veröffentlichten Kommentare.	int	nein

Tabelle 5.25 Felder der Link-Stat-Tabelle (Forts.)

5.2.20 Mailbox-Folder-Tabelle

Die FQL-Tabelle mailbox_folder bildet die unterschiedlichen Ordner des Facebook-Nachrichten-Postfachs des aktuellen Benutzers ab. In der Graph API werden diese durch die Verknüpfungen inbox, outbox und updates des user-Objekts repräsentiert (siehe Abschnitt 3.2.1, »Das User-Objekt«). Die Nachrichten-Ordner können nur für den aktuellen Benutzer gelesen werden und erfordern ein Access Token mit der read_mailbox-Berechtigung.

Beispiel – Lesen aller Ordner des aktuellen Benutzers:

```
SELECT name, unread_count, total_count
FROM mailbox_folder
WHERE viewer_id = me()
```

Feldname	Beschreibung	Typ	Indiziert
folder_id	Numerische ID des Ordners. Diese ID ist für jeden Benutzer gleich und kann die Werte 0 (Inbox), 1 (Outbox) oder 4 (Updates) annehmen.	int	ja
viewer_id	ID des Benutzers, dessen Nachrichten-Postfach abgefragt wird	int	ja
name	Name des Ordners (Inbox)	string	nein
unread_count	Anzahl der ungelesenen Nachrichten	int	nein
total_count	Anzahl der Nachrichten insgesamt	int	nein

Tabelle 5.26 Felder der Mailbox-Folder-Tabelle

5.2.21 Message-Tabelle

Die FQL-Tabelle `message` enthält die einzelnen Nachrichten im Postfach des aktuellen Benutzers. Die Nachrichten können nur für den aktuellen Benutzer gelesen werden und erfordern ein Access Token mit der `read_mailbox`-Berechtigung. Die Tabelle entspricht dem in Abschnitt 3.2.20, »Das Nachrichtenobjekt«, beschriebenen Graph-Objekt `message`.

Beispiel – Lesen aller Nachrichten eines Threads:

```
SELECT message_id, body FROM message
WHERE thread_id=10150421384539627
```

Beispiel – Lesen einer bestimmten Nachricht:

```
SELECT message_id, thread_id, author_id, body, created_time
FROM message WHERE message_id='10150421384539627_0'
```

Feldname	Beschreibung	Typ	Indiziert
`message_id`	ID der Nachricht	`string`	ja
`thread_id`	ID des Threads	`int`	ja
`author_id`	Benutzer-ID des Senders der Nachricht	`int`	nein
`body`	Inhalt der Nachricht	`string`	nein
`created_time`	UNIX-Zeitstempel der Erstellung der Nachricht	UNIX-Zeitstempel	nein
`attachment`	Anhang der Nachricht	Array	nein
`viewer_id`	Benutzer-ID des Empfängers der Nachricht bzw. Besitzers des abgefragten Postfachs	`int`	nein

Tabelle 5.27 Felder der Message-Tabelle

5.2.22 Note-Tabelle

Die FQL-Tabelle `note` enthält auf der Pinnwand veröffentlichte Notizen und entspricht dem in Abschnitt 3.2.16, »Das Notizenobjekt«, beschriebenen Graph-Objekt. Während öffentlich von Seiten gepostete Notizen mit einem beliebigen Access Token gelesen werden können, ist für von Benutzern veröffentlichte Notizen die Berechtigung `user_notes` bzw. `friends_notes` notwendig.

Beispiel – Lesen aller Notizen des aktuellen Benutzers:

```
SELECT uid, note_id, title, content_html, content,
created_time, FROM note WHERE uid=me()
```

Beispiel – Lesen einer bestimmten Notiz:

```
SELECT uid, note_id, title, content_html, content,
created_time, FROM note WHERE note_id=122788341354
```


Feldname	Beschreibung	Typ	Indiziert
uid_id	Benutzer-ID des Erstellers der Notiz	int	ja
note_id	Objekt-ID der Notiz	int	ja
created_time	UNIX-Zeitstempel der Erstellung der Notiz	UNIX-Zeit-stempel	nein
updated_time	UNIX-Zeitstempel des letzten Updates der Notiz	UNIX-Zeit-stempel	nein
content	Inhalt der Notiz als Text	string	nein
content_html	Inhalt der Notiz als HTML-Text	string	nein
title	Titel der Notiz	string	nein

Tabelle 5.28 Felder der Note-Tabelle

5.2.23 Notification-Tabelle

Die FQL-Tabelle `notification` enthält die Benachrichtigungen des aktuellen Benutzers und entspricht damit der `notifications`-Verknüpfung des `user`-Objekts, wie in Abschnitt 3.2.1, »Das User-Objekt«, beschrieben. Das Lesen der Benachrichtigungen ist nur für den aktuellen Benutzer möglich und erfordert die erweiterte Berechtigung `manage_notifications`.

Beispiel – Abfrage der ungelesenen Benachrichtigungen des aktuellen Benutzers:

```
SELECT notification_id, sender_id, title_html, body_html, href
FROM notification
WHERE recipient_id=me() AND is_unread = 1 AND is_hidden = 0
```

Beispiel – Abfrage einer bestimmten Benachrichtigung – da die `notification_id` allein nicht eindeutig ist, muss die Abfrage gemeinsam mit der `recipient_id` erfolgen:

```
SELECT notification_id, sender_id, title_html, body_html, href
FROM notification
WHERE notification_id= 156574449 and recipient_id=me()
```

Feldname	Beschreibung	Typ	Indiziert
notification_id	ID der Benachrichtigung. Diese ID ist allein nicht eindeutig und muss daher immer gemeinsam mit dem Feld recipient_id abgefragt werden.	int	nein
sender_id	Benutzer-ID des Absenders	int	nein
recipient_id	Benutzer-ID des Empfängers, also des aktuellen Benutzers	int	ja
created_time	UNIX-Zeitstempel der Erstellung der Benachrichtigung	UNIX-Zeitstempel	nein
updated_time	UNIX-Zeitstempel der Erstellung der Benachrichtigung bzw. des Zeitpunktes, zu dem sie vom Empfänger versteckt oder sichtbar gemacht wurde	UNIX-Zeitstempel	nein
title_html	Benachrichtungstext als HTML-Text	string	nein
title_text	Benachrichtungstext	string	nein
body_html	zusätzlicher Inhalt als HTML-Text	string	nein
body_text	zusätzlicher Inhalt als Text	string	nein
href	URL, die mit der Benachrichtigung verknüpft ist	string	nein
app_id	ID der Anwendung, über die die Benachrichtigung erzeugt wurde	string	nein
is_unread	Zeigt an, ob die Benachrichtigung bereits gelesen wurde.	boolean	nein
is_hidden	Zeigt an, ob die Benachrichtigung vom Empfänger versteckt wurde.	boolean	nein
object_id	Objekt-ID der Benachrichtigung	int	nein

Tabelle 5.29 Felder der Notification-Tabelle

Feldname	Beschreibung	Typ	Indiziert
object_type	Typ des Objekts, auf das sich die Nachricht bezieht (photo, friend, stream ...)	string	nein
icon_url	URL zum Icon der Benachrichtigung (16 x 16 Pixel)	string	nein

Tabelle 5.29 Felder der Notification-Tabelle (Forts.)

5.2.24 Object-URL-Tabelle

Die FQL-Tabelle object_url bildet die Zuordnung von externen URLs zu Objekten im sozialen Graphen von Facebook ab und ist damit äquivalent zum in Abschnitt 3.2.6, »Das Domain-Objekt«, beschriebenen Objekt. Der Zugriff kann ohne Access Token erfolgen.

Beispiel – Abfrage der Objekt-ID anhand einer externen URL:

```
SELECT url, id, type, site FROM object_url
WHERE url = 'http://die.socialisten.at/2011/10/last-fm-scrobbler-for-facebook-open-graph-timeline/'
```

Beispiel – Abfrage der Objektinformationen anhand einer bekannten ID:

```
SELECT url, id, type, site FROM object_url
WHERE id = 230200637040140
```

Feldname	Beschreibung	Typ	Indiziert
url	externe URL des Objekts	string	ja
id	Objekt-ID des Objekts	int	ja
type	Typ des Objekts (page oder andere Open-Graph-Typen)	string	nein
site	normalisierte Domain der externen URL	string	nein

Tabelle 5.30 Felder der Object-URL-Tabelle

5.2.25 Page-Tabellen

Die FQL-Tabelle page bildet Seitenobjekte des sozialen Graphen ab, wie in Abschnitt 3.2.2, »Das Seitenobjekt«, beschrieben. Öffentliche Seiten können ohne Access Token, nicht-öffentliche oder geographisch eingeschränkt verfügbare Seiten mit dem Access Token eines berechtigten Benutzers eingelesen werden.

Beispiel – Lesen der Anzahl der Fans einer der ID nach bekannten Seite:

```
SELECT username, name, fan_count FROM page
WHERE page_id = 129842310398883
```

Beispiel – Lesen der Anzahl der Fans einer dem eindeutigen Namen nach bekannten Seite:

```
SELECT name, fan_count FROM page
WHERE username = 'diesocialisten'
```

Feldname	Beschreibung	Typ	Indiziert
page_id	Objekt-ID der Seite	int	ja
name	Name der Seite	string	ja
username	eindeutiger Kurzname der Seite	string	ja
description	Beschreibung der Seite	string	nein
categories	Seitenkategorien	Array	nein
is_community_page	Zeigt an, ob eine Seite eine Community-Seite ist.	boolean	nein
pic_small	URL zum kleinen Profilbild der Seite (maximal 50 x 150 Pixel)	string	nein
pic_big	URL zum großen Profilbild der Seite (maximal 200 x 600 Pixel)	string	nein
pic_square	URL zum quadratischen Profilbild der Seite (50 x 50 Pixel)	string	nein
pic	URL zum mittelgroßen Profilbild der Seite (maximal 100 x 300 Pixel)	string	nein
pic_large	URL zum größten Profilbild der Seite (maximal 396 x 1.188 Pixel)	string	nein

Tabelle 5.31 Felder der Page-Tabelle

Feldname	Beschreibung	Typ	Indiziert
page_url	URL zum Profil der Seite	string	nein
fan_count	Anzahl der Fans der Seite	int	nein
type	Typ der Seite	string	nein
website	URL zur Website der Seite	string	nein
has_added_app	Zeigt an, ob die Seite die Anwendung, von der die FQL-Abfrage gestellt wurde, als Reiter installiert hat.	boolean	nein
general_info	allgemeine Informationen zur Seite	string	nein
can_post	Zeigt an, ob der aktuelle Benutzer auf der Pinnwand der Seite posten darf.	boolean	nein
checkins	Anzahl der Checkins (nur für Seiten vom Typ PLACE)	int	nein
founded	nur für Seiten vom Typ COMPANY	string	nein
company_overview	nur für Seiten vom Typ COMPANY	string	nein
mission	nur für Seiten vom Typ COMPANY	string	nein
products	nur für Seiten vom Typ COMPANY	string	nein
location	nur für Seiten vom Typ PLACE	Array	nein
parking	nur für Seiten vom Typ BUSINESS oder PLACE	Array	nein
hours	nur für Seiten vom Typ BUSINESS oder PLACE	Array	nein
pharma_safety_info	nur für Seiten vom Typ PHARMACEUTICAL	string	nein
public_transit	nur für Seiten vom Typ RESTAURANT oder NIGHTLIFE	string	nein
attire	nur für Seiten vom Typ RESTAURANT oder NIGHTLIFE	string	nein

Tabelle 5.31 Felder der Page-Tabelle (Forts.)

Feldname	Beschreibung	Typ	Indiziert
payment_options	nur für Seiten vom Typ RESTAURANT oder NIGHTLIFE	string	nein
culinary_team	nur für Seiten vom Typ RESTAURANT oder NIGHTLIFE	string	nein
general_manager	nur für Seiten vom Typ RESTAURANT oder NIGHTLIFE	string	nein
price_range	nur für Seiten vom Typ RESTAURANT oder NIGHTLIFE	string	nein
restaurant_services	nur für Seiten vom Typ RESTAURANT oder NIGHTLIFE	Array	nein
restaurant_specialities	nur für Seiten vom Typ RESTAURANT oder NIGHTLIFE	Array	nein
phone	Telefonnumer	string	nein
release_date	nur für Seiten vom Typ MOVIE	string	nein
genre	nur für Seiten vom Typ MOVIE	string	nein
starring	nur für Seiten vom Typ MOVIE	string	nein
screenplay_by	nur für Seiten vom Typ MOVIE	string	nein
directed_by	nur für Seiten vom Typ MOVIE	string	nein
produced_by	nur für Seiten vom Typ MOVIE	string	nein
studio	nur für Seiten vom Typ MOVIE	string	nein
awards	nur für Seiten vom Typ MOVIE	string	nein
plot_ouline	nur für Seiten vom Typ MOVIE	string	nein
season	nur für Seiten vom Typ TV SHOW	string	nein
network	nur für Seiten vom Typ TV SHOW	string	nein
schedule	nur für Seiten vom Typ TV SHOW	string	nein
written_by	nur für Seiten vom Typ TV SHOW	string	nein
band_members	nur für Seiten vom Typ MUSICIAN/BAND	string	nein

Tabelle 5.31 Felder der Page-Tabelle (Forts.)

Feldname	Beschreibung	Typ	Indiziert
hometown	nur für Seiten vom Typ MUSICIAN/BAND	string	nein
current_location	nur für Seiten vom Typ MUSICIAN/BAND	string	nein
record_label	nur für Seiten vom Typ MUSICIAN/BAND	string	nein
booking_agent	nur für Seiten vom Typ MUSICIAN/BAND	string	nein
press_contact	nur für Seiten vom Typ MUSICIAN/BAND	string	nein
artists_we_like	nur für Seiten vom Typ MUSICIAN/BAND	string	nein
influences	nur für Seiten vom Typ MUSICIAN/BAND	string	nein
band_interests	Nur für Seiten vom Typ MUSICIAN/BAND	string	nein
bio	nur für Seiten vom Typ MUSICIAN/BAND	string	nein
affiliation	nur für Seiten vom Typ PEOPLE	string	nein
birthday	nur für Seiten vom Typ PEOPLE	string	nein
personal_info	nur für Seiten vom Typ PEOPLE	string	nein
personal_interests	nur für Seiten vom Typ PEOPLE	string	nein
built	nur für Seiten vom Typ CARS	string	nein
features	nur für Seiten vom Typ CARS	string	nein
mpg	nur für Seiten vom Typ CARS	string	nein

Tabelle 5.31 Felder der Page-Tabelle (Forts.)

Page-Admin-Tabelle

Die FQL-Tabelle page_admin enthält alle Benutzer, die Administrationsrechte für eine Seite besitzen, und entspricht damit der in Abschnitt 3.2.2, »Das Seitenobjekt«,

beschriebenen admins-Verknüpfung. Die Berechtigung manage_pages ist notwendig, um die administrierten Seiten des aktuellen Benutzers auszulesen.

Beispiel – Lesen aller Seiten, für die der aktuelle Benutzer Administrationsrechte besitzt:

```
SELECT page_id, type from page_admin WHERE uid=me()
```

Beispiel – Lesen aller Administratoren einer bestimmten Seite:

```
SELECT uid, type from page_admin WHERE page_id=129842310398883
```

Feldname	Beschreibung	Typ	Indiziert
uid	Benutzer-ID des Administrators	int	ja
page_ud	ID der Seite	int	ja
type	Typ der Seite	string	nein

Tabelle 5.32 Felder der Page-Admin-Tabelle

Page-Blocked-User-Tabelle

Die FQL-Tabelle page_blocked_users enthält alle durch Administratoren gesperrten Benutzer einer Seite und entspricht damit der in Abschnitt 3.2.2, »Das Seitenobjekt«, beschriebenen blocked-Verknüpfung. Die Berechtigung manage_pages ist notwendig, um die gesperrten Benutzer einer Seite auszulesen.

Beispiel – Lesen aller gesperrten Benutzer einer bestimmten Seite:

```
SELECT uid FROM page_blocked_user WHERE page_id = 129842310398883
```

Feldname	Beschreibung	Typ	Indiziert
page_id	ID der Seite	int	ja
uid	ID des gesperrten Benutzers	int	nein

Tabelle 5.33 Felder der Page-Blocked-User-Tabelle

Page-Fan-Tabelle

Die FQL-Tabelle page_fans bildet die Gefällt-mir-Beziehung zwischen Benutzern und Seiten ab und entspricht damit der in Abschnitt 3.2.1, »Das User-Objekt«, beschriebenen likes-Verknüpfung. Die Abfrage erfolgt bei öffentlichen Beziehungen

mit einem beliebigen Access Token, bei nicht-öffentlichen GEFÄLLT MIR mit der erweiterten Berechtigung `user_likes` bzw. `friend_likes`.

Hinweis: Da das einzige indizierte Feld der Tabelle `page_fans` die Spalte `uid` ist, kann sie lediglich dazu benutzt werden, alle oder eine bestimmte GEFÄLLT-MIR-Beziehung eines bestimmten Benutzers auszulesen. Ein Auslesen aller GEFÄLLT-MIR-Beziehungen einer einzelnen Seite ist hingegen nicht möglich!

Beispiel – Lesen aller GEFÄLLT-MIR-Beziehungen des aktuellen Benutzers:

```
SELECT page_id, profile_section FROM page_fan WHERE uid=me()
```

Beispiel – Prüfen, ob ein bestimmter Benutzer Fan einer bestimmten Seite ist:

```
SELECT page_id, profile_section FROM page_fan
WHERE page_id=129842310398883 AND uid=633616172
```

Feldname	Beschreibung	Typ	Indiziert
`uid`	Benutzer-ID	`int`	ja
`page_ID`	ID der Seite	`int`	nein
`type`	Typ der Seite	`string`	nein
`profile_section`	Gibt an, in welchem Profilabschnitt des Benutzers die Seite angezeigt wird (`music`, `activities` ...).	`string`	nein
`created_time`	UNIX-Zeitstempel der Herstellung der GEFÄLLT-MIR-Beziehung	UNIX-Zeitstempel	nein

Tabelle 5.34 Felder der Page-Fan-Tabelle

5.2.26 Permission-Tabelle

Die FQL-Tabelle `permission` bildet die Berechtigungen ab, die ein Benutzer der aktuellen Anwendung erteilt hat. Damit entspricht sie der in Abschnitt 3.2.1, »Das User-Objekt«, beschriebenen `permissions`-Verknüpfung. Erteilte Berechtigungen können immer nur für die eigene Anwendung und den aktuellen Benutzer ermittelt werden. Dafür ist ein Access Token des Benutzers oder der Anwendung notwendig.

Beispiel – Prüfen der Berechtigungen `manage_pages` und `publish_stream` für den aktuellen Benutzer:

```
SELECT manage_pages, publish_stream FROM permissions
WHERE uid = me()
```

Hinweis: Die Tabelle `permission` besitzt eine virtuelle Spalte je erteilter Berechtigung. Die Spalten sind dabei analog, wie in Abschnitt 2.6, »Erweiterte Zugriffsrechte (»Extended Permissions«)«, beschrieben, benannt und werden daher an dieser Stelle nicht vollständig wiederholt.

Feldname	Beschreibung	Typ	Indiziert
`uid`	Benutzer-ID	`int`	ja
`user_relationship`	Zeigt an, ob die Berechtigung erteilt wurde.	`boolean`	nein
`user_groups`	Zeigt an, ob die Berechtigung erteilt wurde.	`boolean`	nein
`user_events`	Zeigt an, ob die Berechtigung erteilt wurde.	`boolean`	nein
...	...	...	...
`PERMISSION_NAME`	Jede von der Facebook-Plattform unterstützte Berechtigung wird durch eine entsprechende FQL-Spalte repräsentiert.	`boolean`	nein

Tabelle 5.35 Felder der Permission-Tabelle

5.2.27 Fototabellen

In der FQL-Tabelle `photo` werden alle auf der Facebook-Plattform gespeicherten Fotos abgebildet. Das entsprechende Graph-Objekt wurde in Abschnitt 3.2.7, »Das Fotoobjekt«, beschrieben. Zum Zugriff auf die öffentlichen Fotos einer Seite ist ein beliebiges Access Token ausreichend, während private Fotos von Benutzern ein Token mit der Berechtigung `user_photos` oder `friends_photos` erfordern.

Beispiel – Lesen aller Fotos eines bestimmten Albums:

```
SELECT pid, src FROM photo WHERE aid='20531316728_324257'
```

Beispiel – Lesen eines bestimmten Fotos:

```
SELECT pid, src FROM photo WHERE pid='20531316728_7844072'
```

Feldname	Beschreibung	Typ	Indiziert
pid	ID des Fotos	string	ja
aid	ID des Albums, in dem das Foto gespeichert ist	string	ja
owner	Benutzer- oder Seiten-ID des Erstellers des Fotos	int	nein
src_small	URL zu Thumbnail des Fotos (maximal 75 x 225 Pixel)	string	nein
src_small_width	Breite des Thumbnails in Pixeln	int	nein
src_small_height	Höhe des Thumbnails in Pixeln	int	nein
src_big	URL zum Foto in höchster verfügbarer Auflösung (maximal 720 x 720 Pixel, ab 1. März 2012: maximal 960 x 960 Pixel)	string	nein
src_big_width	Breite des voll aufgelösten Fotos	int	nein
src_big_height	Höhe des voll aufgelösten Fotos	int	nein
src	URL zur Album-Ansicht des Fotos (maximal 130 x 130 Pixel)	string	nein
src_width	Breite des Fotos in der Album-Ansicht	int	nein
src_width	Höhe des Fotos in der Album-Ansicht	int	nein
link	URL zum Foto	string	nein
caption	Beschriftung des Fotos	string	nein
created	UNIX-Zeitstempel der Veröffentlichung des Fotos	UNIX-Zeitstempel	nein
modified	UNIX-Zeitstempel der letzten Änderung des Fotos	UNIX-Zeitstempel	nein
position	Position des Fotos im Album	int	nein
object_id	Objekt-ID des Fotos	int	ja

Tabelle 5.36 Felder der Fototabelle

Feldname	Beschreibung	Typ	Indiziert
album_object_id	Objekt-ID des Albums, in dem das Foto gespeichert ist	int	ja
images	Enthält alle URLs sowie Breiten- und Höhenangaben zu allen verfügbaren Auflösungen des Fotos.	Array	nein

Tabelle 5.36 Felder der Fototabelle (Forts.)

Photo-Tag-Tabelle

Die FQL-Tabelle `photo_tag` enthält Fotomarkierungen (*Tags*) und entspricht damit der `tags`-Verknüpfung des `photo`-Objekts, wie in Abschnitt 3.2.7, »Das Fotoobjekt«, beschrieben. Zum Lesen von Fotomarkierungen ist die Berechtigung `user_photos` bzw. `friends_photos` erforderlich.

Beispiel – Lesen aller Fotomarkierungen des aktuellen Benutzers:

```
SELECT pid, object_id FROM photo_tag WHERE subject = me()
```

Beispiel – Lesen aller auf einem bestimmten Foto markierten Benutzer:

```
SELECT pid, object_id, subject FROM photo_tag
WHERE pid = 3129746340803180882
```

Neben Fotomarkierungen wird `photo_tag` auch genutzt, um die Fotos, die mit einer Gruppe oder einem Event verknüpft sind, auszulesen. Dazu muss als `subject` lediglich die ID der Gruppe oder des Events abgefragt werden.

Beispiel – Lesen aller Fotos, die mit einer bestimmten Gruppe verknüpft sind:

```
SELECT pid, object_id FROM photo_tag
WHERE subject = 269627173053176
```

Feldname	Beschreibung	Typ	Indiziert
pid	ID des Fotos	string	ja
subject	ID des markierten Benutzers bzw. der Veranstaltung oder Gruppe	int	ja
object_id	Objekt-ID des Fotos	int	nein
text	Beschriftung der Markierung	string	nein

Tabelle 5.37 Felder der Photo-Tag-Tabelle

Feldname	Beschreibung	Typ	Indiziert
xcoord	X-Koordinate der Markierung als prozentueller Abstand vom linken Rand	float	nein
ycoord	Y-Koordinate der Markierung als prozentueller Abstand vom linken Rand	float	nein

Tabelle 5.37 Felder der Photo-Tag-Tabelle (Forts.)

5.2.28 Place-Tabelle

Die FQL-Tabelle `place` bildet `page`-Objekte im sozialen Graphen ab, die eine geographische Verortung über die Adresse erlauben und somit als »Ort« für Checkins zur Verfügung stehen. Da Orte im sozialen Graphen mittlerweile immer als Seitenobjekt gespeichert werden, existiert für sie kein eigener Objekttyp.

Beispiel – Lesen der Detailinformationen eines Ortes:

```
SELECT name, description, geometry,
latitude, longitude, checkin_count
FROM place WHERE page_id = 163483520335945
```

Feldname	Beschreibung	Typ	Indiziert
page_id	ID des Ortes bzw. der Seite	int	ja
name	Name des Ortes	string	nein
description	Beschreibung des Ortes	string	nein
geometry	Array, das in den Feldern `type` und `coordinates` die geographische Verortung des Ortes enthält	Array	nein
latitude	Breitengrad	float	nein
longitude	Längengrad	float	nein
checkin_count	Anzahl der Checkins	int	nein
display_subtext	Kurzinfo zum Ort (enthält den Typ des Ortes und die Anzahl der Checkins)	string	nein

Tabelle 5.38 Felder der Place-Tabelle

5.2.29 Privacy-Tabellen

Mithilfe der FQL-Tabelle `privacy` können die Privatsphäre-Einstellungen ermittelt werden, mit denen der aktuelle Benutzer Videos, Fotos, Alben, Notizen oder Links veröffentlicht hat. Die Privatsphäre-Einstellungen dieser Objekte können nur für den aktuellen Benutzer ermittelt werden, dessen Access Token beim Zugriff notwendig ist.

Beispiel – Lesen der Privatsphäre-Einstellungen eines bestimmten Objekts:

```
SELECT value, description FROM privacy
WHERE id = 10150146071791729
```

Feldname	Beschreibung	Typ	Indiziert
`id`	Objekt-ID des Videos, Fotos, Albums, der Notiz oder des Links	`int`	ja
`object_id`	Alias für die Spalte `id`	`int`	ja
`value`	Privatsphäre-Einstellung – `EVERYONE`, `CUSTOM`, `ALL_FRIENDS`, `NETWORKS_FRIENDS` oder `FRIENDS_OF_FRIENDS`	`string`	nein
`description`	Textbeschreibung der Privatsphäre-Einstellungen	`string`	nein
`allow`	kommaseparierte Liste an numerischen IDs von Benutzern oder Freundeslisten, denen Zugriff auf das Objekt gewährt wurde	`string`	nein
`deny`	kommaseparierte Liste an numerischen IDs von Benutzern oder Freundeslisten, denen Zugriff auf das Objekt verwehrt wurde	`string`	nein
`owner_id`	Benutzer-ID des Besitzers des Objekts	`int`	nein
`networks`	ID des Benutzernetzwerks, das Zugriff auf das Objekt hat	`int`	nein
`friends`	Zeigt an, welchen Freunden Zugriff auf das Objekt gewährt wurde – `EVERYONE`, `NETWORKS_FRIENDS`, `FRIENDS_OF_FRIENDS`, `ALL_FRIENDS`, `SOME_FRIENDS`, `SELF`, `NO_FRIENDS`.	`string`	nein

Tabelle 5.39 Felder der Privacy-Tabelle

Privacy-Setting-Tabelle

Die FQL-Tabelle `privacy_setting` gibt Auskunft über die Standard-Privatsphäre-Einstellungen, die der aktuelle Benutzer für die eigenen Anwendung gesetzt hat. Diese Einstellungen können nur für den aktuellen Benutzer ermittelt werden, ein entsprechendes Access Token ist beim Zugriff notwendig.

Hinweis: Derzeit kennt Facebook nur eine anwendungsbezogene Einstellung namens `default_stream_privacy`. Diese regelt die Privatsphäre-Einstellungen aller Veröffentlichungen auf der Pinnwand des Benutzers, die durch die Anwendung vorgenommen werden.

Beispiel – Lesen der Standard-Privatsphäre-Einstellungen für die aktuelle Anwendung:

```
SELECT name, value, description, allow, deny, networks, friends
FROM privacy_setting WHERE name = 'default_stream_privacy'
```

Feldname	Beschreibung	Typ	Indiziert
`Name`	Name der Privatsphäre-Einstellungen – derzeit immer `default_stream_privacy`	`string`	ja
`value`	Standard-Privatsphäre-Einstellung – `EVERYONE`, `CUSTOM`, `ALL_FRIENDS`, `NETWORKS_FRIENDS` oder `FRIENDS_OF_FRIENDS`	`string`	nein
`description`	Textbeschreibung der Standard-Privatsphäre-Einstellungen	`string`	nein
`allow`	kommaseparierte Liste an numerischen IDs von Benutzern oder Freundeslisten, denen Zugriff auf Objekte gewährt wird	`string`	nein
`deny`	kommaseparierte Liste an numerischen IDs von Benutzern oder Freundeslisten, denen Zugriff auf Objekte verwehrt wird	`string`	nein

Tabelle 5.40 Felder der Privacy-Setting-Tabelle

Feldname	Beschreibung	Typ	Indiziert
networks	ID des Benutzernetzwerks, das Zugriff auf Objekte hat	int	nein
friends	Zeigt an, welchen Freunden Zugriff auf veröffentlichte Objekte gewährt wird: EVERYONE, NETWORKS_FRIENDS, FRIENDS_OF_FRIENDS, ALL_FRIENDS, SOME_FRIENDS, SELF, NO_FRIENDS	string	nein

Tabelle 5.40 Felder der Privacy-Setting-Tabelle (Forts.)

5.2.30 Profile-Tabelle

Die FQL-Tabelle `profile` bildet Profildaten von unterschiedlichen Objekten wie Benutzern, Gruppen, Seiten, Veranstaltungen und Anwendungen zusammengefasst in einer Tabelle ab. Damit entspricht sie keinem eigenen Objekttypen im Graphen von Facebook, sondern nimmt eine Art übergeordnete Rolle ein. Bei der Abfrage stellen sich je nach Objekttyp folgende Voraussetzungen an das verwendete Access Token:

- Benutzerobjekte können ohne Access Token abgefragt werden, mit Ausnahme des Feldes `url`, das ein beliebiges Token erfordert.
- Öffentliche Gruppen können ohne Access Token, private Gruppen mit dem Token eines Mitglieds abgefragt werden.
- Für die Abfrage von Seiten ist kein Access Token notwendig.
- Für die Abfrage von Anwendungen ist kein Access Token notwendig.
- Öffentliche Veranstaltungen können ohne Access Token, private Veranstaltungen mit dem Token eines eingeladenen Benutzers abgefragt werden.

Beispiel – Abfrage des Profils einer Seite anhand ihres eindeutigen Namens:

```
SELECT id, name, url, pic FROM profile
WHERE username = 'diesocialisten'
```

Feldname	Beschreibung	Typ	Indiziert
id	Objekt-ID von Benutzer, Seite, Anwendung, Veranstaltung oder Gruppe	int	ja
can_post	Zeigt an, ob der aktuelle Benutzer auf der Pinnwand des abgefragten Objekts posten darf (nur unter Verwendung eines Access Tokens gesetzt).	boolean	nein
name	Name des abgefragten Objekts	string	nein
pic_small	URL zum kleinen Profilbild des Objekts (maximal 50 x 150 Pixel)	string	nein
pic_big	URL zum großen Profilbild des Objekts (maximal 200 x 600 Pixel)	string	nein
pic_square	URL zum quadratischen Profilbild des Objekts (50 x 50 Pixel)	string	nein
pic	URL zum mittelgroßen Profilbild des Objekts (maximal 100 x 300 Pixel)	string	nein
pic_crop	URL zum großen quadratischen Profilbild des Objekts (mindestens 100 x 100 Pixel)	string	nein
type	Typ des Objekts (user, group, application, event oder page)	string	nein

Tabelle 5.41 Felder der Profile-Tabelle

5.2.31 Question-Tabellen

In der FQL-Tabelle question werden auf Pinnwänden veröffentlichte Fragen gespeichert. Die Tabelle entspricht damit dem in Abschnitt 3.2.12, »Das Fragenobjekt«, beschriebenen Objekt des sozialen Graphen. Der Zugriff auf Fragen erfordert in jedem Fall ein Access Token mit der Berechtigung user_questions bzw. friends_questions.

Beispiel – Lesen aller gestellten Fragen des aktuellen Benutzers:

```
SELECT id, question FROM question WHERE owner=me()
```

Feldname	Beschreibung	Typ	Indiziert
id	ID der Frage	int	ja
owner	ID des Benutzers, der die Frage veröffentlicht hat	int	ja
question	Text der Frage	string	nein
created_time	UNIX-Zeitstempel der Erstellung der Frage	UNIX-Zeitstempel	nein
updated_time	UNIX-Zeitstempel des letzten Updates der Frage	UNIX-Zeitstempel	nein

Tabelle 5.42 Felder der Question-Tabelle

Question-Option-Tabelle

Während `question` nur den Fragetext enthält, müssen die einzelnen Antwortoptionen aus der FQL-Tabelle `question_option` abgefragt werden. Auch hierzu ist ein Access Token mit der Berechtigung `user_questions` bzw. `friends_questions` erforderlich.

Beispiel – Lesen aller Antwortoptionen einer bestimmten Frage:

```
SELECT id, name FROM question_option
WHERE question_id = 326069260742300
```

Feldname	Beschreibung	Typ	Indiziert
id	ID der Antwortoption	int	ja
question_id	ID der zugehörigen Frage	int	ja
name	Text der Antwortoption	string	nein
votes	Anzahl der Stimmen	int	nein
object_id	optionale ID einer mit der Antwortoption verknüpften Seite	int	nein
owner	ID des Benutzers, der die Antwortoption veröffentlicht hat	int	nein
created_time	UNIX-Zeitstempel der Erstellung der Antwortoption	UNIX-Zeitstempel	nein

Tabelle 5.43 Felder der Question-Option-Tabelle

Question-Option-Votes-Tabelle

Die einzelnen Stimmen, die eine Antwortoption in einer Frage erhalten hat, können über die Tabelle `question_option_votes` abgefragt werden. Auch hierzu ist ein Access Token mit der Berechtigung `user_questions` bzw. `friends_questions` erforderlich.

Beispiel – Lesen aller Stimmen einer bestimmten Antwortoption:

```
SELECT voter_id FROM question_option_votes
WHERE option_id = 201037246642437
```

Feldname	Beschreibung	Typ	Indiziert
`option_id`	ID der Antwortoption	`int`	ja
`voter_id`	ID des Benutzers, der für diese Option gestimmt hat	`int`	nein

Tabelle 5.44 Felder der Question-Option-Votes-Tabelle

5.2.32 Review-Tabelle

Die FQL-Tabelle `review` speichert Benutzer-Reviews zu Plattform-Anwendungen und entspricht dem in Abschnitt 3.2.17, »Das Review-Objekt«, beschriebenen Objekttyp. Da Reviews öffentlich sind, ist zum Zugriff kein Access Token notwendig.

Beispiel – Lesen aller Reviews zu einer bestimmten Anwendung:

```
SELECT review_id, message, rating FROM review
WHERE reviewee_id = 32788395891
```

Feldname	Beschreibung	Typ	Indiziert
`reviewee_id`	ID der bewerteten Anwendung	`int`	ja
`reviewer_id`	ID des Benutzers, der die Bewertung veröffentlicht hat	`int`	ja
`message`	Text der Bewertung	`string`	nein
`created_time`	UNIX-Zeitstempel der Erstellung der Bewertung	UNIX-Zeit-stempel	nein
`rating`	numerische Bewertung	`int`	nein

Tabelle 5.45 Felder der Review-Tabelle

5.2.33 Standard-Friend-Info-Tabelle

Die FQL-Tabelle `standard_friend_info` kann verwendet werden, um die Freundeslisten von Benutzern der aktuellen Anwendung abzufragen. Dazu ist kein Benutzer-Access-Token notwendig, sondern das entsprechenden Access Token der Anwendung. *Hinweis:* Da die Abfrage somit unabhängig von einem aktiv angemeldeten Benutzer erfolgt, ist die Information aus dieser Tabelle nur für den internen Gebrauch durch die Anwendung, nicht aber zur Anzeige in der Applikation gedacht.

Beispiel – Lesen aller Freunde eines bestimmten Anwendungsbenutzers:

```
SELECT uid2 FROM standard_friend_info WHERE uid1 = 633616172
```

Feldname	Beschreibung	Typ	Indiziert
`uid1`	ID des ersten abgefragten Anwendungsbenutzers	`int`	ja
`uid2`	ID des zweiten abgefragten Anwendungsbenutzers	`int`	ja

Tabelle 5.46 Felder der Standard-Friend-Info-Tabelle

5.2.34 Standard-User-Info-Tabelle

Ähnlich wie `standard_friend_info` kann die FQL-Tabelle `standard_user_info` verwendet werden, um Informationen über die Benutzer, die eine Anwendung bereits installiert haben, mit dem Anwendungs-Access-Token abzufragen. Auch hier gilt: Informationen aus dieser Tabelle sind nur für den internen Gebrauch der Anwendung und nicht zur Anzeige in der Applikation gedacht.

Beispiel – Lesen der Benutzerinfos eines bestimmten Anwendungsbenutzers:

```
SELECT name, username, sex, locale FROM standard_user_info
WHERE uid = 609190863
```

Feldname	Beschreibung	Typ	Indiziert
`uid`	ID des abgefragten Anwendungsbenutzers	`int`	ja
`name`	voller Name des Benutzers	`string`	ja
`username`	eindeutiger Benutzername	`string`	ja

Tabelle 5.47 Felder der Standard-User-Info-Tabelle

Feldname	Beschreibung	Typ	Indiziert
third_party_id	eine anonymisierte, eindeutige Benutzerkennung	string	ja
first_name	Vorname	string	nein
last_name	Nachname	string	nein
locale	Lokalisierungsinformationen des Benutzers	string enthält den ISO-Sprach- und Landescode, z. B. de_DE	nein
affiliations	Netzwerk-Zugehörigkeit des Benutzers	Array	nein
profile_url	URL zum Profil des Benutzers	string	nein
timezone	Zeitzone des Benutzers als Differenz zu UTC	string	nein
birthday	Geburtstag	string	nein
sex	Geschlecht	string	nein
proxied_email	Proxy-E-Mail-Adresse	string	nein
current_location	aktuelle Heimatstadt	string	nein
allowed_restrictions	kommaseparierte Liste der demographischen Zugriffseinschränkungen, die der Benutzer erfüllt – derzeit nur alcohol	string	nein

Tabelle 5.47 Felder der Standard-User-Info-Tabelle (Forts.)

5.2.35 Status-Tabelle

Die FQL-Tabelle `status` speichert auf der Pinnwand veröffentlichte Status-Updates von Benutzern und entspricht damit der `statuses`-Verknüpfung des in Abschnitt 3.2.1, »Das User-Objekt«, beschriebenen Objekts. Zum Lesen dieser Tabelle ist die erweiterte Berechtigung `read_stream` notwendig.

Beispiel – Lesen der Status-Updates des aktuellen Benutzers:

```
SELECT status_id, message FROM status WHERE uid=me()
```

Beispiel – Lesen eines bestimmten Status-Updates:

```
SELECT status_id, message FROM status
WHERE status_id = 10150453764222297
```

Feldname	Beschreibung	Typ	Indiziert
uid	ID des abgefragten Benutzers	int	ja
status_id	ID des Status-Updates	int	ja
time	UNIX-Zeitstempel der Veröffentlichung des Updates	UNIX-Zeitstempel	nein
source	ID der Anwendung, über die das Update veröffentlicht wurde	int	nein
message	Text des Status-Updates	string	nein

Tabelle 5.48 Felder der Status-Tabelle

5.2.36 Stream-Tabellen

Die FQL-Tabelle `stream` repräsentiert die veröffentlichten Einträge einer Pinnwand und entspricht damit der `feed`-Verknüpfung der Objekttypen `user`, `page`, `application`, `event` und `group`. Zum Lesen der `stream`-Tabelle ist ein Access Token mit `read_stream`-Berechtigung notwendig. Um auch das Statistik-Feld `impressions` von der Pinnwand einer Seite zu lesen, ist zusätzlich die `read_insights`-Berechtigung notwendig.

Hinweis: Abfragen auf die Tabelle `stream` liefern grundsätzlich nur maximal 50 Postings und Postings der letzten 30 Tage. Durch Aufnahme des Feldes `created_time` in die `WHERE`-Klausel können aber beliebige Zeiträume abgefragt werden.

Beispiel – Abfrage der Pinnwand-Einträge des aktuellen Benutzers:

```
SELECT post_id, actor_id, target_id, message
FROM stream WHERE source_id = me()
```

Feldname	Beschreibung	Typ	Indiziert
post_id	ID des Pinnwand-Postings	string	ja
viewer_id	ID des aktuellen Benutzers, der die Abfrage stellt	int	nein

Tabelle 5.49 Felder der Stream-Tabelle

Feldname	Beschreibung	Typ	Indiziert
app_id	ID der Anwendung, über die das Posting veröffentlicht wurde	int	nein
source_id	ID des Benutzers, der Seite, Anwendung, Veranstaltung oder Gruppe, deren Pinnwand abgefragt wird	int	ja
updated_time	UNIX-Zeitstempel des letzten Updates des Postings durch einen Kommentar	UNIX-Zeitstempel	nein
created_time	UNIX-Zeitstempel der Veröffentlichung des Postings	UNIX-Zeitstempel	nein
filter_key	Filterkriterium, mit dem die Pinnwand abgefragt werden soll (siehe Stream-Filter-Tabelle)	string	ja
attribution	Name der Anwendung, über die das Posting veröffentlicht wurde	string	nein
actor_id	ID des Benutzers, der Seite oder Anwendung, die das Posting veröffentlicht hat	int	nein
target_id	ID des Benutzers, der Seite, Gruppe, Anwendung oder Veranstaltung, an die das Posting gerichtet ist	int	nein
message	Text des Postings	string	nein
app_data	optional mit dem Posting verknüpfte, zusätzliche Daten	Array	nein
action_links	Enthält Text und URL der Action-Links.	Array	nein
attachment	Anhang zum Posting	Array	nein
comments	Enthält die Kommentare zum Posting als Objekt mit den Feldern can_remove (zeigt an, ob der aktuelle Benutzer Kommentare löschen kann), can_post (zeigt an, ob der aktuelle Benutzer Kommentare posten kann) und comment_list (Array mit den Kommentaren).	Array	nein

Tabelle 5.49 Felder der Stream-Tabelle (Forts.)

Feldname	Beschreibung	Typ	Indiziert
impressions	Gibt an, wie oft das Posting anderen Benutzern angezeigt wurde (benötigt read_insights-Berechtigung).	int	nein
likes	Enthält die Gefällt-mir-Einträge zum Posting als Objekt mit den Feldern href (URL zu einer Seite, die alle Gefällt mir anzeigt), count (Anzahl der Gefällt mir), sample (Array mit den Gefällt mir), friends (Array von Freunden des aktuellen Benutzers, die Gefällt mir angeklickt haben), user_likes (zeigt an, ob der aktuelle Benutzer auf Gefällt mir geklickt hat), can_like (zeigt an, ob bei dem Posting auf Gefällt mir geklickt werden kann).	Array	nein
privacy	Privatsphäre-Einstellungen des Postings	Array	nein
permalink	Link zum Posting	string	nein
xid	Pinnwände von Live-Stream-Plugins enthalten in diesem Feld die mit der Box verknüpfte xid.	string	nein
tagged_ids	IDs der Benutzer, Seiten, Veranstaltungen etc., die im Posting markiert wurden	Array	nein
message_tags	Array, das die Markierungen von Benutzern, Seiten, Veranstaltungen etc. mit ihrer genauen Position im Text angibt – name (Name der Markierung), offset (Anzahl der Zeichen ab Beginn), length (Länge der Markierung)	Array	nein
description	Text von implizit generierten Pinnwand-Postings	string	nein
description_tags	Array, das die Markierungen von Benutzern, Seiten, Veranstaltungen etc. mit ihrer genauen Position in implizit generierten Postings angibt – name (Name der Markierung), offset (Anzahl der Zeichen ab Beginn), length (Länge der Markierung)	Array	nein

Tabelle 5.49 Felder der Stream-Tabelle (Forts.)

Stream-Filter-Tabelle

Die `stream_filter`-Tabelle enthält alle Filtermöglichkeiten, die einem bestimmten Benutzer im Frontend von Facebook.com zur Verfügung stehen. Diese Filter können mit einem beliebigen Access Token, aber nur für den aktuellen Benutzer ausgelesen werden.

Beispiel – Lesen aller Filtermöglichkeiten des aktuellen Benutzers:

```
SELECT filter_key,name,type
FROM stream_filter WHERE uid=me()
```

Das Ergebnis dieser Abfrage sieht etwa so aus:

```
{
  "data": [
    {
      "filter_key": "nf",
      "name": "News Feed",
      "type": "newsfeed",
    },
    {
      "filter_key": "fl_10150456156070864",
      "name": "Familie",
      "type": "friendlist",
    },
    {
      "filter_key": "fl_10150387764450864",
      "name": "Arbeitskollegen",
      "type": "friendlist",
    },
    ...
```

Listing 5.2 JSON-Array mit Ergebnissen der Stream-Filter-Tabelle

Feldname	Beschreibung	Typ	Indiziert
`uid`	ID des Benutzers, dessen Filter abgefragt werden	`int`	ja
`filter_key`	ID des Filter	`string`	ja
`name`	Name des Filters, z. B. »News Feed« oder der Name einer Freundesliste	`string`	nein

Tabelle 5.50 Felder der Stream-Filter-Tabelle

Feldname	Beschreibung	Typ	Indiziert
rank	Reihenfolge des Filters im Frontend von Facebook.com	int	nein
icon_url	URL zum Icon des Filters. Entspricht bei Anwendungsfiltern dem Icon der Anwendung.	string	nein
is_visible	Zeigt an, ob der Filter auf der Startseite von Facebook.com sichtbar ist.	boolean	nein
type	Typ des Filters (application, newsfeed, friendlist, network, public_profiles)	string	nein
value	numerische ID des Filtertyps	int	nein

Tabelle 5.50 Felder der Stream-Filter-Tabelle (Forts.)

Beispiel – In Kombination mit der stream-Tabelle und deren filter_key-Spalte kann so etwa der Newsfeed (die Facebook-Startseite) eines bestimmten Benutzers ausgelesen werden:

```
SELECT post_id, actor_id, target_id, message
FROM stream WHERE filter_key = 'nf'
```

Beispiel – Auslesen aller Pinnwand-Einträge von Seiten, dessen Fan der aktuelle Benutzer ist (entspricht dem Newsfeed, eingeschränkt auf Seiten):

```
SELECT post_id, actor_id, target_id, message
FROM stream WHERE filter_key = 'pp'
```

Stream-Tag-Tabelle

Die FQL-Tabelle stream_tag bildet ab, wenn Benutzer oder Seiten in Wall-Postings markiert wurden. Während Markierungen in öffentlichen Postings mit einem beliebigen Access Token gelesen werden können, ist für private Postings die Berechtigung read_stream notwendig.

Beispiel – Lesen aller Postings, in denen der aktuelle Benutzer markiert wurde:

```
SELECT post_id,actor_id FROM stream_tag
WHERE target_id=me()
```

Beispiel – Lesen aller Postings, in denen der aktuelle Benutzer andere Seiten oder Benutzer markiert hat:

```
SELECT post_id,actor_id FROM stream_tag
WHERE actor_id=me()
```


Feldname	Beschreibung	Typ	Indiziert
post_id	ID des Postings	int	ja
actor_id	ID des Benutzers, der das Posting veröffentlicht hat	int	ja
target_id	ID des Benutzers oder der Seite, die im Posting markiert wurde	int	ja

Tabelle 5.51 Felder der Stream-Tag-Tabelle

5.2.37 Thread-Tabelle

Die FQL-Tabelle `thread` enthält die Konversationen im Postfach des aktuellen Benutzers. Die Konversationen können nur für den aktuellen Benutzer gelesen werden und erfordern ein Access Token mit der `read_mailbox`-Berechtigung. Die Tabelle entspricht dem in Abschnitt 3.2.19, »Das Konversationenobjekt«, beschriebenen Graph-Objekt `thread`.

Beispiel – Lesen aller Konversationen des Inbox-Ordners:

```
SELECT thread_id, subject, recipients FROM thread
WHERE folder_id = 0
```

Feldname	Beschreibung	Typ	Indiziert
thread_id	ID der Konversation	int	ja
folder_id	ID des Nachrichten-Ordners – 0 (Inbox), 1 (Outbox) oder 4 (Updates)	int	ja
subject	Betreff der Konversation	string	nein
recipients	Benutzer-IDs der Empfänger der Konversation	Array	nein

Tabelle 5.52 Felder der Thread-Tabelle

Feldname	Beschreibung	Typ	Indiziert
updated_time	UNIX-Zeitstempel des letzten Updates der Konversation	UNIX-Zeit-stempel	nein
parent_ message_id	Wenn diese Konversation von einer Nachricht abstammt, enthält dieses Feld die ID der Nachricht, ansonsten 0.	string	nein
parent_ thread_id	Wenn diese Konversation von einer anderen Konversation abstammt, enthält dieses Feld die ID der Konversation, ansonsten 0.	int	nein
message_count	Anzahl der Nachrichten in dieser Konversation	int	nein
snippet	kurze Textvorschau auf die neueste Nachricht der Konversation	string	nein
snippet_ author	Benutzer-ID des Erstellers der neuesten Nachricht	int	nein
object_id	ID des Objekts, das die Nachricht versandt hat. 0, wenn die Nachricht nicht von einem Objekt stammt.	int	nein
unread	Anzahl der ungelesenen Nachrichten in der Konversation	int	nein
viewer_id	Benutzer-ID des Besitzers der Konversation bzw. Besitzers des abgefragten Postfachs	int	nein

Tabelle 5.52 Felder der Thread-Tabelle (Forts.)

5.2.38 Translation-Tabelle

Die FQL-Tabelle `translation` enthält die Zeichenketten-Übersetzungen zur Internationalisierung der aktuellen Facebook-Anwendung. Sie kann mit dem Anwendungs-Access-Token gelesen werden.

Beispiel – Lesen aller Übersetzungen der aktuellen Anwendung für die Lokalisierung `de_DE`:

```
SELECT native_hash, pre_hash_string, native_string, description,
translation FROM translation WHERE locale = 'de_DE'
```

Feldname	Beschreibung	Typ	Indiziert
locale	Lokalisierung, die abgefragt werden soll	string	ja
native_hash	Interne Hash-Abbildung der übersetzten Zeichenkette. Der Hash wird mit dem Tiger128,3-Algorithmus aus dem Feld pre_hash_string errechnet.	string	ja
native_string	Die ursprüngliche Zeichenkette in der primären Sprache der Anwendung	string	nein
description	Beschreibungstext zur Zeichenkette. Diese Beschreibung wird vom Entwickler vergeben und soll das Zuordnen einer Zeichenkette erleichtern.	string	nein
translation	Übersetzung der in native_string enthaltenen Zeichenkette in der abgefragten locale	string	nein
approval_status	Zeigt an, ob die Übersetzung bestätigt wurde – auto-approved, approved oder unapproved.	string	nein
pre_hash_string	zusammengesetztes Feld aus native_string, gefolgt von drei Doppelpunkten, dem Feld description und einem abschließenden Doppelpunkt	string	nein
best_string	Enthält die übersetzte Zeichenkette, die Benutzern in der entsprechenden locale angezeigt wird.	string	nein

Tabelle 5.53 Felder der Translation-Tabelle

5.2.39 Unified-Message-Tabellen

Hinweis: Facebook bereitet derzeit eine Überarbeitung des Nachrichten-Postfachs unter dem Titel *Unified Messaging* vor. Die Tabellen unified_message, unified_thread, unified_thread_action und unified_thread_count werden künftig den Zugriff auf das neue Postfach erlauben. Zum aktuellen Zeitpunkt sind diese FQL-Tabellen aber nur für Benutzer zugänglich, die als Developer in der aktuellen Anwendung registriert sind. Bis zum öffentlichen Roll-out von Unified Messaging dürfen daher ausschließlich die bereits behandelten Tabellen message und thread zum Zugriff auf das Postfach des aktuellen Benutzers verwendet werden!

Der Zugriff auf die Tabellen des neuen Unified Messaging wird ausschließlich für den aktuellen Benutzer möglich sein und erfordert die Berechtigung `read_mailbox`.

Unified-Thread-Tabelle

Die FQL-Tabelle `unified_thread` erlaubt den Zugriff auf die Konversationen im Unified-Messaging-Postfach.

Feldname	Beschreibung	Typ	Indiziert
`action_id`	Numerische Versions-ID der Konversation – diese ID spiegelt den Status einer Konversation zu einem bestimmten Zeitpunkt wider und wird laufend erhöht.	`string`	nein
`archived`	Zeigt an, ob die Konversation archiviert wurde.	`boolean`	ja
`can_reply`	Zeigt an, ob der aktuelle Benutzer in dieser Konversation antworten kann.	`boolean`	nein
`folder`	Name des Ordners (`inbox`, `other` oder `spam`)	`string`	nein
`former_participants`	Enthält eine Liste aller Benutzer (Felder `name`, `id`, `email`), die die Konversation verlassen haben.	Array	nein
`has_attachments`	Zeigt an, ob die Konversation einen Anhang enthält.	`boolean`	nein
`is_subscribed`	Zeigt an, ob der aktuelle Benutzer die Konversation abonniert hat (nur enthalten bei Konversationen mit mehreren Teilnehmern).	`boolean`	nein
`last_visible_add_action_id`	Versions-ID, die einer der folgenden Aktionen entspricht: `new message in thread`, `new thread participant`, `participant left the thread`	`string`	nein
`name`	Name der Konversation	`string`	nein
`num_messages`	Anzahl der Nachrichten in dieser Konversation	`int`	nein
`num_unread`	Anzahl der ungelesenen Nachrichten in dieser Konversation	`int`	nein

Tabelle 5.54 Felder der Unified-Thread-Tabelle

Feldname	Beschreibung	Typ	Indiziert
object_participants	Liste der Konversationsteilnehmer, die eine Seite, eine Gruppe oder eine Veranstaltung sind	Array	nein
participants	Liste der Konversationsteilnehmer, die Benutzer sind (Felder name, email, id)	Array	nein
senders	Liste der Konversationsteilnehmer, die Benutzer sind und eine Nachricht in der Konversation gesendet haben (Felder name, email, id)	Array	nein
single_recipient	Bei Konversationen mit nur zwei Teilnehmern enthält dieses Feld die ID des anderen Teilnehmers.	int	ja
snippet	Kurztext zur Voransicht	string	nein
snippet_sender	Ersteller der Nachricht, die im Feld snippet angezeigt wird (Felder name, email, id)	Array	nein
thread_id	numerische ID der Konversation	string	ja
thread_participants	Liste der Konversationsteilnehmer, die Benutzer sind (Felder name, email, id)	Array	nein
timestamp	UNIX-Zeitstempel des letzten Updates der Konversation	UNIX-Zeitstempel	ja
unread	Zeigt an, ob die Konversation ungelesene Nachrichten enthält.	boolean	ja

Tabelle 5.54 Felder der Unified-Thread-Tabelle (Forts.)

Beispiel – Lesen aller Konversationen mit dem Tag inbox mithilfe der Funktion has_tags():

```
SELECT subject, snippet FROM unified_thread
WHERE has_tags('inbox')
```

Beispiel – Lesen aller Konversationen aus dem Ordner inbox:

```
SELECT subject, snippet FROM unified_thread
WHERE folder='inbox'
```

Beispiel – Lesen aller Konversationen, die ein bestimmtes Schlüsselwort enthalten – dies kann mit der Hilfsfunktion `contains()` erreicht werden:

```
SELECT subject, snippet FROM unified_thread
WHERE contains('schlüsselwort')
```

Unified-Thread-Count-Tabelle

Die FQL-Tabelle `unified_thread_count` gibt Auskunft über die Anzahl der Konversationen in den Ordnern des Unified-Messaging-Postfachs.

Beispiel – Abfrage der Anzahl der Konversationen in allen Ordnern:

```
SELECT folder, unread_count, unseen_count, total_threads
FROM unified_thread_count WHERE 1
```

Feldname	Beschreibung	Typ	Indiziert
`folder`	Name des Ordners (`inbox`, `other` oder `spam`)	`string`	ja
`unread_count`	Anzahl der ungelesenen Konversationen im abgefragten Ordner	`int`	ja
`unseen_count`	Anzahl der noch nie angezeigten Konversationen im abgefragten Ordner	`int`	ja
`last_action_id`	die höchste `action_id` im abgefragten Ordner	`int`	ja
`total_threads`	Gesamtanzahl der Konversationen im abgefragten Ordner	`int`	ja

Tabelle 5.55 Felder der Unified-Thread-Count-Tabelle

Unified-Thread-Action-Tabelle

Die FQL-Tabelle `unified_thread_action` bildet die Aktionen ADD PEOPLE und LEAVE CONVERSATION im neuen Unified-Messaging-Postfach ab.

Beispiel – Lesen aller Aktionen einer bestimmten Konversation:

```
SELECT actor, type, users FROM unified_thread_action
WHERE thread_id='t_id.2O4485922964776'
```

Feldname	Beschreibung	Typ	Indiziert
action_id	Numerische Versions-ID der Konversation – diese ID spiegelt den Status einer Konversation zu einem bestimmten Zeitpunkt wider und wird laufend erhöht.	string	nein
actor	Benutzer, der die Aktion durchgeführt hat (Felder name, id, email)	Array	nein
thread_id	ID der Konversation, in der die Aktion durchgeführt wurde	string	ja
timestamp	UNIX-Zeitstempel der Aktion	UNIX-Zeitstempel	nein
type	Zeigt an, ob ein Benutzer zur Konversation hinzugefügt (1) wurde oder ob ein Benutzer die Konversation verlassen hat (2).	int	nein
users	Liste der Benutzer, für die die Aktion durchgeführt wurde (Felder name, id, email)	Array	nein

Tabelle 5.56 Felder der Unified-Thread-Action-Tabelle

Unified-Message-Tabelle

Die FQL-Tabelle unified_messages enthält alle Nachrichten des neuen Unified-Messaging-Postfachs.

Beispiel – Lesen aller Nachrichten einer bestimmten Konversation:

```
SELECT subject, body FROM unified_message
WHERE thread_id='t_id.204485922964776'
```

Feldname	Beschreibung	Typ	Indiziert
message_id	ID der Nachricht	string	ja
thread_id	ID der Konversation	string	ja
subject	Betreff der Nachricht	string	nein
body	Text der Nachricht	string	nein

Tabelle 5.57 Felder der Unified-Message-Tabelle

Feldname	Beschreibung	Typ	Indiziert
unread	Zeigt an, ob die Nachricht noch nicht gelesen wurde.	boolean	ja
action_id	Numerische Versions-ID der Konversation – diese ID spiegelt den Status einer Konversation zu einem bestimmten Zeitpunkt wider und wird laufend erhöht.	string	nein
timestamp	UNIX-Zeitstempel der Veröffentlichung der Nachricht	UNIX-Zeitstempel	ja
tags	Tags der Nachricht	Array	nein
sender	Absender der Nachricht (Felder name, id, email)	Array	nein
recipients	Liste der Empfänger der Nachricht (Felder name, id, email)	Array	nein
object_sender	Absender der Nachricht, wenn dieser eine Seite, Veranstaltung oder Gruppe ist (Felder object_address_type und id)	Array	nein
html_body	Text der Nachricht als HTML-String	string	nein
attachments	Liste der Anhänge der Nachricht	Array	nein
attachments_map	Verknüpfung der Anhänge zu Objekten mit weiterführenden Informationen (Felder id, type, mime_type, filename, file_size)	Array	nein
shares	Liste der geteilten Objekte (Links, Videos, Fotos ...)	Array	nein
shares_map	Verknüpfung der gesharten Objekte mit weiterführenden Informationen (share_id, type, href, title, summary, image)	Array	nein

Tabelle 5.57 Felder der Unified-Message-Tabelle (Forts.)

5.2.40 URL-Like-Tabelle

Die FQL-Tabelle url_like enthält alle externen, per Open Graph integrierten Seiten, auf denen ein Benutzer den GEFÄLLT-MIR-Button angeklickt hat. Zum Lesen der Tabelle ist die erweiterte Berechtigung user_likes bzw. friends_likes notwendig.

Beispiel – Lesen aller Gefällt mir eines bestimmten Benutzers:

```
SELECT url FROM url_like WHERE user_id = 532617296
```

Feldname	Beschreibung	Typ	Indiziert
user_id	ID des Benutzers, der auf Gefällt mir geklickt hat	int	ja
url	URL der externen Seite, auf der der Gefällt-mir-Button angeklickt wurde	string	nein

Tabelle 5.58 Felder der URL-Like-Tabelle

5.2.41 User-Tabelle

Die FQL-Tabelle user enthält Benutzerobjekte, wie in Abschnitt 3.2.1, »Das User-Objekt«, beschrieben. Zur Abfrage der einzelnen Felder sind jeweils jene Access Tokens und erweiterten Berechtigungen notwendig, die auch beim Zugriff über die Graph API notwendig sind.

Beispiel – Lesen der Benutzerinformationen des aktuellen Benutzers:

```
SELECT uid, name, username, pic FROM user WHERE uid=me()
```

Feldname	Beschreibung	Typ	Indiziert
uid	ID des Benutzers	int	ja
username	eindeutiger Benutzername	string	ja
first_name	Vorname	string	nein
middle_name	zweiter Vorname	string	nein
last_name	Nachname	string	nein
name	vollständiger Name	string	ja
pic_small	URL zum kleinen Profilbild des Benutzers (maximal 50 x 150 Pixel)	string	nein
pic_big	URL zum großen Profilbild des Benutzers (maximal 200 x 600 Pixel)	string	nein

Tabelle 5.59 Felder der User-Tabelle

Feldname	Beschreibung	Typ	Indiziert
pic_square	URL zum quadratischen Profilbild des Benutzers (50 x 50 Pixel)	string	nein
pic	URL zum mittelgroßen Profilbild des Benutzers (maximal 100 x 300 Pixel)	string	nein
affiliations	Netzwerke, denen der Benutzer angehört	Array	nein
profile_update_time	UNIX-Zeitstempel der letzten Aktualisierung des Profils. 0, wenn diese länger als drei Tage zurückliegt.	UNIX-Zeitstempel	nein
timezone	Zeitzone des Benutzers als Differenz zu UTC	int	nein
religion	Religionszugehörigkeit	string	nein
birthday	Geburtstag in lokalisiertem Format	string	nein
birthday_date	Geburtstag im Format MM/DD/YYY	string	nein
sex	Geschlecht	string	nein
hometown_location	Heimatstadt	Array	nein
meeting_sex	Geschlechter, die der Benutzer kennenlernen möchte	Array	nein
meeting_for	Gründe, aus denen der Benutzer andere Benutzer kennenlernen möchte	Array	nein
relationship_status	Beziehungsstatus	string	nein
significant_other_id	Benutzer-ID des Partners	int	nein
political	politische Einstellung	string	nein
current_location	aktueller Aufenthaltsort	Array	nein
activities	Aktivitäten	string	nein
interests	Interessen	string	nein

Tabelle 5.59 Felder der User-Tabelle (Forts.)

Feldname	Beschreibung	Typ	Indiziert
is_app_user	Zeigt an, ob der Benutzer die aktuelle Anwendung installiert hat.	boolean	nein
music	Lieblingsmusik	string	nein
tv	Lieblings-TV-Show	string	nein
movies	Lieblingsfilme	string	nein
books	Lieblingsbücher	string	nein
quotes	Lieblingszitate	string	nein
about_me	Kurz-Bio des Benutzers	string	nein
hs_info	Highschool (eingestellt)	Array	nein
education_history	Ausbildung (eingestellt)	Array	nein
work_history	Arbeitserfahrung (eingestellt)	Array	nein
notes_count	Anzahl der veröffentlichten Notizen	int	nein
wall_count	Anzahl der veröffentlichten Pinnwand-Einträge	int	nein
status	aktueller Status	string	nein
has_added_app	eingestellt, entspricht dem Feld is_app_user	boolean	nein
online_presence	Status im Facebook-Chat (active, idle, offline, error)	string	nein
locale	Lokalisierungseinstellung des Benutzers	string	nein
proxied_email	Enthält die Proxy-E-Mail-Adresse des Benutzers, wenn diese bei der Erteilung der Berechtigung email bereitgestellt wurde.	string	nein
profile_url	URL zum Profil des Benutzers	string	nein
email_hashes	Liste der bestätigten E-Mail-Adressen als Hashes (eingestellt) mit Facebook-Icon als Overlay	Array	nein

Tabelle 5.59 Felder der User-Tabelle (Forts.)

Feldname	Beschreibung	Typ	Indiziert
pic_small_with_logo	URL zum kleinen Profilbild des Benutzers (maximal 50 x 150 Pixel) mit Facebook-Icon als Overlay	string	nein
pic_big_with_logo	URL zum großen Profilbild des Benutzers (maximal 200 x 600 Pixel) mit Facebook-Icon als Overlay	string	nein
pic_square_with_logo	URL zum quadratischen Profilbild des Benutzers (50 x 50 Pixel) mit Facebook-Icon als Overlay	string	nein
pic_with_logo	URL zum mittelgroßen Profilbild des Benutzers (maximal 100 x 300 Pixel) mit Facebook-Icon als Overlay	string	nein
allowed_restrictions	kommaseparierte Liste der demographischen Zugriffseinschränkungen, die der Benutzer erfüllt – derzeit nur alcohol	string	nein
verified	Zeigt an, ob das Konto des Benutzers per Handy oder Kreditkarte verifiziert wurde.	boolean	nein
profile_blurb	Text unter dem Profilbild des Benutzers	string	nein
family	Liste der unmittelbaren Verwandten des Benutzers (Felder relationship, uid, name, birthday)	Array	nein
website	Website des Benutzers	string	nein
is_blocked	Zeigt an, ob der abgefragte Benutzer gegenüber dem aktuellen Benutzer geblockt ist.	boolean	nein
contact_email	primäre E-Mail-Adresse des Benutzers	string	nein
email	primäre E-Mail-Adresse oder Proxy-E-Mail-Adresse des Benutzers	string	nein
third_party_id	eine anonymisierte, eindeutige Benutzerkennung	string	ja
name_format	Angabe zur Formatierung des vollen Namens, z. B. »{first} {last}«	string	nein

Tabelle 5.59 Felder der User-Tabelle (Forts.)

Feldname	Beschreibung	Typ	Indiziert
video_upload_limits	maximale Länge und Dateigröße von Video-Uploads (Felder length und size)	Array	nein
hames	Lieblingsspiel	string	nein
is_minor	Zeigt an, ob der Benutzer minderjährig ist.	boolean	nein
work	Liste der Arbeitserfahrung (Felder employer, location, position, start_date, end_date)	Array	nein
education	Liste der Ausbildungen (Felder name, id, type, year, degree, concentration, classes)	Array	nein
sports	Liste der ausgeübten Sportaren (Felder id, name)	Array	nein
favorite_athletes	Liste der Lieblingsathleten (Felder id, name) – eingestellt	Array	nein
favorite_teams	Liste der Lieblingsteams (Felder id, name) – eingestellt	Array	nein
inspirational_people	Liste der inspirierenden Personen (Felder id, name)	Array	nein
languages	Liste der gesprochenen Sprachen (Felder id, name)	Array	nein
likes_count	Anzahl der Seiten, auf denen der Benutzer auf Gefällt mir geklickt hat	int	nein
mutual_friend_count	Anzahl der Freunde, die der abgefragte Benutzer mit dem aktuellen Benutzer gemeinsam hat	int	nein
can_post	Zeigt an, ob der aktuelle Benutzer auf der Pinnwand des abgefragten Benutzers posten kann.	boolean	nein

Tabelle 5.59 Felder der User-Tabelle (Forts.)

5.2.42 Videotabellen

Die FQL-Tabelle video bildet Videoobjekte, wie in Abschnitt 3.2.8, »Das Videoobjekt«, beschrieben, ab. Während öffentliche Videos mit einem beliebigen Access Token

abgefragt werden können, ist für private Videos ein Access Token mit der erweiterten Berechtigung `user_videos` bzw. `friends_videos` notwendig.

Beispiel – Lesen aller Videos des aktuellen Benutzers:

```
SELECT vid, owner, title, description, thumbnail_link, embed_html,
updated_time, created_time
FROM video WHERE owner=me()
```

Feldname	Beschreibung	Typ	Indiziert
`vid`	ID des Videos	`int`	ja
`owner`	Benutzer-ID des Besitzers des Videos	`int`	ja
`album_id`	ID des zugehörigen Albums	`int`	ja
`title`	Titel des Videos	`string`	nein
`description`	Beschreibung des Videos	`string`	nein
`link`	URL zum Video	`string`	nein
`thumbnail_link`	URL zum Thumbnail-Bild des Videos	`string`	nein
`embed_html`	HTML-Code zum Einbinden des Videos in externe Seiten	`string`	nein
`created_time`	UNIX-Zeitstempel der Erstellung des Videos	UNIX-Zeitstempel	nein
`updated_time`	UNIX-Zeitstempel des letzten Updates des Videos	UNIX-Zeitstempel	nein
`length`	Dauer des Videos in Sekunden	`int`	nein
`src`	URL zur Quelldatei des Videos in normaler Auflösung	`string`	nein
`src_hq`	URL zur Quelldatei des Videos in hoher Auflösung	`string`	nein

Tabelle 5.60 Felder der Videotabelle

Video-Tag-Tabelle

Die FQL-Tabelle `video_tag` enthält Videomarkierungen (*Tags*) von Benutzern und entspricht damit dem `tags`-Feld des `video`-Objekts, wie in Abschnitt 3.2.8, »Das Videoobjekt«, beschrieben. Abgesehen von den öffentlichen Videos einer Seite, ist zum Lesen von Videomarkierungen die Berechtigung `user_videos` bzw. `friends_videos` erforderlich.

Beispiel – Lesen aller Videomarkierungen des aktuellen Benutzers:

```
SELECT vid FROM video_tag WHERE subject = me()
```

Beispiel – Lesen aller auf einem bestimmten Video markierten Benutzer:

```
SELECT subject FROM video_tag WHERE vid=10150309485965864
```

Neben Videomarkierungen wird `video_tag` auch genutzt, um die Videos, die mit einer Gruppe oder einem Event verknüpft sind, auszulesen. Dazu muss als `subject` lediglich die ID der Gruppe oder des Events abgefragt werden.

Beispiel – Lesen aller Videos, die mit einer bestimmten Gruppe verknüpft sind:

```
SELECT vid FROM video_tag
WHERE subject = 269627173053176
```

Feldname	Beschreibung	Typ	Indiziert
`vid`	ID des Videos	`int`	ja
`subject`	ID des markierten Benutzers, der Veranstaltung oder Gruppe	`int`	ja
`created_time`	UNIX-Zeitstempel der Erstellung der Videomarkierung	UNIX-Zeitstempel	nein
`updated_time`	UNIX-Zeitstempel des letzten Updates der Videomarkierung	UNIX-Zeitstempel	nein

Tabelle 5.61 Felder der Video-Tag-Tabelle