Using SQL: The Basics

Getty Vocabulary Program

Patricia Harpring
8 April 2009

- SQL uses Boolean logic that you are already familiar with
- Structured Query Language, called "sequel"

- Boolean operators **and**, **or**, **not**
- Plus the use of parentheses and other punctuation to place phrases in logical order

```
baumann and (lydia or anna or daniel)
```

```
(baumann and lydia) or (baumann and anna) or (baumann and daniel)
```
SQL uses Boolean logic that you are already familiar with:

- baumann and (lydia or ann% or daniel)
- bauman% and (lydia or ann% or daniel)
- (bauman% and (lydia or ann% or daniel)) or (bauer and (carl or karl or johan%))

To create SQL queries, use Relational Tables Diagram:
- Each field is a column in a table (here horizontal);
- (each instance is a row)
- The tables are linked (often using subject_id)
- Easiest queries are done within a single table.
Using SQL to Formulate Queries in VCS

Select xxxx From xxxx Where xxxx

In VCS, “select” must always be a count or subject_id

Select subject_id  from revision_history
where user_name like 'PHARPRING'
and date_time > '8 Apr 2009'

Select count subject_id  from revision_history
where user_name like 'LOADER'
and date_time > '1 Jul 2008'

Select count(distinct subject_id)  from revision_history
where user_name like 'LOADER'
and date_time > '1 Jul 2008'

In VCS, “select” must always be a count or subject_id
Select distinct subject_ID from language_rels
where language_code like '70431'
or language_code like '70432'

Select distinct subject_ID from biography
where preferred like 'P'
and biography like '%&%'

What does this retrieve?

What does this retrieve?

If you need to search on two or more columns in different tables, you need a “join”

```sql
select count(distinct term.term_id)
from term,subject,language_rels
where language_rels.language_code not like '70051'
and subject.candidate_stat like 'N'
and term.subject_id=subject.subject_id
and term.subject_id=language_rels.subject_id
```

Join the three tables using subject_id