

SECTION 2

CSE444

January 13, 2011

Today

- Questions on Project 1???
- Create tables
- Insert/Update/Delete
- Constraints
- Basic SQL review
- Practice with grouping and aggregation

Document index database

Author (aid, name)

Auth_Doc (aid, did)

Document (did, title, year)

Doc_Word (did, word)

Word (word)

Underlined = key (unique identifier for a tuple)



Create tables from schema

Author (aid, name)

Auth_Doc (aid, did)

Document (did, title, year)

Doc_Word (did, word)

Word (word)

Underlined = key (unique identifier for a tuple)

- What types of constraints are there?

Insert, Update, and Delete

INSERT INTO AUTHOR VALUES (312,'Michael Chabon',45);

UPDATE AUTHOR SET AGE=46 WHERE aid=312;

DELETE FROM AUTHOR WHERE aid=312;

Note: for **DELETE** [be careful! don't forget the WHERE condition!]

Constraints

- What are examples of ICs (Incentive Compatibility) constraints that we might want?
 - Keys
 - Foreign keys
 - Attribute level
 - Tuple level
 - Global constraints
- Policies?
 - Reject
 - Cascade
 - Set NULL

SQL warmup

In groups we'll do two exercises practicing these techniques

1. Join

- Who wrote this paper?
- “Molecular structure of nucleic acids: a structure for deoxyribose nucleic acid” (1953)

2. Aggregate (with and without group by)

- Find authors who wrote ≥ 20 docs

Exercise 1

Who wrote this paper?

“Molecular structure of nucleic acids: a structure for deoxyribose nucleic acid” (1953)

Authors of double-helix paper

-- Authors of the double-helix paper

```
SELECT a.name
```

```
FROM Author a, Auth_Doc ad, Document d
```

```
WHERE a.aid = ad.aid AND
```

```
      ad.did = d.did AND
```

```
      d.year = 1953 AND
```

```
      d.title = 'Molecular structure of nucleic acids: a  
structure for deoxyribose nucleic acid';
```

Exercise 2

Find authors who wrote ≥ 20 docs

1. With GROUP BY
2. Without GROUP BY

Find authors who wrote ≥ 20 docs

-- Authors who wrote more than 20 papers
(without group by)

```
SELECT name
FROM Author a
WHERE 20 <= (SELECT COUNT(*)
              FROM Auth_Doc ad
              WHERE ad.aid = a.aid)
```

Find authors who wrote ≥ 20 docs

-- Use grouping to eliminate the subquery:

```
SELECT name  
FROM Author a, Auth_Doc ad  
WHERE a.aid = ad.aid  
GROUP BY a.aid, a.name  
HAVING COUNT(*)  $\geq$  20
```

Find authors who wrote ≥ 20 docs

Use grouping to eliminate the subquery:

```
SELECT name
```

```
FROM Author a, Auth_Doc ad
```

```
WHERE a.aid = ad.aid
```

```
GROUP BY a.aid, a.name
```

```
HAVING COUNT(*)  $\geq$  20
```

← One row per
(a.aid, a.name)
pair

Find authors who wrote ≥ 20 docs

Use grouping to eliminate the subquery:


```
SELECT name  
FROM Author a, Auth_Doc ad  
WHERE a.aid = ad.aid  
GROUP BY a.aid, a.name  
HAVING COUNT(*)  $\geq$  20
```

← Only groups that
combine ≥ 20
tuples will match

Find authors who wrote ≥ 20 docs

Use grouping to eliminate the subquery:

```
SELECT name  
FROM Author a, Auth_Doc ad  
WHERE a.aid = ad.aid  
GROUP BY a.aid, a.name  
HAVING COUNT(*)  $\geq$  20
```



If aid is the key,
why group by
name?

If we deleted a.name...

ERROR: Column 'name' is invalid in the select list because it is not contained in either an aggregate function or the GROUP BY clause.

Find Authors who wrote the most docs

```
SELECT name
FROM Author a, Auth_Doc ad
WHERE a.aid = ad.aid
GROUP BY a.aid, a.name
HAVING COUNT(*) >= 20
```

- What do we need to change from this query that returns authors who have written more than 20 documents?
- Hint: Can you think of a way using TOP 1?
- Hint: Can you think of a way using using “HAVING not exists ...”?

Find authors who wrote the most docs

One solution:

```
SELECT TOP 1 name, COUNT(ad.aid)
FROM Author a, Auth_Doc ad
WHERE a.aid = ad.aid
GROUP BY a.aid, a.name
ORDER BY COUNT(ad.aid) DESC;
```

Find authors who wrote the most docs

Will return multiple rows in case of ties:

```
SELECT name, COUNT(ad.aid)
FROM Author a, Auth_Doc ad
WHERE a.aid = ad.aid
GROUP BY a.aid, a.name
HAVING not exists (SELECT a2.aid
                   FROM Author a2, Auth_Doc ad2
                   WHERE a2.aid = ad2.aid
                   GROUP BY a2.aid
                   HAVING COUNT(ad2.aid) >
                          COUNT(ad.aid))
```

Find authors who wrote the most docs

Will return multiple rows in case of ties (alternative):

```
SELECT name, COUNT(ad.did)
FROM Author a, Auth_Doc ad
WHERE a.aid = ad.aid
GROUP BY a.aid, a.name
HAVING COUNT(ad.did) >= ALL (SELECT COUNT(ad.did)
                             FROM Author a2, Auth_Doc ad2
                             WHERE a2.aid = ad2.aid
                             GROUP BY a2.aid)
```

Find the average word count by author

- For each author, return the average number of words that author used in their documents

Average word count by author

- -- Average word count by author

```
SELECT name, COUNT(*) / COUNT(DISTINCT ad.did) AS  
avg_word_count  
FROM Author a, Auth_Doc ad, Doc_Word dw  
WHERE a.aid = ad.aid AND ad.did = dw.did  
GROUP BY a.aid, a.name;
```

More examples (try at home)

- For each author, give the total number of words in all documents he has (co-)written.
- For each author, give the average length in words of his documents.
- Give the author with the longest average documents.
- All words used by at least 10 authors
- The most frequently used word
- The longest document
- Authors of the longest document

Questions

Have a good long weekend!