Today

- Creating tables in SQL (cont)
- Basic SQL Queries

Assignments

- HW2 due
- HW3 assigned Tues
- Quiz 2 on Tues
Declaring Tables in SQL (redux)

Defining the Account table:

- better names, more specific types

```sql
CREATE TABLE account (
    account_id INT UNSIGNED,
    owner TINYTEXT NOT NULL,
    balance DECIMAL(8,2) UNSIGNED NOT NULL,
    account_type ENUM('checking', 'savings'),
    PRIMARY KEY (account_id)
) ENGINE = InnoDB;
```

Defining the Check table:

```sql
CREATE TABLE check_transaction (  
    account_id INT UNSIGNED,
    check_number INT UNSIGNED NOT NULL,
    check_date DATE,
    amount DECIMAL(8,2) NOT NULL,
    PRIMARY KEY (account_id, check_number),
    FOREIGN KEY (account_id) REFERENCES account (account_id)
) ENGINE = InnoDB;
```
Naming your constraints

CREATE TABLE check_transaction (  
    account_id  INT UNSIGNED,  
    check_number INT UNSIGNED NOT NULL,  
    check_date DATE,  
    amount DECIMAL(8,2) UNSIGNED NOT NULL,  
    PRIMARY KEY (account_id, check_number),  

    CONSTRAINT account_fk FOREIGN KEY (account)  
        REFERENCES account (number)  
) engine = InnoDB;

Remove constraints “after the fact” (by name)

ALTER TABLE check DROP FOREIGN KEY account_fk;

NOTE 1: Don’t usually do this for foreign keys ...

NOTE 2: MySQL and FKs

- the default database engine in MySQL is MyISAM (more later)
- which doesn’t support FKs
- instead use engine=InnoDB:

    CREATE TABLE mytable ( ... ) engine = InnoDB;
Adding constraints “after the fact”

```
ALTER TABLE check_transaction ADD CONSTRAINT account_fk
    FOREIGN KEY (account_id) REFERENCES account (account_id);
```

Dropping a foreign-key constraint

```
ALTER TABLE check_transaction DROP FOREIGN KEY account_fk;
```

CHECK constraints ...

```
CREATE TABLE check_transaction (  
    account_id    INT UNSIGNED,  
    check_number  INT UNSIGNED,  
    check_date    DATE,  
    amount        DECIMAL(8,2) UNSIGNED,  
    PRIMARY KEY (account_id, check_number),

    CONSTRAINT account_fk FOREIGN KEY (account)
        REFERENCES account (number),

    CONSTRAINT check_amt_range
        CHECK (amount > 0 and amount <= 100000)

) engine = InnoDB;
```

Or can add it “after the fact”

Can also drop it “after the fact”

```
ALTER TABLE check_transaction DROP CONSTRAINT check_amt_range;
```
More on Modifying Tables using SQL

Drop a column from an existing table ... 

    ALTER TABLE t DROP COLUMN c;     -- remove column c

Change name and type of an attribute ... 

    ALTER TABLE t CHANGE c d type;     -- change c to "d type"

Change type of an attribute (leave name unchanged) ... 

    ALTER TABLE t MODIFY c new_type;    -- change c’s type to new_type

Add a new attribute ... 

    ALTER TABLE t ADD d type;         -- add "d type" attribute
More on Populating Tables using SQL

Removing rows from tables

    DELETE FROM checks;

- Can also remove specific rows using an SQL query (more later)
- For example:

    DELETE FROM checks
    WHERE account = 101;

Updating rows

    UPDATE account SET balance = 5.00;

- sets all account balances to $5.00!

    UPDATE account SET balance = 5.00
    WHERE number = 101;

- only updates 101’s account balance

Use AUTO_INCREMENT for surrogate keys (start at 1, ...)

    CREATE TABLE pet (        
        id INT UNSIGNED NOT NULL AUTO_INCREMENT,        
        name CHAR(30) NOT NULL,        
        PRIMARY KEY (id)        
    );        
    INSERT INTO pet (name) VALUES (‘dog’), (‘cat’), (‘bird’);
Basic SQL Queries: Example Tables

account

<table>
<thead>
<tr>
<th>number</th>
<th>owner</th>
<th>balance</th>
<th>type</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td>J. Smith</td>
<td>1000.00</td>
<td>checking</td>
</tr>
<tr>
<td>102</td>
<td>W. Wei</td>
<td>2000.00</td>
<td>checking</td>
</tr>
<tr>
<td>103</td>
<td>J. Smith</td>
<td>5000.00</td>
<td>savings</td>
</tr>
<tr>
<td>104</td>
<td>M. Jones</td>
<td>1000.00</td>
<td>checking</td>
</tr>
<tr>
<td>105</td>
<td>H. Martin</td>
<td>10000.00</td>
<td>checking</td>
</tr>
</tbody>
</table>

deposit

<table>
<thead>
<tr>
<th>account</th>
<th>transaction_id</th>
<th>date</th>
<th>amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>102</td>
<td>1</td>
<td>10/22/11</td>
<td>500.00</td>
</tr>
<tr>
<td>102</td>
<td>2</td>
<td>10/29/11</td>
<td>200.00</td>
</tr>
<tr>
<td>104</td>
<td>3</td>
<td>10/29/11</td>
<td>1000.00</td>
</tr>
<tr>
<td>105</td>
<td>4</td>
<td>11/2/11</td>
<td>10000.00</td>
</tr>
</tbody>
</table>

check

<table>
<thead>
<tr>
<th>account</th>
<th>check_number</th>
<th>date</th>
<th>amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td>924</td>
<td>10/23/11</td>
<td>125.00</td>
</tr>
<tr>
<td>101</td>
<td>925</td>
<td>10/24/11</td>
<td>23.98</td>
</tr>
</tbody>
</table>

FK's:
- deposit.account → account.number
- check.account → account.number
SQL Queries: Query Example 1

<table>
<thead>
<tr>
<th>account</th>
<th>transaction_id</th>
<th>date</th>
<th>amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>102</td>
<td>1</td>
<td>10/22/11</td>
<td>500.00</td>
</tr>
<tr>
<td>102</td>
<td>2</td>
<td>10/29/11</td>
<td>200.00</td>
</tr>
<tr>
<td>104</td>
<td>3</td>
<td>10/29/11</td>
<td>1000.00</td>
</tr>
<tr>
<td>105</td>
<td>4</td>
<td>11/2/11</td>
<td>10000.00</td>
</tr>
</tbody>
</table>

```
SELECT account, amount
FROM deposit
WHERE amount < 1000
```

The result of an SQL query is always a new table

We often read queries “inside out”

- The FROM clause specifies what tables are being queried
- The WHERE clause is evaluated for each row in the table
  - Which rows “match” the WHERE clause amount < 1000?
  - Can think of this as creating an “intermediate” table
- The SELECT clause lists attributes to keep in the answer
  - Here we only keep account and amount
  - Drop these from the intermediate table to get query answer

Query answer:

<table>
<thead>
<tr>
<th>account</th>
<th>amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>102</td>
<td>500.00</td>
</tr>
<tr>
<td>102</td>
<td>200.00</td>
</tr>
</tbody>
</table>
Query Example 2

<table>
<thead>
<tr>
<th>account</th>
<th>transaction_id</th>
<th>date</th>
<th>amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>102</td>
<td>1</td>
<td>10/22/11</td>
<td>500.00</td>
</tr>
<tr>
<td>102</td>
<td>2</td>
<td>10/29/11</td>
<td>200.00</td>
</tr>
<tr>
<td>104</td>
<td>3</td>
<td>10/29/11</td>
<td>1000.00</td>
</tr>
<tr>
<td>105</td>
<td>4</td>
<td>11/2/11</td>
<td>10000.00</td>
</tr>
</tbody>
</table>

SELECT *
FROM deposit
WHERE transaction_id = 3;

- SELECT * is shorthand for returning every column
- What does this query return?

What this might look like as Python code ...

```python
new_table = []  # new result table
for row in deposit:  # go row by row
    if row['transaction_id'] == 3:  # check condition
        new_table.append(row[:])  # add a copy of the row

return new_table  # return the result
```
Query Example 3 & 4

<table>
<thead>
<tr>
<th>number</th>
<th>owner</th>
<th>balance</th>
<th>type</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td>J. Smith</td>
<td>1000.00</td>
<td>checking</td>
</tr>
<tr>
<td>102</td>
<td>W. Wei</td>
<td>2000.00</td>
<td>checking</td>
</tr>
<tr>
<td>103</td>
<td>J. Smith</td>
<td>5000.00</td>
<td>savings</td>
</tr>
<tr>
<td>104</td>
<td>M. Jones</td>
<td>1000.00</td>
<td>checking</td>
</tr>
<tr>
<td>105</td>
<td>H. Martin</td>
<td>10000.00</td>
<td>checking</td>
</tr>
</tbody>
</table>

SELECT *
FROM account
WHERE type = 'checking';

• What does this query return?

SELECT *
FROM account
WHERE type = 'savings';

• What about this query?
Query Example 5

**account**

<table>
<thead>
<tr>
<th>number</th>
<th>owner</th>
<th>balance</th>
<th>type</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td>J. Smith</td>
<td>1000.00</td>
<td>checking</td>
</tr>
<tr>
<td>102</td>
<td>W. Wei</td>
<td>2000.00</td>
<td>checking</td>
</tr>
<tr>
<td>103</td>
<td>J. Smith</td>
<td>5000.00</td>
<td>savings</td>
</tr>
<tr>
<td>104</td>
<td>M. Jones</td>
<td>1000.00</td>
<td>checking</td>
</tr>
<tr>
<td>105</td>
<td>H. Martin</td>
<td>10000.00</td>
<td>checking</td>
</tr>
</tbody>
</table>

```sql
SELECT owner 
FROM account;
```

- What does this query return?

Query answer:

```
<table>
<thead>
<tr>
<th>owner</th>
</tr>
</thead>
<tbody>
<tr>
<td>J. Smith</td>
</tr>
<tr>
<td>W. Wei</td>
</tr>
<tr>
<td>J. Smith</td>
</tr>
<tr>
<td>M. Jones</td>
</tr>
<tr>
<td>H. Martin</td>
</tr>
</tbody>
</table>
```