Lecture 5: Database Design

Wednesday, February 11, 2015
Review: What’s wrong with this SQL query?

```
SELECT v.vendor_id, v.vendor_name, COUNT(*) AS number_invoices
FROM vendors v, invoices i
WHERE v.vendor_id = i.vendor_id
GROUP BY v.vendor_name
ORDER BY number_invoices
```
Corrected SQL query (which also includes empty groups)

```sql
SELECT v.vendor_id, v.vendor_name, COUNT(invoice_number) AS number_invoices
FROM vendors v LEFT OUTER JOIN invoices i
ON v.vendor_id = i.vendor_id
GROUP BY v.vendor_id, v.vendor_name
ORDER BY number_invoices
```

Notice that we also replace COUNT(*) with COUNT(invoice_number) when using an outer join to avoid adding up the NULL records.
Database Design Process

Phase 1: Conceptual Model
Phase 2: Relational Model
Phase 3: Normalization
Phase 4: Physical Schema
ERD Notation

Entities
- Customer

Attributes
- name

Relationships between entities
- places
Note: Every entity must have a primary key
id

Customer

name
address

Item

id
name
price

places
contains

Order

id
order_date
shipped_date
Types of relationships:

one-to-one:

one-to-many:

many-to-one:

many-to-many:
What do these relationships mean?
Multi-way Relationships

- Order
- Item
- date

Contains relationship

Order contains date

Item contains date
What’s wrong with this design?
What’s wrong with this design?
From E/R Diagrams to Relational Schema

• Entity $\rightarrow$ Relation
• Relationship $\rightarrow$ Relation
From Entity to Relation

Item(id, name, price)

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>price</th>
</tr>
</thead>
<tbody>
<tr>
<td>90</td>
<td>iPad mini</td>
<td>399.99</td>
</tr>
<tr>
<td>20</td>
<td>iPhone5</td>
<td>299.99</td>
</tr>
</tbody>
</table>
CREATE TABLE Item (  
id NUMBER(8) PRIMARY KEY,  
name VARCHAR(20),  
price NUMBER(6,2)  
)
How do we convert these relationships from ER diagram into SQL?
Create Table Statement

```
CREATE TABLE Customer (  
id NUMBER(8) PRIMARY KEY,  
first_name VARCHAR(50) NOT NULL,  
last_name VARCHAR(50) NOT NULL,  
address VARCHAR(50) NOT NULL,  
city VARCHAR(30) NOT NULL,  
state CHAR(2) NOT NULL,  
zip CHAR(5) NOT NULL,  
phone CHAR(10) NOT NULL,  
)
```
CREATE TABLE Order ( 
    id NUMBER(8) PRIMARY KEY, 
    order_date DATE NOT NULL, 
    shipped_date DATE, 
    customer_id NUMBER(8) NOT NULL, 
    FOREIGN KEY REFERENCES Customer(id) 
)
OrderDetails(order-id, item-id, date)

<table>
<thead>
<tr>
<th>order-id</th>
<th>item-id</th>
<th>date</th>
</tr>
</thead>
<tbody>
<tr>
<td>55</td>
<td>90</td>
<td>02/11/2015</td>
</tr>
<tr>
<td>55</td>
<td>20</td>
<td>02/09/2015</td>
</tr>
</tbody>
</table>
Create Table Statement

```sql
CREATE TABLE OrderDetail(
    order-id NUMBER(8),
    item-id NUMBER(8),
    date_added DATE,
    PRIMARY KEY (order-id, item-id),
    FOREIGN KEY (order-id) REFERENCES Order(id),
    FOREIGN KEY (item-id) REFERENCES Item(id)
)
```
## A one-to-many example

### vendors
- `vendor_id`
- `vendor_name`
- `vendor_address`
- `vendor_city`
- `vendor_state`
- `vendor_zip_code`
- `vendor_phone`
- `vendor_contact_first_name`
- `vendor_contact_last_name`
- `terms`
- `account_no`

### invoices
- `invoice_id`
- `vendor_id`
- `invoice_number`
- `invoice_date`
- `invoice_total`
- `payment_total`
- `credit_total`
- `terms`
- `invoice_due_date`
- `payment_date`
- `account_no`

### invoice_line_items
- `invoice_id`
- `invoice_sequence`
- `account_no`
- `line_item_description`
- `item_quantity`
- `item_unit_price`
- `line_item_amount`
A many-to-many example

Linking table
A one-to-one example

employees

- employee_id
- first_name
- last_name

employee_photos

- employee_id
- employee_photo

employees employee_photos

employee_id

employee_id