Relational Database Design

CS 327E
September 27, 2017
Announcements:

• Athena credits
• Reminder: Lab 2 next week
1) Given the create table statement, which values in the sample data do not satisfy the table definition?

```sql
create table Pokemon_Characters(
    id int primary key,
    name varchar(50) not null,
    type int,
    height_ft double,
    weight_lbs double,
    health_pts int,
    foreign key (type) references Pokemon_Types(id) on delete cascade
);
```

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>type</th>
<th>height_ft</th>
<th>weight_lbs</th>
<th>health_pts</th>
</tr>
</thead>
<tbody>
<tr>
<td>16114</td>
<td>Ponyta</td>
<td>Fire</td>
<td>3.03</td>
<td>66.1</td>
<td>60</td>
</tr>
<tr>
<td>5620</td>
<td>Tyranitar</td>
<td>802</td>
<td>6.07</td>
<td>445.3</td>
<td>120</td>
</tr>
<tr>
<td>2298</td>
<td>Vaporeon</td>
<td>Water</td>
<td>3.03</td>
<td>63.9</td>
<td>90</td>
</tr>
</tbody>
</table>

A) 'Fire'
B) 802
C) 'Tyranitar'
D) 'Fire' and 'Water'
2) In order to create the table Pokemon_Characters, the referenced table Pokemon_Types must already exist in the database.

```sql
CREATE TABLE Pokemon_Characters(
    id INT PRIMARY KEY,
    name VARCHAR(50) NOT NULL,
    type INT,
    height_ft DOUBLE,
    weight_lbs DOUBLE,
    health_pts INT,
    FOREIGN KEY (type) REFERENCES Pokemon_Types(id) ON DELETE CASCADE)
```

A) True  
B) False
3) After creating the table Pokemon_Characters, we decide to add another column to store a pokemon’s region. What value will we get if we retrieve the region immediately after modifying the table?

create table Pokemon_Characters(
    id int primary key,
    name varchar(50) not null,
    type int,
    height_ft double,
    weight_lbs double,
    health_pts int,
    foreign key (type) references PokemonTYPES(id)
);

alter table Pokemon_Characters add region varchar(30);

A) NULL  
B) 0  
C) "  
D) None
4) What would happen to the data in Pokemon_Characters if we delete the record Pokemon_Types.id = 802?

```
create table Pokemon_Characters(
    id int primary key,
    name varchar(50) not null,
    type int,
    height_ft double,
    weight_lbs double,
    health_pts int,
    foreign key (type) references Pokemon_Types(id) on delete cascade
);
```

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>type</th>
<th>height_ft</th>
<th>weight_lbs</th>
<th>health_pts</th>
</tr>
</thead>
<tbody>
<tr>
<td>16114</td>
<td>Ponyta</td>
<td>Fire</td>
<td>3.03</td>
<td>66.1</td>
<td>60</td>
</tr>
<tr>
<td>5620</td>
<td>Tyranitar</td>
<td>802</td>
<td>6.07</td>
<td>445.3</td>
<td>120</td>
</tr>
<tr>
<td>2298</td>
<td>Vaporeon</td>
<td>Water</td>
<td>3.03</td>
<td>63.9</td>
<td>90</td>
</tr>
</tbody>
</table>

A) No changes to the data in Pokemon_Characters  
B) All records with type = 802 would get deleted from Pokemon_Characters  
C) The value 802 in Pokemon_Characters.type would get set to NULL  
D) The table Pokemon_Characters would get dropped
5) What type of constraint would we use to require unique values in Pokemon_Characters.name?

create table Pokemon_Characters(
    id int primary key,
    name varchar(50) not null,
    type int,
    height_ft double,
    weight_lbs double,
    health_pts int,
    foreign key (type) references Pokemon_Types(id) on delete cascade
);

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>type</th>
<th>height_ft</th>
<th>weight_lbs</th>
<th>health_pts</th>
</tr>
</thead>
<tbody>
<tr>
<td>16114</td>
<td>Ponyta</td>
<td>Fire</td>
<td>3.03</td>
<td>66.1</td>
<td>60</td>
</tr>
<tr>
<td>5620</td>
<td>Tyranitar</td>
<td>802</td>
<td>6.07</td>
<td>445.3</td>
<td>120</td>
</tr>
<tr>
<td>2298</td>
<td>Vaporeon</td>
<td>Water</td>
<td>3.03</td>
<td>63.9</td>
<td>90</td>
</tr>
</tbody>
</table>

A) DISTINCT  
B) PRIMARY KEY  
C) UNIQUE  
D) CHECK
Demo: Postgres RDS
(see snippets repo for instructions)
Denormalization
Instacart ERD in 3NF
Instacart Dimensional Schema

**Users**
- **PK**: user_id
- first_name: varchar(50)
- last_name: varchar(50)
- address: varchar(100)
- city: varchar(100)
- state: char(2)
- zipcode: char(5)
- phone: char(10)

**Order_Facts**
- product_id: int
- user_id: int
- store_id: int
- quantity_ordered: int
- number_of_orders: int
- order_dollars: double

**Products**
- **PK**: product_id
- product_name: varchar(160)
- department_id: int
- department_name: varchar(20)
- aisle_id: int
- aisle_name: varchar(30)

**Stores**
- **PK**: store_id
- store_name: varchar(50)
Practice Problem 1: The Instacart business leaders would like to see how these order facts change over time. In particular, they want to analyze `quantity_ordered` by **Day of Week**, **User**, **Product**, and **Store**; they also want to analyze `quantity_ordered` by **Hour of Day**, **User**, **Product**, and **Store**.
Practice Problem 1: The Instacart business leaders would like to see how these order facts change over time. In particular, they want to analyze `quantity_ordered` by Day of Week, User, Product, and Store; they also want to analyze `quantity_ordered` by Hour of Day, User, Product, and Store.

How many new Order_Facts tables are needed to support the analyses?
A) 0  B) 1  C) 2  D) > 2
Demo: DDL and copy command
(see snippets repo for code)