Lecture 3:
- Quiz 1
- Relational Model (Part I: Representing data)

Homework:
- Homework 1 out
- Quiz 2 next Tues
More on Keys

Relations can have more than one set of key attributes:

Customer(c_id, email, ssn, fname, ...)

- for example, this relation has three possible keys!
- each is a separate candidate key (vs a composite key)

Designating attributes as keys is a type of (operational) constraint

- DBMS will not allow duplicate row values to be inserted!
- we say the DBMS enforces a (designated) key constraint

A primary key is a special designated key in a DBMS

- the primary key is chosen by the database designer (physical design)
- DBMS enforces the primary key and uses it to optimize storage (more later)
- if multiple candidate keys, only one selected as primary
- other keys can be enforced by marking them as “unique” (constraint)
We are usually interested in **minimal** candidate keys

- minimal implies can’t remove an attribute from a key (and still have a key)
- this means all simple keys are minimal
- note that we can always add attributes to a key and still have a key

**Examples:**

Vehicle(\texttt{v\_id}, \texttt{vt\_id}, \texttt{lat}, \texttt{lon})

- \texttt{v\_id} is the primary key
- thus, each row has a unique \texttt{v\_id} value (key constraint)
- \texttt{v\_id} is a minimal key (since it is simple)
- \((\texttt{v\_id}, \texttt{lat})\) is a non-minimal candidate key

Trip(\texttt{v\_id}, \texttt{start\_time}, \texttt{c\_id}, \texttt{charge})

- \((\texttt{v\_id}, \texttt{start\_time})\) is the primary key and is minimal
- \((\texttt{v\_id}, \texttt{start\_time}, \texttt{charge})\) is a non-minimal candidate key

**Check In:**

- What constraint is implied by a key?
- What is the difference between a simple and composite key?
- What is the difference between a composite key and a candidate key?
- What is a primary key?
- Given a table, identify the potential candidate keys
Foreign Key Constraints

VehicleType

<table>
<thead>
<tr>
<th>vt_id</th>
<th>make</th>
<th>model</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Segway</td>
<td>Ninebot</td>
</tr>
<tr>
<td>2</td>
<td>Lime-S</td>
<td>Gen3</td>
</tr>
<tr>
<td>3</td>
<td>Lime-S</td>
<td>Gen4</td>
</tr>
</tbody>
</table>

Vechicle

<table>
<thead>
<tr>
<th>v_id</th>
<th>vt_id</th>
<th>lat</th>
<th>lon</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>2</td>
<td>47.666</td>
<td>-117.403</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>47.668</td>
<td>-117.402</td>
</tr>
<tr>
<td>20</td>
<td>4</td>
<td>47.667</td>
<td>-117.401</td>
</tr>
</tbody>
</table>

Q: Should the last Vechicle row be legal? ... no vehicle type 4 in DB yet

- represents a “bad” reference to VehicleType

We can prevent these cases using **foreign keys** (FKs)

- FKS state that a column(s) values must come from a key in another table
- e.g., Vechicle.vt_id should be a foreign key to VehicleType.vt_id
- requiring Vehicle rows to refer to VechicleType rows
- note: foreign keys can be NULL, but primary keys cannot

Enforcing foreign key constraints helps maintain **referential integrity**

Check In:

- What is the difference between a primary key and a foreign key?
- What is meant by referential integrity and how do foreign keys help?
Check In: What are the foreign keys in the following tables?

VehicleType(vt_id, make, model, default_plan)
Vehicle(v_id, vt_id, lat, lon)
PricingPlan(p_id, price_per_min, unlock_price)
AllowedPlan(vt_id, p_id)

The foreign keys are:

- Vehicle.vt_id references VehicleType.vt_id
- AllowedPlan.vt_id references VehicleType.vt_id
- AllowedPlan.p_id references PricingPlan.p_id
- VehicleType.default_plan references PricingPlan.p_id

Note: AllowedPlan has a compound key
- a composite key where each attribute is a foreign key
Notes on Foreign Keys

1. A foreign key must reference the entire primary key
   • if \( P(x, y, z) \), \( Q(u, x, y) \), and \( Q \) has a FK to \( P \)
   • then \( Q.(x, y) \) references \( P.(x, y) \) could be a valid FK
   • but \( Q.x \) references \( P.x \) and \( Q.y \) references \( P.y \) are not valid FKS
   • note: the FK attributes must also have compatible data types (domains)

2. FK values must be either NULL or be values of the referenced PK
   • if a VehicleType does not have a default pricing plan
   • the row \( (3, \text{Lime-S}, \text{Gen4}, \text{NULL}) \) does not violate the FK constraint
   • however, if no PricingPlan row has \( p\_id = 10 \)
   • then \( (3, \text{Lime-S}, \text{Gen4}, 10) \) violates the FK

3. DBMSs enforce FK constraints
   • reject row insertions/updates that violate FKS
   • note: has implications on insertion order for tables ... why?

Check In:
• Given \( P(x, y, z) \) and \( Q(u, x, y) \), can \( Q.x \) be a FK to \( P.x \)?
• Can a foreign key reference a composite key? (Yes if the FK is composite)
• Can a primary key be NULL? Can a foreign key be NULL?
Primary and Foreign Keys and Schema Design

Keys add a number of design choices with subtle differences

Q: For example, what do the following key constraints imply?

1. `AllowedPlan(vt_id, p_id)
   • a vehicle type can have multiple allowed pricing plans
   • a pricing plan is allowed for different vehicle types

2. `AllowedPlan(vt_id, p_id)
   • a vehicle type has at most one allowed pricing plan
   • a pricing plan can be used for different vehicle types

3. `AllowedPlan(vt_id, p_id)
   • a vehicle type can have many different allowed pricing plans
   • a pricing plan is used for at most one vehicle (unique to each vehicle)

Note: Domains, PKs, and FKs are examples of integrity constraints
   • other types of integrity constraints can also be supported (more later)

Check in: shipping(courier, supplier, store)
   • Explain the meaning of different key constraints (e.g., PK courier).
   • Given a desired constraint, give the corresponding keys.
Surrogate Keys

Natural primary keys
- many tables have attributes that make for “natural” keys (e.g., student id)
- sometimes, the only candidate key is a composite key
- but composite keys aren’t always ideal as foreign keys (e.g., storage space)

Artificial keys are not “naturally occurring” keys
- added in place of composite keys or if no candidate keys exist

Surrogate keys are artificial keys with DBMS-generated values
- DBMS handles creation of surrogate key values
- specified in MySQL using `AUTO_INCREMENT`

When to use surrogate keys ...
- no natural primary key but need to reference rows (foreign keys)
- need to “shorten” composite keys (because of foreign key references)
- for latter, can cause consistency issues (more later)

Check in:
- What are natural, artificial, and surrogate keys?
- Why would surrogate keys be used?