Relational Data Model

• a data model in which all data is modelled as relations (tables)
  • a way of looking at data
  • a prescription for a way of
    • representing data
    • manipulating data (relational algebra)
    • representing integrity constraints

A Relation

A relation contains a SET of tuples

PARTS(Name; Price: Real; Category; Manufacturer: String)

<table>
<thead>
<tr>
<th>Name</th>
<th>Price ($)</th>
<th>Category</th>
<th>Manufacturer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>19.99</td>
<td>gadgets</td>
<td>GizmoWorks</td>
</tr>
<tr>
<td>Power gizmo</td>
<td>29.99</td>
<td>gadgets</td>
<td>GizmoWorks</td>
</tr>
<tr>
<td>SingleTouch</td>
<td>149.99</td>
<td>photography</td>
<td>Canon</td>
</tr>
<tr>
<td>MultiTouch</td>
<td>203.99</td>
<td>household</td>
<td>Hitachi</td>
</tr>
</tbody>
</table>

Integrity

• restrictions on data defined by users
  • on individual tables
    • age > 18; salary < 100k
  • on more than one table
    • if budget < 10M then salary < 50k
  • implicit in the data model

Integrity Constraints (ICs)

• IC: condition that must be true for any instance of the database
  • e.g., domain constraints
    • Each attribute has values taken from a domain. ICs are specified when schema is defined.

Primary and foreign keys

<table>
<thead>
<tr>
<th>E_id</th>
<th>E_name</th>
<th>Dept_id</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>E1</td>
<td>Smith</td>
<td>D1</td>
<td>40K</td>
</tr>
<tr>
<td>E2</td>
<td>John</td>
<td>D1</td>
<td>42K</td>
</tr>
<tr>
<td>E3</td>
<td>Stella</td>
<td>D2</td>
<td>39K</td>
</tr>
<tr>
<td>E4</td>
<td>Art</td>
<td>D3</td>
<td>35K</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dept_id</th>
<th>Dept_name</th>
<th>Budget</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>Marketing</td>
<td>10M</td>
</tr>
<tr>
<td>D2</td>
<td>Development</td>
<td>12M</td>
</tr>
<tr>
<td>D3</td>
<td>Research</td>
<td>5M</td>
</tr>
</tbody>
</table>

More Integrity Constraints

• Key constraints: each tuple must be distinct. A key is a subset of fields that uniquely identifies a tuple (superkey), and for which no subset of the key has this property.

• Referential integrity constraints: a field in one relation may refer to a tuple in another relation by including its key (foreign key). The referenced tuple must exist in the other relation for the database instance to be valid.

• Typically, a relation may have several candidate keys one of which is chosen as the primary key.
Relational Operations

- **Selection** (σ) Selects a subset of rows from relation.
- **Projection** (Π) Deletes unwanted columns from relation.
- **Join** (⋈) Allows us to combine two relations.
- **Set-difference** (−) Tuples in reln. 1, but not in reln. 2.
- **Union** (∪) Tuples in reln. 1 and in reln. 2.
- **Aggregation** (SUM, MIN, etc.) and GROUP BY

Example

```
SELECT S.sname
FROM Reserves R, Sailors S
WHERE R.sid=S.sid AND R.bid=100 AND S.rating>5
```

Equality Joins With One Join Column

```
SELECT *
FROM Reserves R, Sailors S
WHERE R.sid=S.sid
```

- In algebra: R ® S.
- Most frequently used operation; very costly operation.

Join Example

```
sid  sname  rating  age
22  dusty  7      45.0
28  puppy  9      35.0
31  lubber 8      55.5
44  guppy  5      35.0
58  rusty  10     35.0
```

```
sid  bid    day     rname
31  101    10/11/96 lubber
58  103    11/12/96 dustin
```

Join Example

```
sid  sname  rating  age
22  dusty  7      45.0
28  puppy  9      35.0
31  lubber 8      55.5
44  guppy  5      35.0
58  rusty  10     35.0
```

```
sid  bid    day    rname
31  101    10/11/96 lubber
58  103    11/12/96 dustin
```

Simple Nested Loops Join

- For each tuple in the outer relation R, we scan the entire inner relation S.

```
foreach tuple r in R do
    foreach tuple s in S do
        if r.sid == s.sid then add <r, s> to result
```