Databases and relations

Lukas Prokop

26th of January 2012

Contents

1 Databases **1**
1.1 Information Systems 1
1.2 Database design 1
1.3 DataBase Management System (DBMS) 2
1.4 Database Languages 2
1.5 Roles in a DBMS 2
1.6 Protections 2
1.7 Data Manipulation Facilities 3

2 Relational Database Model **3**
2.1 Terminology 3
2.2 Keys 3
2.3 Constraints 3
2.3.1 Integrity constraints 3

3 Normalization **3**
3.1 Anomalies 3
3.2 Terminology 4
3.3 1. Normal Form (1NF) 4
3.4 2. Normal Form (2NF) 4
3.5 3. Normal Form (3NF) 4

4 Relational Algebra **4**
4.1 Selection σ 4
4.2 Projection π 5
4.3 Rename ρ 5
4.4 Natural join $\bowtie$ 5
4.5 Division $\div$ 5
4.6 Union $\cup$ 5
4.7 Intersection $\cap$ 5
4.8 Difference $\setminus$ 5
4.9 NULL values 5
4.10 Relational algebra example 6

5 Glossary **6**

6 Entity Relationships **6**
6.1 Entities 6
6.2 Relationships 6
6.3 Constraints 6
6.3.1 Cardinality constraints 6
6.4 Terminology 6
6.5 Entity Relationship Diagrams 7

1 Databases

Note. Based on lectures "Databases 1", "Databases 2", "New Information systems" and "Object-oriented analysis and design" I visited.

1.1 Information Systems

An *information model* describes the problem (ie. application) domain, which gets reduced in the software development process to the domain of interest. This domain captures only relevant parts of the application and provides a focus.

A **data object** is a collection of attribute-value pairs that model an individual entity. A database is an information model of the real world.

1.2 Database design

Database design consists of three phases:

- Conceptual design (DBMS independent)
- Logical design (transactions independent)
- Physical design (schema for a particular DBMS and transactions)

Conceptual design relates in some kind to the requirement analysis phase in a software development process. The goals of a conceptual design are:

- to identify data objects

- to identify all attributes of data objects
- to identify relationships between data objects
- to identify integrity constraints

Quality attributes for the conceptual design are:

- integrability
- learnability and memorability
- completness
- visualability

1.3 DataBase Management System (DBMS)

A DBMS' goal is to keep its data integrity while applying operations to create, modify and access data in the database. A DBMS represents a data model and separates a database structure (*database schema*) from its contents (*database*).

- A *data model* consists of a collection of data structure types, a collection of operators and rules and a collection of general integrity rules (implicitly and explicitly).
- *Database transactions* have to be ACID (atomic, consistent, isolated and durable). Transactions are a logical unit of work and are an application-specified sequence of data manipulation operations to avoid inconsistent state after an error half-way through the sequence. Therefore a transaction is either executed in its entirety or totally aborted.

1.4 Database Languages

Data Description Language (DDL) A DDL is used to define database schemas. Its syntax is a collection of statements for the description of data types.

Data Manipulation Language (DML) A DML allows the user to manipulate data objects and relationships between them. It's a collection of operators that can be applied to data objects. Manipulation includes insertion, deletion and modification of tuples.

Query Language (QL) A QL uses queries as parameters to select and project data set from one or several relations.

There are different approaches to use DDL/DML/QLs:

Self-Contained Language For example SQL. It's command-oriented, english-like and can become complex.

Embedded Host Language For example C++. The program uses a DML (eg. SQL) from an application backend perspective to manipulate the data. A possible approach is to make simple queries (QL) to the database and resolve its relationships within the application programming language.

1.5 Roles in a DBMS

Database Administrator (DBA) A DBA is capable of using a DDL to create and maintain a consistent set of database schemas to satisfy the needs of the problem domain.

Application Developers (AD) An AD uses an interface from the application to communicate to the database using a DML. He develops specific functions around the database and manipulates the data.

End User The end user does not get in contact with the structure of a database and only gets an user interface to enter data. The application resolves that data to DML queries.

1.6 Protections

Data protection is crucial to an application.

Physical Protection It addresses protection against physical loss of data. This loss can be caused by natural disasters, theft or accidental damage to equipment.

Operational Protection Operational Protection prevents human errors on the databases' integrity. Integrity constraints specified by a DBA in the database schema can help to protect from these errors.

Authorisational Protection Read and write access to the databases should only be given to authorised users to ensure confidentiality and correctness.

1.7 Data Manipulation Facilities

Can be done using a DML.

- insert or add new tuples into the relation
- delete or erase existing tuples from the relation
- modify or change data in an existing relation. assignment or computation (readed, computed, stored)

2 Relational Database Model

2.1 Terminology

attributes distinct columns in a relation. Each attribute has a associated domain.

cardinality number of rows in a table

degree number of columns in a table

domain a set of different values with the same property types ("data type"). The NULL value is possible if not explicitly stated otherwise.

relation a table (visual view) with a unique name. a two dimensional, inhomogeneous matrix (mathematical view). The ordering of tuples in a relation is important.

schema schema of a relation refers to the permanent characteristics of a relation

tuple the set of values in a row. Those values are instances of the attribute domain.

2.2 Keys

The definition of a relation includes the definition of attributes, domains, a primary key and additional constraints. The definition of primary keys per relation allows us to simply refer entries in other relations.

Keys are unique.

Keys shall be kept small to reduce the foreign key table in terms of storage size. Furthermore a small key allows fast and efficient index lookups.

Superkeys are attributes identifying a tuple uniquely.

A *candidate key* is a superkey with a minimal set of attributes and candidate for a primary key.

One of the candidate keys is declared as *primary key* and satisfies the relation constraint and does not contain a NULL value.

A *foreign key* is an attribute which holds the primary key of another relation.

Possible notation
<relation name>: (<attribute name 1>, <attribute2>, <attribute3>)
relation: (**primary key**, attribute, *foreign key*)
Customer: (**cid**, name, age, *userRightsId*)

2.3 Constraints

2.3.1 Integrity constraints

Integrity is the reliability through imposition of constraints during data manipulation (data integrity).

- implicit

  entity integrity A primary key cannot be set to NULL.

  referential integrity A foreign key can have only two possible values either the relevant primary key or NULL value.

- explicit (examples are given)

  domain upper/lower limits for integer values

  tuple if attr1 == x then y has to be < 50

  relation no duplicates as primary keys

3 Normalization

3.1 Anomalies

Update anomaly Data inconsistency or loss of data integrity can arise from data redundancy or repetition and partial update

Insertion anomaly Data cannot be added because some other data is absent.

Deletion anomaly Data maybe unintentionally lost through the deletion of other data.

3.2 Terminology

- A *determinant* is an attribute or a set of non-redunant attributes which can act as a unique identifier of (a set of) another attribute of a given value. So if two tuples have the same value for A, it's required to have the same value for B in both tuples. $A \to B$ ("A determines B" or "A is a determinant of B"). The opposite is $A \nrightarrow B$ ("A does not determine B").
- If A determines B then B is *functionally dependent* on A. *Full functional dependency* is given if A determines B and no subset of A determines B.
- The relation $(A \to B \land B \to C) \Rightarrow A \to C$ describes transitivity. Transitivity is called *indirect dependency*.

3.3 1. Normal Form (1NF)

A relation in 1NF has to be free of repeating groups. Even though this definition is considered differently by various researchers, it basically means that no value contains more than one value and there are no duplicate tuples. Codd also makes references to the concept of atomicity:

> values in the domains on which each relation is defined are required to be atomic with respect to the DBMS.

3.4 2. Normal Form (2NF)

A table is in 2NF if it is in 1NF and no attribute—that is not a part of any candidate key of the table—is dependent on any proper subset of any candidate key of the table.

So 2NF can be achieved by taking a relation in 1NF and eliminating all partial key dependencies.

3.5 3. Normal Form (3NF)

A table is in 3NF if it is in 2NF and every non-key attribute is fully and directly dependent on the primary key. So 3NF can be achieved by taking a relation in 2NF and eliminating all indirect dependencies on the primary key.

4 Relational Algebra

σ	selection
π	projection
$\cup$	union
$-$	difference
$\times$	cartesian product
ρ	rename
$\bowtie$	join
$\ltimes$	left semi join
$\rtimes$	right semi join
$\cap$	intersection
$\div$	division

Relation Algebra is a procedural relationally complete language. A language that can define any relation definable in relational calculus is relationally complete. Using this algebra we perform set operations on relations.

4.1 Selection σ

A selection is a unary operation to select tuples from relations.

```
select <source relation name>
where <predicate>
giving <result relation name>
```

- `predicate` might be any kind of propositional formula using the logical operators AND, OR or NOT. NOT has to be prefixed; other operators get infixed. The precedence is defined as followed: NOT, AND, OR.
- intension (schema) stays the same.
- the result is a horizontal subset of source.
- `giving` clause can be ommitted to use the result inline.

4.2 Projection π

The projection is an operation to reduce a relation containing containing only specified columns. All duplicate result tuples will be removed.

```
project <source relation name>
over <list of attribute names>
giving <result relation name>
```

The list of attribute names shall be given as comma separated list.

4.3 Rename ρ

The rename operation renames an attribute of a relation to a new name.

```
Rename <attribute name> <target name>
```

4.4 Natural join $\bowtie$

A natural join of two tables combines those tables i.e. extends the tuples of one relation with their counterparts. Two tuples (of each table) will be contactenated if their shared (specified) attributes match. It ommits tuples which don't have an associated tuple in the other relation.

```
join <first relation> AND <second relation>
over <list of attribute names>
giving <result relation name>
```

The list of attribute names shall be given as comma separated list.

4.5 Division $\div$

A division builds sets of tuples that are identical in their non-shared attributes and filters sets, which don't contain all corresponding shared values of the divisor. All shared values are dropped and resulting duplicates get removed.

```
divide <source relation name>
by <divisor relation name>
giving <result relation name>
```

The divisor relation has to share attributes with the source relation.

4.6 Union $\cup$

The union operation combines two tuple-sets and ommits duplicates. Two tuples are union-compatible, if they have identical intensions.

```
<first source relation>
union <second source relation>
giving <result relation>
```

4.7 Intersection $\cap$

The intersection outputs all tuples which are in both source relations.

```
<first source relation>
intersect <second source relation>
giving <result relation>
```

4.8 Difference $\setminus$

This operation outputs all tuples from the first relation, which are not also in the second relation.

```
<first source relation>
minus <second source relation>
giving <result relation>
```

4.9 NULL values

Any comparison involving NULL returns FALSE.

- So in a selection operation tuples with NULL values in condition attributes are omitted in the resulting relation.
- In a projection, the resulting tuples with NULL values are not considered to be duplicates and therefore do not get reduced.
- A join operation with tuples containing NULL in a shared attribute excludes those tuple from the list of tuples to concatenate.
- A division operation will fail if there is a NULL value in the divisor (because not tuple set will match). The result is an empty relation.
- For other operations (union, intersection, difference) tuples containing NULL are considered to be different.

4.10 Relational algebra example

For example, the SQL SELECT statement combines the operators Selection, Projection, Join and the Cartesian Product of relational algebra.

```
select Movie
where m_title = 'x' or m_title = 'y'
giving XYMovie;

project XYMovie
over m_id
giving XYMovieId;

divide Role
by XYMovieId
giving XYRole;

join XYRole and Human
over h_id
giving XYHuman;

project XYHuman
over h_first_name, h_last_name
giving Result;
```

5 Glossary

RDBMS Relational Database Management System

DBMS Database Management System

DDL Data Description Language

DML Data Manipulation Language

DBA Database Administrator

6 Entity Relationships

6.1 Entities

An entity in ER is an instance of a more abstract idea. It's a "thing" or an "object" in the real world that is distinguishable from other objects.

- Each entity occurrence is presented as a number of attributes
- Each attribute has a name and domain.
- Domains specify the set of permitted values for attributes.

An *entity type* is a set of similar entity occurrences. The term "similar" can be perceived as "presenting with different values of the same domains". The model always operates on "entities" and "entity types"; so those terms are used synonymously.

6.2 Relationships

Relationships describe associations between an arbitrary number of entity occurences. Each relationship has a name and may have zero or more attributes. A relationship type is a set of similar relationship instances. The term "similar" can be simply seen as "relationshipy between occurrences of the same entity types, having one and the same name and set of attributes". Accordingly the model operates on "relationship types" so the terms are used synonymously. Pragmatically, we understand the term "relationship" as a description of common properties of a set of relationship instances (name of the relationship, members, attributes).

6.3 Constraints

Cardinality constraints define the number of entity occurrences to which an other entity occurrence can be associated via a relationship instance.

6.3.1 Cardinality constraints

A one-to-one relationship can be visualized by a simple line between the first entity instance and the relationship and another line between the relationship and the second instance.

A one-to-many or many-to-many relationship can be visualized by using a split symbol at the end of the line before it ends into the corresponding instance. The line has to be split into three straight lines.

6.4 Terminology

relationships with total participation
: occurrences of a certain entity must obligatory participate in such relationships (for example, an employee has to participate in a relationship "works_at").

relationships with partial participation occurrences of a certain entity may optionally participate in such relationships (for example, an employee may participate in a relationship "is_manager").

weak entity The primary key of an entity is defined by an attribute and an attribute of a foreign entity.

6.5 Entity Relationship Diagrams

Entity Relationship Diagrams (ER) are a graphical notation for the Entity Relationship Model introduced before.

- Entities are represented by rectangulars.
- Attributes are represented by ellipses with a line to its associated entity
- Relationships are represented by diamonds.
- Attributes of a relationship are represented by ellipses with a line to its associated entity
- Primary key attributes are underlined in the diagram.
- Total participation of an entity in a relationship is represented by double (thick) lines connection the relationship and the entity.