Australian National University

Relational Data Model – Part 1

Schema and State

What is the Relational Data Model?

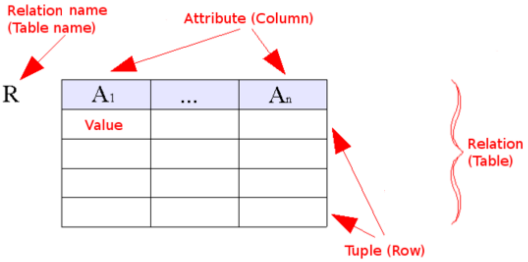
- Introduced by Edgar F. Codd of IBM Research in 1970.

  "A Relational Model for Large Shared Data Banks", Communications of the ACM.

  - A database contains tables (called relations), and each table is made up of columns and rows.
  - Humans have used tables for centuries to keep track of data.

- Used as the standard for relational DBMSs (e.g., Oracle, IBM DB2, Microsofts Access, Microsofts SQL Server, MySQL, postgreSQL, etc.).

Relation



- Correspondence of informal and formal terms:

INFORMAL TERMS	FORMAL TERMS
Table	Relation
Column	Attribute
Data type	Domain
Row	Tuple
Table definition	Relation schema

The Basics

- **Attributes** are used to describe the properties of information. In the relational model, they usually refer to atomic data.

  Example: To capture the information of a person, we can use attributes like Name, Age, Gender, Address and PhoneNumber.

- **Domains** are the sets of all possible values for attributes.

  Example:
  - $\texttt{STRING} = \{A, B, CD, ...\}$;
  - $\texttt{DATE} = \{01/01/2005, 03/07/1978, ...\}$;
  - $\texttt{INT} = \{..., -1, 0, 1, 2, ...\}$.

- Recall that, **Cartesian product** $D_1 \times ... \times D_n$ is the set of all possible combinations of values from the sets $D_1, ..., D_n$.

  Example: Let D_1={book,pen}, D_2={1,2} and D_3={red}. Then

  - $D_1 \times D_2 \times D_3$ ={(book,1,red),(book,2,red),(pen,1,red),(pen,2,red)}

The Basics

- The attributes are StudentID, CourseNo, Semester, Status and EnrolDate.
- The domains of attributes are as follows.
  dom(StudentID)=$\texttt{INT}$; dom(CourseNo)=$\texttt{STRING}$;
  dom(Semester)=$\texttt{STRING}$; dom(Status)=$\texttt{STRING}$;
  dom(EnrolDate)=$\texttt{DATE}$.
- The whole table can be considered as a set {(456, COMP2400, 2016 S2, active, 25/05/2016), (458, COMP1130, 2016 S1, active, 20/02/2016), (459, COMP2400, 2016 S2, active, 11/06/2016)}.

ENROL				
StudentID	CourseNo	Semester	Status	EnrolDate
456	COMP2400	2016 S2	active	25/05/2016
458	COMP1130	2016 S1	active	20/02/2016
459	COMP2400	2016 S2	active	11/06/2016

- Is the above set a subset of

$$\texttt{INT} \times \texttt{STRING} \times \texttt{STRING} \times \texttt{STRING} \times \texttt{DATE}?$$

Answer: Yes.

The Basics

- A **relation schema** has a **relation name** and a list of **attributes**.
- Each attribute is associated with a **domain**.
- A relation schema can be expressed by
  - $R(A_1, ..., A_n)$, or
  - $R(A_1 : dom(A_1), ..., A_n : dom(A_n))$,

  where $A_1, .., A_n$ are attributes of R and $dom(A_i)$ is the domain of A_i.

  Example: The relation schema in the previous example is

  - ENROL(StudentID, CourseNo, Semester, Status, EnrolDate), or
  - ENROL(StudentID: `INT`, CourseNo: `STRING`, Semester: `STRING`, Status: `STRING`, EnrolData: `DATE`).

The Basics

- Let $R(A_1, ..., A_n)$ be a relation schema.
- A **tuple** in R is a list t of values, i.e., $t \in dom(A_1) \times ... \times dom(A_n)$.

  Example: The previous example has the following tuples:

  - (456, COMP2400, 2016 S2, active, 25/05/2016)$\in$ $\texttt{INT} \times \texttt{STRING} \times \texttt{STRING} \times \texttt{STRING} \times \texttt{DATE}$.
  - (458, COMP1130, 2016 S1, active, 20/02/2016)$\in$ $\texttt{INT} \times \texttt{STRING} \times \texttt{STRING} \times \texttt{STRING} \times \texttt{DATE}$.
  - (459, COMP2400, 2016 S2, active, 11/06/2016)$\in$ $\texttt{INT} \times \texttt{STRING} \times \texttt{STRING} \times \texttt{STRING} \times \texttt{DATE}$.

- A **relation** $r(R)$ is a set of tuples $r(R) \subseteq dom(A_1) \times ... \times dom(A_n)$.

  Example: The previous example has the following relation:

  - $r(\text{ENROL}) \subseteq \texttt{INT} \times \texttt{STRING} \times \texttt{STRING} \times \texttt{STRING} \times \texttt{DATE}$.

The Basics

- A **relational database schema** S is
  - a set of relation schemas $S = \{R_1, \ldots, R_m\}$, and
  - a set of integrity constraints IC.
- A **relational database state** of S is a set of relations such that
  - there is just one relation for each relation schema in S, and
  - all the relations satisfy the integrity constraints IC.

The Basics

- Consider a relational database schema StuEnrol that has three relation schemas:
  - Student(StudentID, Name, DoB, Email).
  - Course(No, Cname, Unit);
  - Enrol(StudentID, CourseNo, Semester, Status, EnrolDate);

Student			
StudentID	Name	DoB	Email

Course		
No	Cname	Unit

Enrol				
StudentID	CourseNo	Semester	Status	EnrolDate

- That is, StuEnrol={Student, Course, Enrol}.

The Basics

- Relational Database State – Example

STUDENT			
StudentID	Name	DoB	Email
456	Tom	25/01/1988	tom@gmail.com
458	Peter	23/05/1993	peter@gmail.com
459	Fran	11/09/1987	frankk@gmail.com

COURSE		
No	Cname	Unit
COMP1130	Introduction to Advanced Computing I	6
COMP2400	Relational Databases	6

ENROL				
StudentID	CourseNo	Semester	Status	EnrolDate
456	COMP2400	2016 S2	active	25/05/2016
458	COMP1130	2016 S1	active	20/02/2016
459	COMP2400	2016 S2	active	11/06/2016

Relational Data Model – Part 2

Integrity Constraints

Integrity Constraints over Relations

- Constraints are **conditions** that must hold on *all* relations in a database state.
- The *main types* of constraints in the relational data model include:
  1. **Domain constraints**;
  2. **Key constraints**;
  3. **Entity integrity constraints**;
  4. **Referential integrity constraints**.

(1) Domain Constraints

- Every value in a tuple must be from the **domain of its attribute**.
  - INT
  - VARCHAR
  - DATE
  - SMALLINT
  - NOT NULL

(2) Key Constraints - Observation

- We observe that: **data does not occur independently from one another within individual relations.**

- No two students have the same student ID:

STUDENT			
StudentID	Name	DoB	Email
456	Tom	25/01/1988	tom@gmail.com
458	Peter	23/05/1993	peter@gmail.com
459	Fran	11/09/1987	frankk@gmail.com
...	...	...	...

- No two enrolments have the same student ID, the same course number in the same semester:

ENROL				
StudentID	CourseNo	Semester	Status	EnrolDate
456	COMP2400	2016 S2	active	25/05/2016
458	COMP1130	2016 S1	active	20/02/2016
459	COMP2400	2016 S2	active	11/06/2016
...	...	...	...	...

(2) Key Constraints - Definitions

- Let $R(A_1, \ldots, A_n)$ be a relation schema.
- A **superkey** SK of R is a subset of attributes of R, i.e., $SK \subseteq \{A_1, \ldots, A_n\}$, such that
  - no two distinct tuples in $r(R)$ can have the same value for SK.
- A superkey SK of R is **minimal** if there is no other superkey $SK' \subset SK$ held on R. A minimal superkey is also known as a **candidate key**.
- A **primary key** PK of R is a minimal superkey of R, (i.e., a primary key is one of the candidate keys). If a relation has only one candidate key then that would be the primary key.

(2) Key Constraints - Example

STUDENT			
StudentID	Name	DoB	Email
456	Tom	25/01/1988	tom@gmail.com
458	Peter	23/05/1993	peter@gmail.com
459	Fran	11/09/1987	frankk@gmail.com
460	Tyrion	11/09/1987	tyrion@hotmail.com

- Is {DoB} a superkey of STUDENT? **No!**
- Is {StudentID, DoB} a superkey of STUDENT? **Yes!**
- Is {StudentID, DoB} a candidate key of STUDENT? **No!**
- Is {StudentID} a candidate key of STUDENT? **Yes!**
- Can {StudentID} be chosen as a primary key of STUDENT? **Yes!**
- Can {DoB} be chosen as a primary key of STUDENT? **No!**

(2) Key Constraints - Example

ENROL				
StudentID	CourseNo	Semester	Status	EnrolDate
456	COMP2400	2016 S2	active	25/05/2016
458	COMP1130	2016 S1	active	20/02/2016
459	COMP2400	2016 S2	active	11/06/2016
458	COMP1130	2015 S1	inactive	20/02/2015

- Is {CourseNo, Semester} a superkey of ENROL? **No!**
- Is {StudentID, CourseNo, Semester} a candidate key of ENROL? **Yes!**
- Can {StudentID, CourseNo} be chosen as a primary key of ENROL? **No!**

(3) Entity Integrity Constraints

- Specifying a primary key also invokes the entity integrity constraint.
- **null** is a special value, which represents the value of an attribute that may be unknown or inapplicable.
- The **entity integrity constraint** states that **no primary key value can be NULL**.
  - This is because primary key values are used to *identify* individual tuples in a relation.
- **Note:** Other attributes of R may be constrained to disallow null values, even though they are not attributes in the primary key.

(3) Entity Integrity Constraints – Example

- If STUDENTID is specified as the primary key of STUDENT, then the following relation violates the entity integrity constraint.

STUDENT			
StudentID	Name	DoB	Email
456	Tom	25/01/1988	tom@gmail.com
NULL	Peter	23/05/1993	peter@gmail.com
459	Fran	11/09/1987	frankk@gmail.com

- How about the case when EMAIL is the primary key of STUDENT?

Answer: The relation does not violate the entity integrity constraint.

(4) Referential Integrity Constraints - Observation

- We observe that: **data does not occur independently from one another across relations.**
- Every course number appearing in ENROL must exist in COURSE:

STUDENT			
StudentID	Name	DoB	Email
456	Tom	25/01/1988	tom@gmail.com
458	Peter	23/05/1993	peter@gmail.com
459	Fran	11/09/1987	frankk@gmail.com

COURSE		
No	Cname	Unit
COMP1130	Introduction to Advanced Computing I	6
COMP2400	Relational Databases	6

ENROL				
StudentID	CourseNo	Semester	Status	EnrolDate
456	COMP2400	2016 S2	active	25/05/2016
458	COMP1130	2016 S1	active	20/02/2016
459	COMP2400	2016 S2	active	11/06/2016

(4) Referential Integrity Constraints - Definition

- We use $t[A]$ to denote the value of attribute A in tuple t.

  Example: For the tuple t=(459,Fran,11/09/1987,frankk@gmail.com), t[Name]=Fran and t[DoB]=11/09/1987.
- A **referential integrity constraint** specifies a reference between **two** relations, while the previous constraints involve **only one** relation.
- Let R_1 and R_2 be relation schemas in a database schema S, and R_2 has the primary key $\{B_1, \ldots, B_n\}$.
- A **foreign key** on R_1 is a statement $[A_1, \ldots, A_n] \subseteq R_2[B_1, \ldots, B_n]$ restricting states of S to satisfy the following property:
  - for each tuple $t \in r(R_1)$ there exists a tuple $t' \in r(R_2)$ with $t[A_i] = t'[B_i]$ for $i = 1, \ldots, n$.
- R_1 is called the **referencing relation** and R_2 is called the **referenced relation**.

(4) Referential Integrity Constraints – Example

- What foreign keys can be established in the database STUENROL?

STUDENT			
<u>StudentID</u>	Name	DoB	Email
456	Tom	25/01/1988	tom@gmail.com
458	Peter	23/05/1993	peter@gmail.com
459	Fran	11/09/1987	frankk@gmail.com

COURSE		
<u>No</u>	Cname	Unit
COMP1130	Introduction to Advanced Computing I	6
COMP2400	Relational Databases	6

ENROL				
<u>StudentID</u>	<u>CourseNo</u>	<u>Semester</u>	Status	EnrolDate
456	COMP2400	2016 S2	active	25/05/2016
458	COMP1130	2016 S1	active	20/02/2016
459	COMP2400	2016 S2	active	11/06/2016

(4) Referential Integrity Constraints – Example

- In this case, we can establish the following foreign keys on ENROL:
  1. [CourseNo]$\subseteq$ COURSE[No];
  2. [StudentID]$\subseteq$ STUDENT[StudentID].

- This database state satisfies the above two foreign keys because
  - for each tuple t_1 in ENROL, there is a tuple t_2 in COURSE such that the CourseNo value in t_1 is the same with the No value in t_2;
  - for each tuple $t_1^{'}$ in ENROL, there is a tuple $t_2^{'}$ in STUDENT such that the StudentID value in $t_1^{'}$ is the same with the StudentID value in $t_2^{'}$.

(4) Referential Integrity Constraints – Question

- If the database STUENROL is slightly changed as follows, does this database still satisfy the foreign keys in the previous example?

STUDENT			
StudentID	Name	DoB	Email
456	Tom	25/01/1988	tom@gmail.com
458	Peter	23/05/1993	peter@gmail.com
459	Fran	11/09/1987	frankk@gmail.com

COURSE		
No	Cname	Unit
COMP1130	Introduction to Advanced Computing I	6
COMP2400	Relational Databases	6

ENROL				
StudentID	CourseNo	Semester	Status	EnrolDate
456	COMP2400	2016 S2	active	25/05/2016
458	COMP1130	2016 S1	active	20/02/2016
459	COMP2600	2016 S2	active	11/06/2016

(4) Referential Integrity Constraints – Question

Answer: **The following database does not satisfy the foreign key of ENROL: [CourseNo]$\subseteq$ COURSE[No].**

STUDENT			
StudentID	Name	DoB	Email
456	Tom	25/01/1988	tom@gmail.com
458	Peter	23/05/1993	peter@gmail.com
459	Fran	11/09/1987	frankk@gmail.com

COURSE		
No	Cname	Unit
COMP1130	Introduction to Advanced Computing I	6
COMP2400	Relational Databases	6

ENROL				
StudentID	CourseNo	Semester	Status	EnrolDate
456	COMP2400	2016 S2	active	25/05/2016
458	COMP1130	2016 S1	active	20/02/2016
459	COMP2600	2016 S2	active	11/06/2016

Constraint Violations

- There are three basic operations that can change a database state:
  - **Insert**: insert one or more new tuples in a relation;
  - **Delete**: delete tuples in a relation;
  - **Update** (or **Modify**): change the values of attributes in existing tuples.
- Whenever these operations are applied, the integrity constraints specified in a database schema **should not be violated**.
- However,
  - Insert may violate ...
  - Delete may violate ...
  - Update may violate ...