The Relational Data Model Operations

SWEN304 / SWEN435
Trimester 1, 2024

Lecturer: Kevin Shedlock
Engineering and Computer Science

Quick Review

- Introduction to the RDM Constraints
  - ✓ Domain constraints, Attribute constraints
  - ✓ Key constraint, Unique constraint
  - ✓ Interrelation constraints
  - ✓ Constraint violation: database updates

- Operations
  - Insert
  - Edit/ Update
  - Modify
  - Delete

Quick Review

- Constraints in a database model system are rules applied to data to enforce integrity and accuracy.
- Constraints are used to check transactions within a system. They ensure that only valid data is entered into the database, preventing inconsistencies and errors.
- Constraints can include primary key constraints to uniquely identify records, foreign key constraints to maintain referential integrity, and check constraints to validate data against predefined conditions.
- The Candidate Key

Candidate Key?

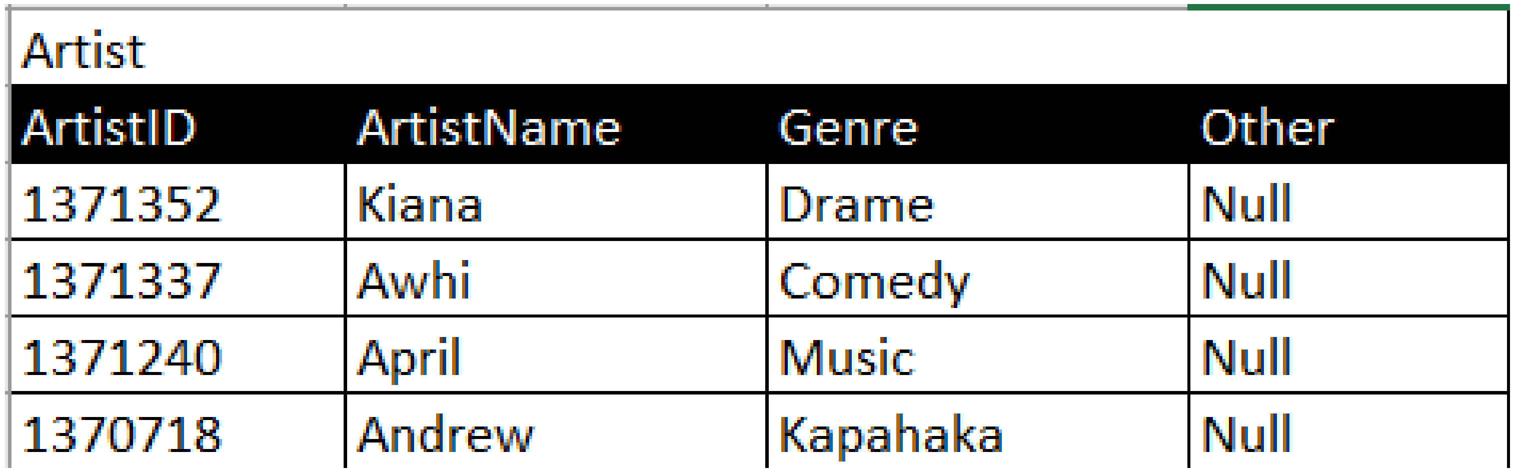
What is the purpose of a candidate key in the definition of a relation?

A candidate key, or simply a key, of a relational database is any set of columns that have a unique combination of values in each row, with the additional constraint that removing any column could produce duplicate combinations of values.

Candidate Key: How does it work?

How does it work?

A primary key acts as a minimal key where the relation can only have one primary key. On the other hand, multiple candidate keys (two or more) can take place in any relationship.



Artist

ArtistID	ArtistName	Genre	Other
1371352	Kiana	Drame	Null
1371337	Awhi	Comedy	Null
1371240	April	Music	Null
1370718	Andrew	Kapahaka	Null

Candidate Key: What does it mean?

What is a candidate key

A candidate key is a specific type of field in a relational database that can identify each unique record independently of any other data. Experts describe a candidate key as, having "no redundant attributes" and being a "minimal representation of a tuple" in a relational database table.

Candidate Keys

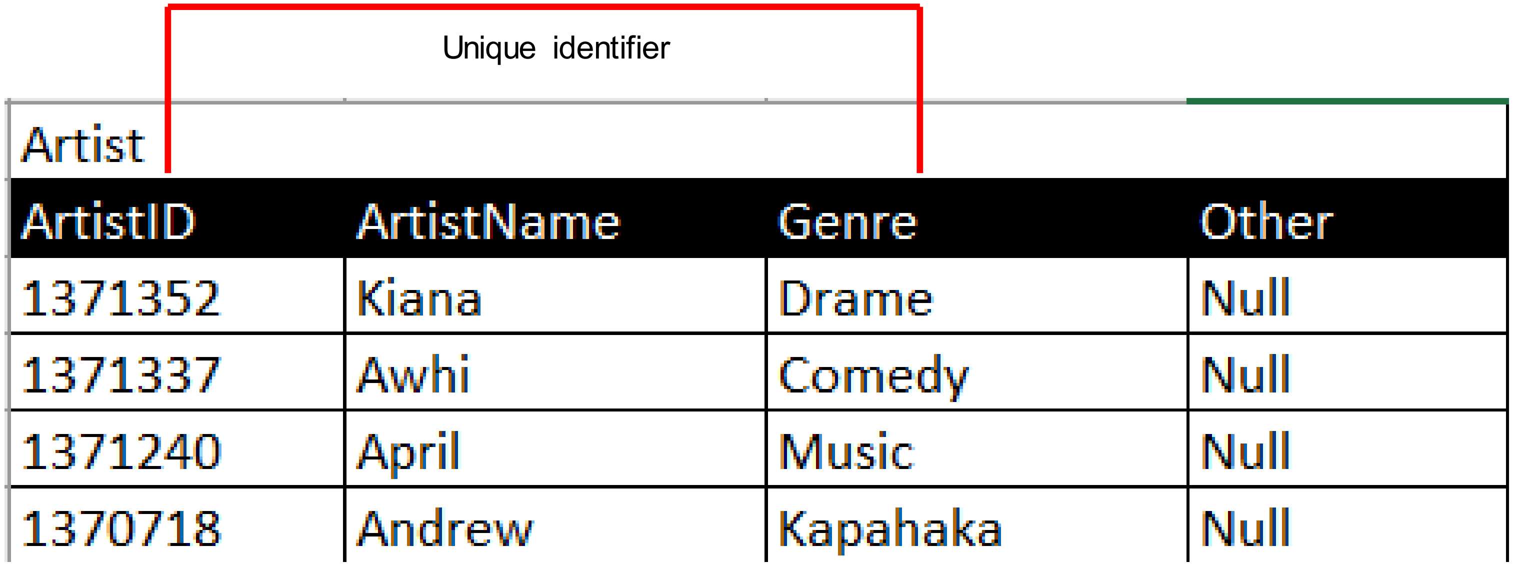
A candidate key is a set of one or more attributes that can uniquely identify a tuple (row) within a relation (table) in a database. In simpler terms, it's a minimal set of attributes that can uniquely identify each record in a table. There are two key properties of candidate keys:

Uniqueness: Each combination of values in a candidate key must uniquely identify a tuple in the relation. In other words, no two tuples can have the same combination of values for a candidate key.

Minimality: Removing any attribute from the candidate key will cause it to lose its uniqueness property. In other words, a candidate key should have the fewest possible attributes while still maintaining uniqueness.

Candidate Keys: Student Artists at TTUPA

For example, consider the table *domain* representing students at TTUPA. A candidate key could be the combination of the *Genre* and their *ArtistID*. Together, these attributes uniquely identify each student studying at TTUPA. However, if we remove either the Genre or the ArtistID from the combination, we would no longer be able to uniquely identify each employee.

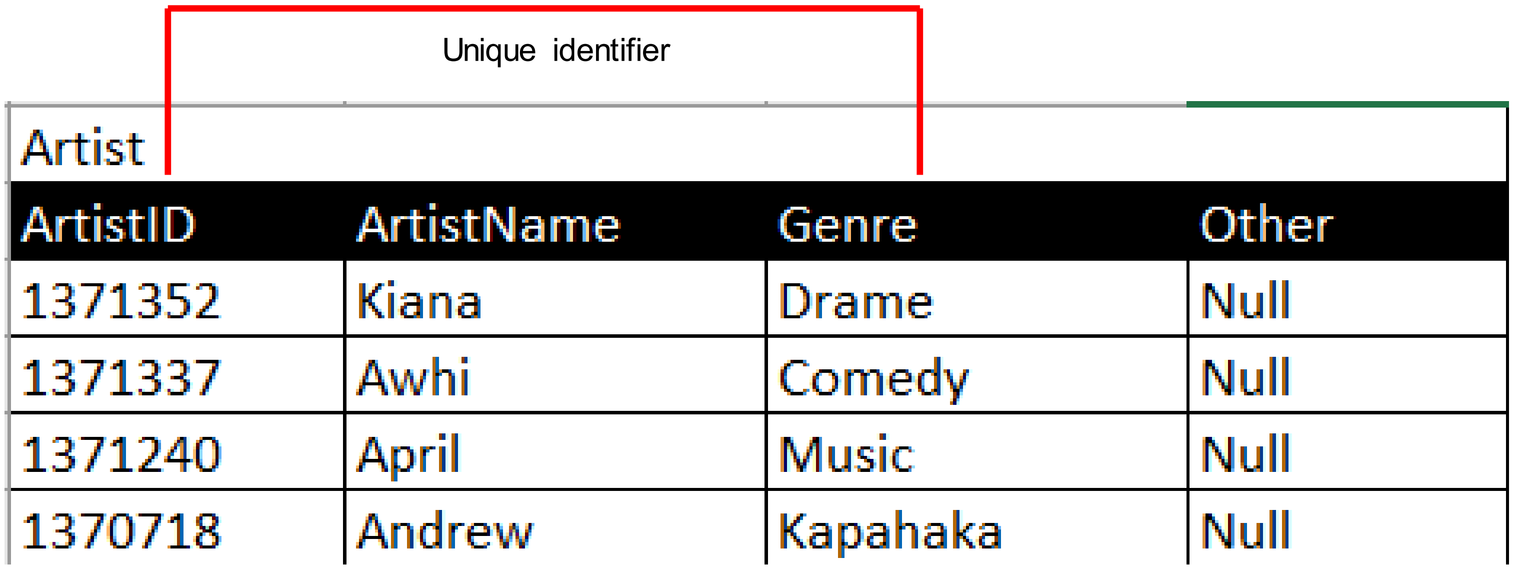

Unique identifier

Artist

ArtistID	ArtistName	Genre	Other
1371352	Kiana	Drame	Null
1371337	Awhi	Comedy	Null
1371240	April	Music	Null
1370718	Andrew	Kapahaka	Null

Candidate Keys: Student Artists at TTUPA

1. In the domain, \`ArtistID\` is a candidate key because it uniquely identifies each Genre in the table. No two Artists have the same \`ArtistID\`.
2. While it's possible that other attributes or combinations of attributes like \`ArtistName\` or \`ArtistName\` + \`Genre\` could also uniquely identify artists, \`ArtistID\` is the minimal set of attributes required to achieve this uniqueness.
3. Thus, the \`ArtistID\` is an example of a candidate key for the “Artists" table.


Unique identifier

Artist

ArtistID	ArtistName	Genre	Other
1371352	Kiana	Drame	Null
1371337	Awhi	Comedy	Null
1371240	April	Music	Null
1370718	Andrew	Kapahaka	Null

Introduction to Operations

- Relational operators is the part of the RDBM that takes two operands, compare their values, and return a Boolean value (true or false).

- They are typically used in conditional expressions to test whether a condition is true, or not. A conditional expression can use arithmetic expressions.

- A central system that deposits and retrieve information responding to a query in some way (similar to the central brain of a system).

Introduction to Operations

Operations is underpinned by algebra expression when applied as a query can be used to retrieve information from the database in some way.

A query can be used to retrieve, manipulate or remove information that resides inside the database by applying the operators.

Here we focus on the update operation which can be used to manipulate the information that resides in the database in some way. There are the INSERT;DELETE; and, MODIFY operations

Relational Database Operations

- Database Management System must implement **update** operations:
  - insert, used to insert one or more tuples into a relation
  - delete, is used to delete tuples and;
  - Modify, (update) is used to change values that currently exist inside tuples
- Database Management System must implement **retrieval** operations:
  - query language
  - Need a well-defined language

Relational Database Instance

- A **database** schema *DBS* as a complex data type defines a finite, but very large number of different database instances

- An instance of the relational database schema $N(S, IC)$ is:

$$db = \{r(N_1), \ldots, r(N_k)\}$$

such that:

  - Each $r(N)$ is an instance of a relation schema $N(R, C)$ in S
  - And db satisfies all constraints in IC

Operations

PostGress (SQL) range across configuration, installation and setup operation, database management, table management, data querying, performance monitoring and, security

Table management considers:

- Create, delete, and modify tables.
- Define columns, data types, and constraints.
- Index columns for improved performance.
- Manage table data insert, update, and delete operations.

Set Operations using Logic

This can be found in groups of information using operations where the logic comes from i.e. Set theory.

Union: $A \cup B$ for the set containing everything in A and B

e.g. {a,b,c} ∪ {a,c,d,e} = {a,b,c,d,e}

Intersection: $A \cap B$ for the set of elements that are in both A and B

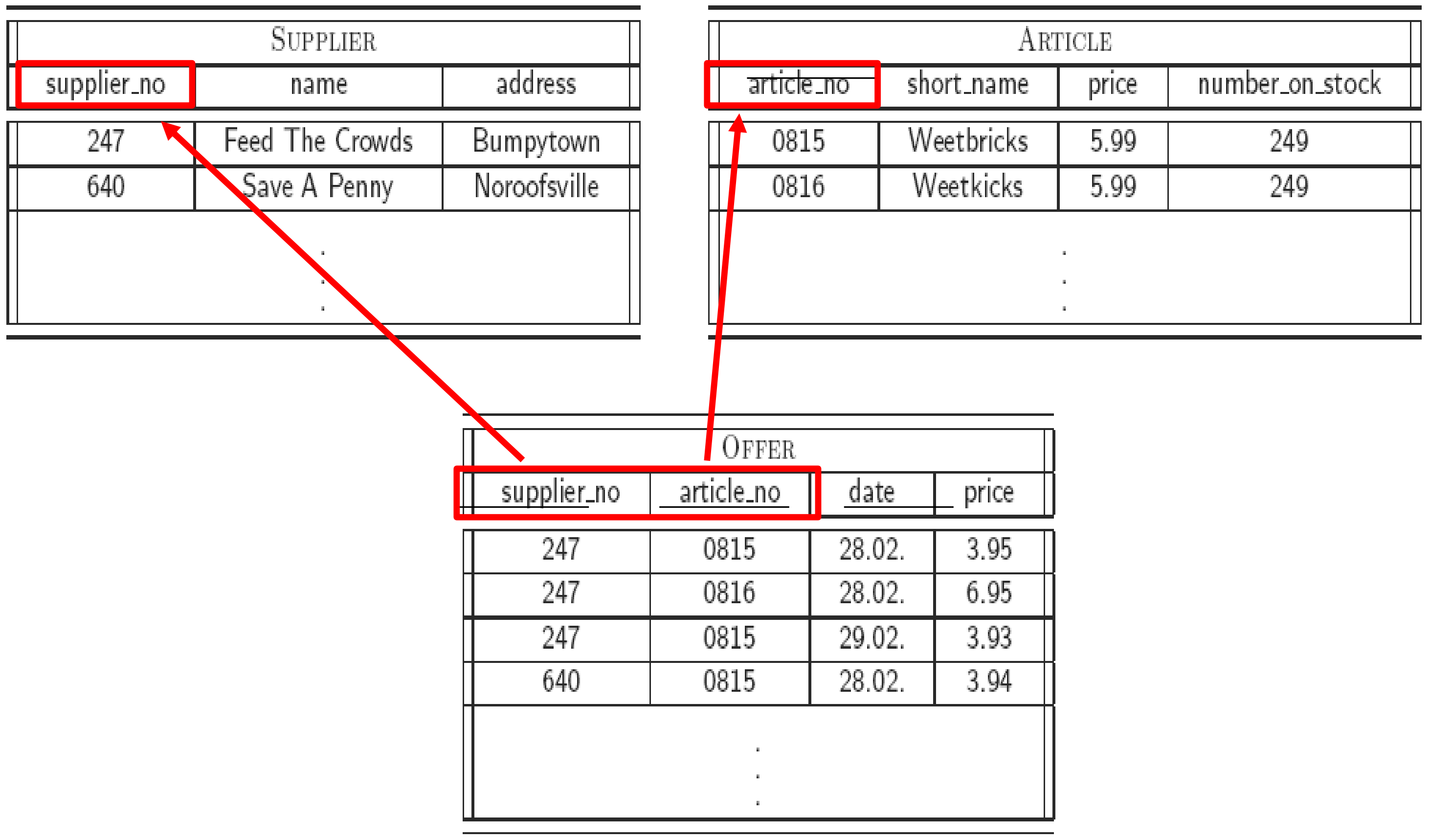
e.g. {a,b,c} ∩ {a,c,d,e} = {a,c}

Difference: $A - B$ for the set of elements from A but not B

e.g. {a,b,c} – {a,c,d,e} = {b}

An Instance of The BOOKSHOP Database

This notation indicates that "Offer" is a subset of "Supplier"
Every item that can be found in the "Offer" set can also be found in the "Supplier" set.



SUPPLIER

supplier_no	name	address
247	Feed The Crowds	Bumpytown
640	Save A Penny	Noroofsville
⋮		

ARTICLE

article_no	short_name	price	number_on_stock
0815	Weetbricks	5.99	249
0816	Weetkicks	5.99	249
⋮			

OFFER

supplier_no	article_no	date	price
247	0815	28.02.	3.95
247	0816	28.02.	6.95
247	0815	29.02.	3.93
640	0815	28.02.	3.94
⋮			

Database Schema and Its Instances

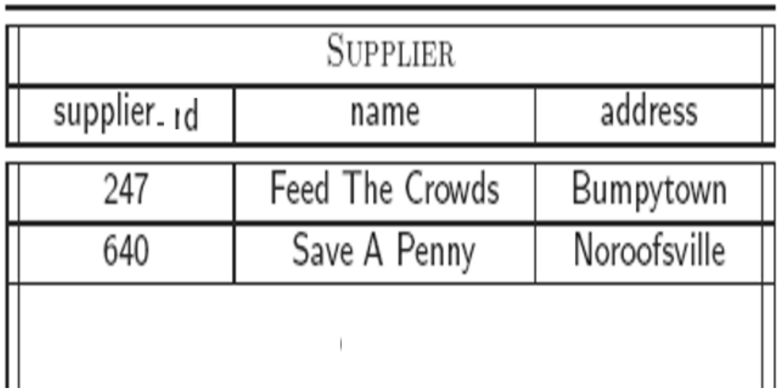
Example: database schema *BOOKSHOP(S, IC)*

- *N* = BOOKSHOP
- *S* = {SUPPLIER({supplier_no, name, address}, {supplier_no}),
  ARTICLE({article_no, short_name, number_on_stock, price}, {article_no}),
  OFFER({supplier_no, article_no, <u>date</u>, price}, {supplier_no, article_no})
  }
- *IC* = { OFFER [supplier_no] $\subseteq$ SUPPLIER [supplier_no]
  OFFER [article_no] $\subseteq$ ARTICLE [article_no]
  }

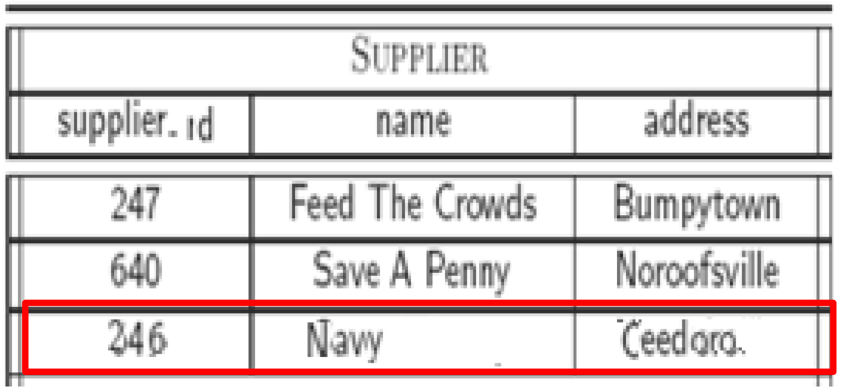
Insert Operations

To insert data using tuples into a database, you typically using PostgreSQL. Here's a general example of how you would insert data into a table using tuples in SQL. Let's say you have a table named `SUPPLIER` with columns `supplier_id`, `name`, and `address`, and you want to insert a tuple (row) into this table:

```
INSERT INTO SUPPLIER (supplier_id, name, address)
VALUES (246, 'Navy','Ceedoro');
```



SUPPLIER		
supplier_id	name	address
247	Feed The Crowds	Bumpytown
640	Save A Penny	Noroofsville



SUPPLIER		
supplier_id	name	address
247	Feed The Crowds	Bumpytown
640	Save A Penny	Noroofsville
246	Navy	Ceedoro.

Inserts and Constraint Violations

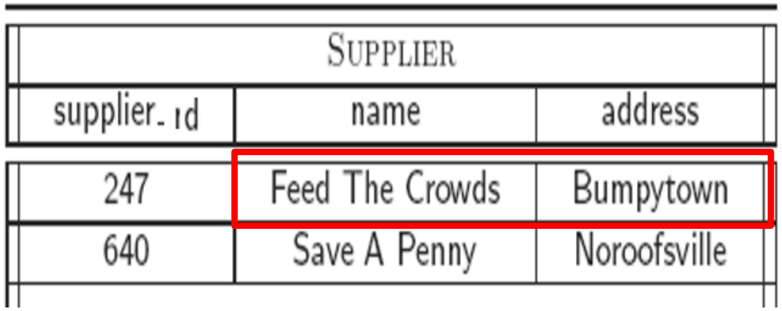
- Inserting a new tuple could violate:
  - Attribute/domain constraints
    (a value is not of the right type or in the required range)
  - Uniqueness constraints
    (the values of the key attributes duplicate another tuple)
  - Not Null constraints
    (an attribute has the value null when it shouldn't)
  - Referential Integrity constraints
    (the values of the attributes of a foreign key do not match any tuple in the other relation)

- Response:
  - **Reject** the operation – there is no change that the DBMS system could safely make to resolve the inconsistency

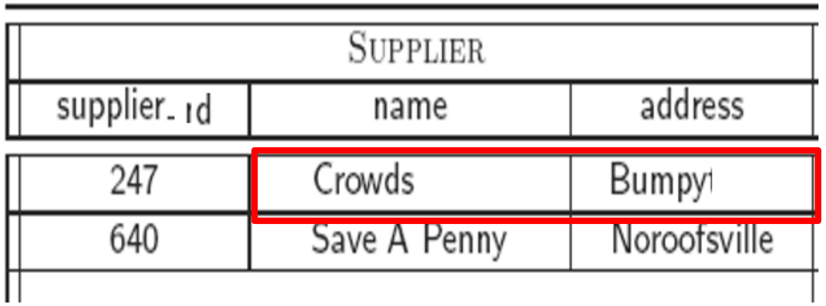
Modify Operations

To modify a tuple (row) in the `SUPPLIER` table you would use the `UPDATE` statement. For argument, to Modify the Feeds the tuples Feed The Crowds and Bumpytown

```
UPDATE SUPPLIER
SET name = ‘Crowds’, address = ‘Bumpy’
WHERE supplier_id = 247;
```



SUPPLIER		
supplier_id	name	address
247	Feed The Crowds	Bumpytown
640	Save A Penny	Noroofsville



SUPPLIER		
supplier_id	name	address
247	Crowds	Bumpyt
640	Save A Penny	Noroofsville

Modify and Constraint Violations

- Modifying/updating the values of attributes in a tuple may violate constraints

  - Attribute/domain constraints
    Response: **reject** (like insert)

  - Key constraints (if attribute is part of a key)
    Response: treat as a delete followed by an insert

  - Referential integrity constraints (if attribute is part of a foreign key). Response: **reject** (like insert), or **cascade**, or **set null,** or **set default** (like delete)

DB Updates and Constraints

- No update operation should leave a database in an inconsistent state (with violated constraints)
- A DBMS must take the actions necessary to prevent a constraint violation:
    - **reject**: do not allow the operation
    - **cascade**: propagate the operation by making necessary consequential changes
    - **set null**, or **set default**: reset other values to maintain consistency

Delete Operations

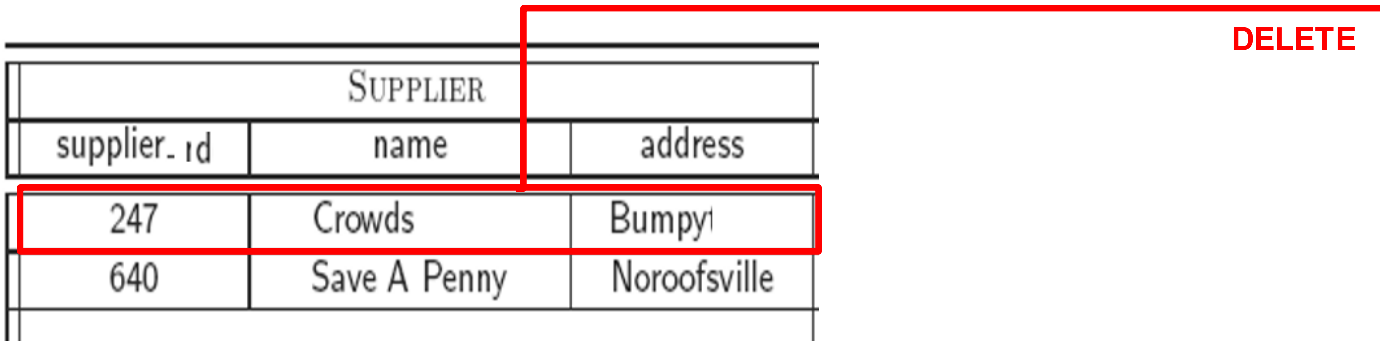
To delete a tuple (row) from the `SUPPLIER` table in PostgreSQL, you would use the `DELETE FROM` statement. Here's how you can do this tasks:

DELETE FROM SUPPLIER

WHERE supplier_id = 247; name = Crowds; address = Bumpy

DELETE FROM SUPPLIER` specifies the table to delete data.

WHERE supplier_id = 'Crowds' is the condition specifying which tuple(s) you want to delete.



SUPPLIER		
supplier_id	name	address
247	Crowds	Bumpy
640	Save A Penny	Noroofsville

Deletes and Constraint Violations

- Deleting a tuple can only violate a referential integrity constraint:
  - If a tuple t is referred to by foreign keys in tuples $t_1, t_2, \dots t_n$ in other relations, then deleting t will make $t_1, t_2, \dots t_n$ inconsistent.
  - For example, deleting a student record from the database means all their grade records will refer to nothing

- There are several options:
  - **Reject** the delete query
  - **Set null / set default**: insert null or a default value in the foreign key attributes of tuples in other relation(s) that refer to t (can't do set null if foreign key attributes are NOT NULL)
  - **Cascade**: delete tuples in other relation(s) that refer to t (appropriate only if the other tuples "*existentially depend*" on t)

Summary: Insert, Modify, Delete

Update operation	Domain / Attribute constraint	Key / Entity integrity constraint,	Referential integrity
insert	reject	reject	reject
delete	no violation	no violation	reject, cascade, set null, set default
modify	reject	reject	reject, cascade, set null, set default

Summary: Insert, Modify, Delete

a) Suppose $Null(N, A) = N$ for all attributes except F in N_2

$r(N_1)$ =

A	B	C	D
a_1	b_1	c_1	d_1
a_2	b_2	c_2	d_2
a_3	b_3	c_3	d_3
a_4	b_3	c_4	d_3
a_5	b_1	c_5	d_3

$r(N_2)$ =

A	B	C	D	E	F
a_1	b_1	c_1	d_1	e_1	f_1
a_1	b_2	c_1	d_2	e_1	f_2
a_2	b_1	c_2	d_1	e_2	f_3
a_1	b_3	c_3	d_1	e_1	ω
a_3	b_1	c_1	d_3	e_2	f_4

b) Suppose now $Null(N_1, C) = Y$ and $Null(N_2, D) = Y$ and $Null(N_2, F) = Y$, and there are some null values in the corresponding columns

A Question for You

- Consider the following database instance

TEXTBOOK			
Title	**<u>ISBN</u>**	**Pcode**	**Pnum**
COD	1111	COMP	203
FDBS	2222	SWEN	ω

COURSE		
<u>Pcode</u>	**<u>Pnum</u>**	**Pname**
COMP	203	CO
SWEN	304	DBS

- Should a DBMS reject the following update operation?
  `UPDATE TEXTBOOK SET PNum = 304 WHERE ISBN = 2222;`

- Should a DBMS reject the following update operation?
  `UPDATE TEXTBOOK SET PNum = 304 WHERE ISBN = 1111;`

Examples of Set Operations

Let $A = \{x, y, z\}$, $B = \{1, 2\}$

Which of the following are correct?

1. $\{y\} \subseteq A$
2. $z \in A$
3. $\{x, y\} \subseteq (A \cup B)$
4. $x \in (A \cap B)$
5. $x \in (A - \{y, z\})$

$(\{x,1\} \cup A) \cap B = ?$