Обмеження / Constraints

Для кожного відношення існують деякі умови, які повинні для нього виконуватися. Ці умови називаються **реляційними обмеженнями цілісності**. Існує три основних обмеження цілісності:

- ключів;
- доменів;
- посилань.

Every relation has some conditions that must hold for it to be a valid relation. These conditions are called **Relational Integrity Constraints**. There are three main integrity constraints:

- Key constraints;
- Domain constraints;
- Referential integrity constraints.

Обмеження ключів / Key constraints

У відношенні повинна бути принаймні одна мінімальна підмножина атрибутів, яка може однозначно ідентифікувати кортеж. Ця мінімальна підмножина атрибутів називається **ключем** для цього відношення. Якщо існує декілька таких мінімальних підмножин, вони називаються **потенційними ключами**.

There must be at least one minimal subset of attributes in the relation, which can identify a tuple uniquely. This minimal subset of attributes is called **key** for that relation. If there are more than one such minimal subsets, these are called **candidate keys**.

Обмеження ключів визначають, що:

- у відношенні з ключовим атрибутом, два кортежі не можуть мати ідентичні значення для ключових атрибутів.
- ключовий атрибут не може мати значення NULL.

Обмеження ключів також називаються обмеженнями сутності.

Key constraints force that:

- in a relation with a key attribute, no two tuples can have identical values for key attributes.
- a key attribute can not have NULL values.

Key constraints are also referred to as Entity Constraints.

Обмеження доменів / Domain constraints

В реальності атрибути мають певні значення. Наприклад, вік може бути лише позитивним цілим числом. Ті самі обмеження спробували застосувати до атрибутів відношення. Кожен атрибут повинен мати певний діапазон значень. Наприклад, вік не може бути менше нуля, а номери телефонів не можуть містити цифри за межами 0-9.

Attributes have specific values in real-world scenario. For example, age can only be a positive integer. The same constraints have been tried to employ on the attributes of a relation. Every attribute is bound to have a specific range of values. For example, age cannot be less than zero and telephone numbers cannot contain a digit outside 0-9.

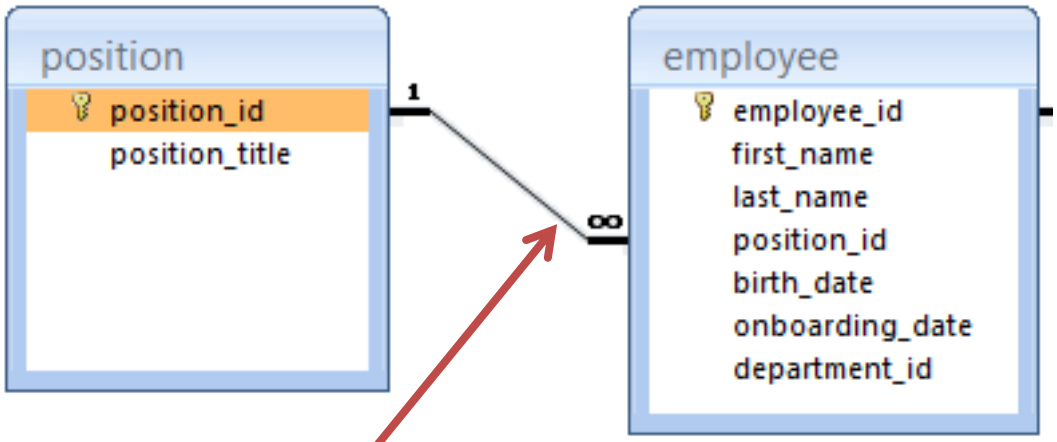
Обмеження цілісності посилань / Referential integrity constraints

Обмеження цілісності посилань оперують з концепцією **зовнішніх ключів**. Зовнішній ключ є ключовим атрибутом відношення, на який можна посилатися в іншому відношенні.

Обмеження цілісності посилань вказує, що якщо відношення посилається на ключовий атрибут іншого або того ж самого відношення, то цей ключовий атрибут повинен існувати.

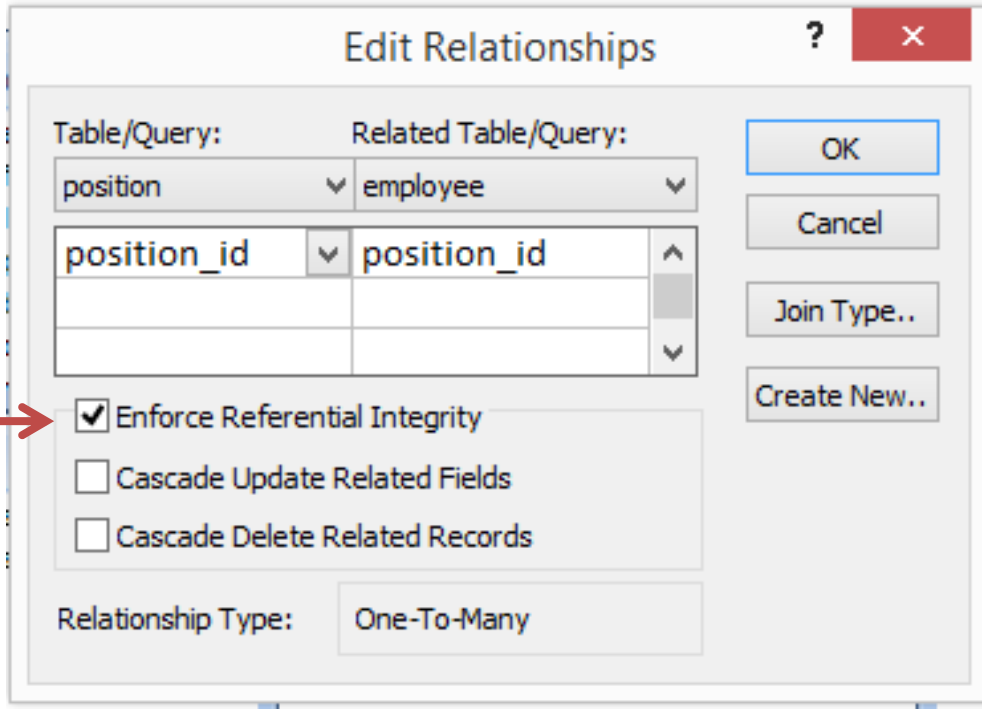
Referential integrity constraints work on the concept of **Foreign Keys**. A foreign key is a key attribute of a relation that can be referred in other relation.

Referential integrity constraint states that if a relation refers to a key attribute of a different or same relation, then that key element must exist.



1) Double-click on the relation

2) Check the first box to enforce referential integrity



Реляційна алгебра / Relational algebra

Реляційна алгебра є процедурною мовою запитів, яка приймає екземпляри відношень та повертає екземпляри відношень. Вона використовує оператори для виконання запитів. Оператор може бути як **унарним**, так і **бінарним**. Вони приймають відношення та повертають також відношення. Реляційна алгебра виконується рекурсивно на відношенні, а проміжні результати також розглядаються як відношення.

Relational algebra is a procedural query language, which takes instances of relations as input and yields instances of relations as output. It uses operators to perform queries. An operator can be either **unary** or **binary**. They accept relations as their input and yield relations as their output. Relational algebra is performed recursively on a relation and intermediate results are also considered relations.

Операція вибору / Select operation

Він вибирає кортежі, які задовольняють даному предикату, з відношення.

Позначення – $\sigma_p(r)$.

Де **σ** означає предикат вибору, а **r** – відношення. **p** – формула логіки препозиції, яка може використовувати з'єднувачі, такі як AND, OR, та NOT. Ці умови можуть використовувати реляційні оператори, такі як: =, ≠, ≥, <,>, ≤.

It selects tuples that satisfy the given predicate from a relation.

Notation − $\sigma_p(r)$.

Where **σ** stands for selection predicate and **r** stands for relation. **p** is prepositional logic formula which may use connectors like AND, OR, and NOT. These terms may use relational operators like: =, ≠, ≥, < , >, ≤.

r = ‘literature’

Book	Pages	Type
SQL Cookbook	218	Technical
Harry Potter and bla-bla-bla	411	Novel
LotR	593	Fantasy
C in Examples	299	Technical
The Walking Dead #39	18	Comics

$$\sigma_{p = \text{Pages} < 300 \text{ AND Type} = \text{‘Comics’}} (r = \text{‘literature’})$$

RESULT?

Операція проекції / Project operation

Вона проектує стовпчик(и), що задовольняє заданому предикату.

Позначення – $\prod_{A1, A2, \ldots, An} (r)$.

Де A_1, A_2, …, A_n – назви атрибутів відношення **r**.

Дубльовані рядки автоматично виключаються, оскільки відношення є множиною.

It projects column(s) that satisfy a given predicate.

Notation − $\prod_{A1, A2, An} (r)$.

Where A_1, A_2, …, A_n are attribute names of relation r.

Duplicate rows are automatically eliminated, as relation is a set.

r = ‘literature’

Book	Pages	Type
SQL Cookbook	218	Technical
Harry Potter and bla-bla-bla	411	Novel
LotR	593	Fantasy
C in Examples	299	Technical
The Walking Dead #39	18	Comics

$$\Pi_{\text{Book, Pages}} (r = \text{‘literature’})$$

RESULT?

Операція об’єднання / Union operation

Вона виконує об’єднання двох відношень і визначається як:
$r \cup s = \{t \mid t \in r \text{ or } t \in s\}$.

Де **r** та **s** – це відношення бази даних або результуючі відношення (тимчасові відношення).

Для того, щоб операція об’єднання була дійсною, r та s повинні мати однакову кількість атрибутів; домени атрибутів повинні бути сумісними; дубльовані кортежі автоматично виключаються.

It performs binary union between two given relations and is defined as:
$r \cup s = \{t \mid t \in r \text{ or } t \in s\}$.

Where **r** and **s** are either database relations or relation result set (temporary relation).

For a union operation to be valid, r and s must have the same number of attributes; attribute domains must be compatible; duplicate tuples are automatically eliminated.

r1 = ‘technical’

Book	Pages	Type
SQL Cookbook	218	Technical
C in Examples	299	Technical

r2 = ‘other’

Book	Pages	Type
Harry Potter and bla-bla-bla	411	Novel
LotR	593	Fantasy
The Walking Dead #39	18	Comics

$r1 \cup r2 = \{t \mid t \in r1 \text{ or } t \in r2\}$

RESULT?

Різниця множин / Set difference

Результатом операції різниці множин є кортежі, які присутні в одному відношенні, але не в другому відношенні.

Позначення – r – s.

Знаходить всі кортежі, які присутні в **r**, але не в **s**.

The result of set difference operation is tuples, which are present in one relation but are not in the second relation.

Notation − r − s.

Finds all the tuples that are present in **r** but not in **s**.

Декартів добуток / Cartesian product

Поєднує інформацію з двох різних відношень в одне.

Позначення – r X s.

Де **r** та **s** – відношення, і їх результат буде визначений як:

r X s = {q t | q ∈ r and t ∈ s}.

Combines information of two different relations into one.

Notation − r X s.

Where **r** and **s** are relations and their output will be defined as:

r X s = {q t | q ∈ r and t ∈ s}.

r1 = ‘books’

Book	Pages
SQL Cookbook	218
LotR	593

r2 = ‘book types’

Book	Type
SQL Cookbook	Technical
LotR	Fantasy

$$r1 \times r2 = \{q\ t \mid q \in r1 \text{ and } t \in r2\}$$

RESULT?

Операція перейменування / Rename operation

Результати реляційної алгебри - це також відношення, але без назви. Операція перейменування дозволяє нам перейменувати вихідне відношення.

Позначення – ρ_x (E).

Де результат виразу **E** зберігається з назвою **x**.

The results of relational algebra are also relations but without any name. The rename operation allows us to rename the output relation.

Notation − ρ_x (E).

Where the result of expression E is saved with name of x.

Реляційне обчислення / Relational Calculus

На відміну від реляційної алгебри, реляційне обчислення є не процедурною мовою запитів, тобто вона визначає, що робити, але ніколи не пояснює, як це зробити.

In contrast to Relational Algebra, Relational Calculus is a non-procedural query language, that is, it tells what to do but never explains how to do it.

Реляційне обчислення кортежів / Tuple Relational Calculus (TRC)

Фільтрація кортежів за змінними
Filtering variable ranges over tuples

$$\{ T \mid Condition \}$$

Повертає усі кортежі **T**, що задовольняють умові
Returns all tuples **T** that satisfies a condition

$$\{ T.name \mid Supplier(T) \text{ AND } T.id = \text{‘3’} \}$$

Яким буде результат?
What will be the output?

Реляційне обчислення доменів / Domain Relational Calculus (DRC)

Змінні фільтрації використовують домен атрибутів замість усіх значень кортежу (як це реалізовано в TRC)

In DRC, the filtering variable uses the domain of attributes instead of entire tuple values (as done in TRC)

$$\{ a1, a2, \ldots, an \mid P(a1, a2, \ldots, an) \}$$

Де **a1**, **a2**, …, **an** – атрибути, **P** – формула, заснована на атрибутах.

Where **a1**, **a2**, …, **an** are attributes and **P** stands for formulae built by inner attributes.

$$\{ \langle name, address \rangle \mid \in Supplier \wedge id = '3' \}$$

Яким буде результат?

What will be the output?