In The Name Of God



Database Implementation

Niloofar Mirzaei Chahardeh, Ph.D Candidate

Contents

- **Chapter 1: Basic principles and concepts**
- **Chapter 2: Relational Database**
- **Chapter 3: Structured Query Language**
- **Chapter 4: Database Design**
- **Chapter 5: Indexing Information**

Reference

- **An introduction to database systems C. J. Date 8th Edition 2003**
- **Database system concepts A. Silberschatz, H. F. Hurth, S. Sudarshan 6th Edition, 2010/7th 2019**
- **Fundamentals of database systems A. Elmasri, 7th Edition, 2016**
- **Database management systems R. Ramakrishnan, J. Gohrhe, 3rd Edition 2005**

Chapter 3:

Structured Query Language

Chapter 3

Introduction to:

- SQL Commands
- Keys
- Data Types
- Learning to Write Queries in SQL
- Relation Algebras Operations
- Special Operations
- Data Dependency
- Functional Dependency(FD)
- Full Functional Dependency(FFD)
- Closure of a set of FD
- Armstrong's Rules
- Derived Rules of Armstrong
- Closure of a set of Attributes

SQL Commands

1. **DDL**
2. **DML**
3. **DCL**

DDL

DDL = Data Definition Language

It refers to a set of instructions that affect the structure of the database and have nothing to do with the content of the database.

For Example = create table, create index, grant, revoke, alter table

DML

DML = Data Manipulation Language

It refers to a set of instructions that affect the content of the database and have nothing to do with the structure of the database.

For Example = select, insert into, delete from, update set

DCL

DCL = Data Control Language

It refers to a set of instructions that do not affect the content and structure of the database and are only related to controlling the execution of operations in the database.

For Example = rollback, commit, exit

پایگاه داده Bank

- ✓ این پایگاه داده مربوط به یک موسسه مالی و اعتباری فرضی است، که دارای شعب متعدد در شهرهای مختلف است.
- ✓ نام شعب و نام مشتریان یکتا است.
- ✓ یک مشتری می‌تواند بدون محدودیت حساب‌های بانکی متعدد در شعب مختلف افتتاح نماید.
- ✓ هر مشتری می‌تواند بدون محدودیت وام‌های مختلفی را از شعب مختلف این موسسه دریافت نماید.
- ✓ مشتریان می‌توانند در یک شعبه حساب داشته باشند و از شعبه دیگر وام دریافت نمایند.
- ✓ یک مشتری می‌تواند بدون آنکه حساب بانکی داشته باشد از شعب مختلف وام دریافت نماید.

Customer (c-name نام مشتری, c-city شهر مشتری, c-street خیابان مشتری)

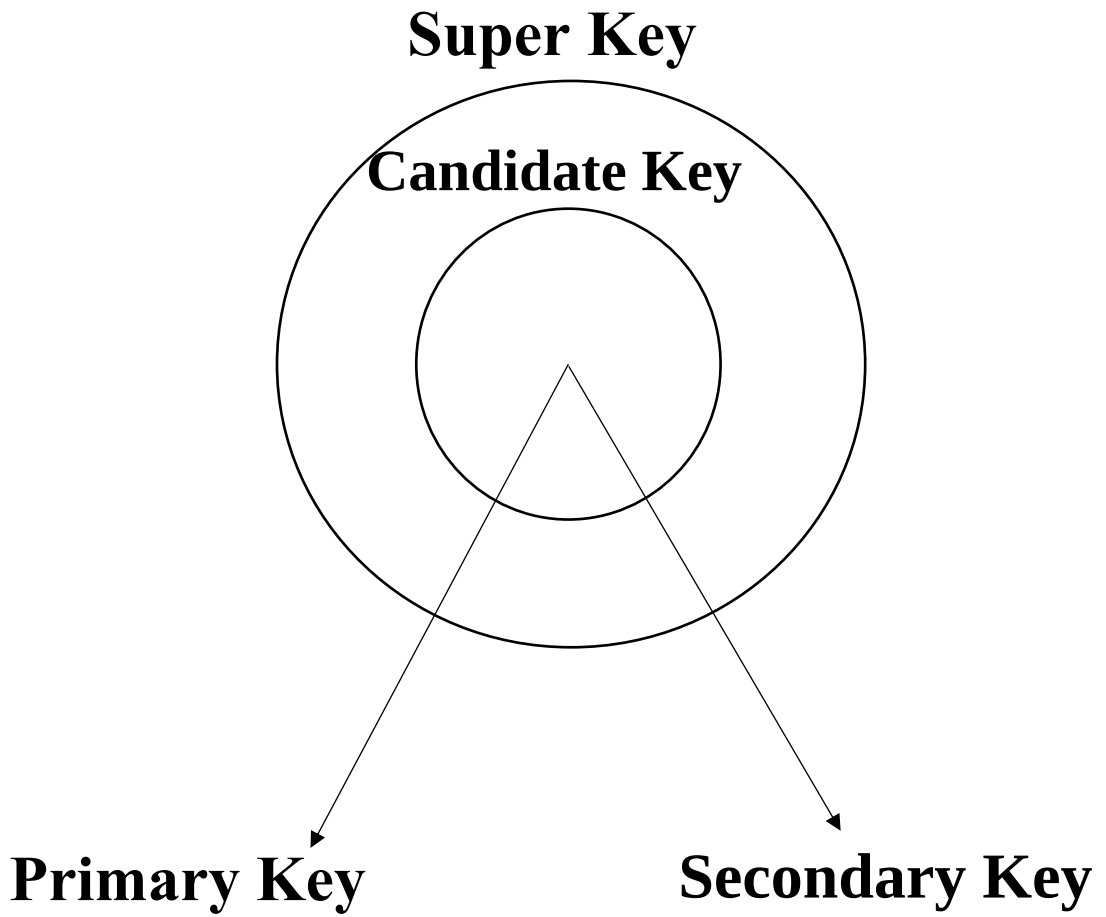
Branch (b-name نام شعبه, b-city شهر شعبه)

Deposit (c-name نام صاحب‌حساب, b-name نام شعبه‌حساب, a-no شماره‌حساب, balance موجودی حساب)

Borrow (c-name نام وام گیرنده, b-name نام شعبه وام‌دهنده, l-no شماره وام, amount مبلغ وام)

Keys

- Super Key
- Candidate Key
- Primary Key
- Secondary Key
- Foreign Key



Super Key

Any combination of features that does not have a repeating value, gives us the power to distinguish and differentiate.

Features:

- ☑ **They do not have duplicate values.**
- ☑ **Their function is to create separation and distinction between table rows.**

Candidate Key

In the terminology, a minimal key superkey is called a key superkey. It is also called a minimal superkey.

Features:

- ☑ **They do not have duplicate values.**
- ☑ **Their function is to create separation and distinction between table rows.**

Primary Key

It is one of the candidate keys.

It is suggested that we consider the candidate key as the primary key that contains the fewest attributes.

Features:

- ☑ **They do not have duplicate values.**
- ☑ **Their function is to create separation and distinction between table rows.**

Secondary Key

It is one of the candidate keys.

Features:

- ☑ **They do not have duplicate values.**
- ☑ **Their function is to create separation and distinction between table rows.**

Foreign Key

A foreign key refers to a candidate key in a reference table.

Features:

- ☑ **Establishing relationships between tables**
- ☒ **They do not have duplicate values.**

Data Types in SQL

- **Int(integer)**
- **Smallint**
- **Char(n)**
- **Varchar(n)**
- **Date**
- **Time**
- **Logical**
- **Blob**

Learning to Write Queries in SQL

1) **Create Database** → **Create Database Bank**

2) **Create Table** → **Create Table Branch**

```
Create Table Branch(b_name char(10), b_city char(20),
primary key(b_name))
```

Learning to Write Queries in SQL

```
2)  Create Table Borrow (c_name     char(10),          not null
                         b_name     char(10),          not null
                         l_no       char(8),
                         amount     Integer,
                        primary key(l_no))
Foreign key(c_name) references Customer
Foreign key(b_name) references Branch
```

Learning to Write Queries in SQL

Error:

1. Referential Integrity Error

☑We don't have a customer table at all.

☑Not referenced by any other table.

☑The type of the primary table and the secondary table must be the same.

☑If either the primary key type or the foreign key type changes, we get this error.

It doesn't check the name, but checks the typing and then gives an error.

Learning to Write Queries in SQL

Error:

2. Semantic Error

☑ Foreign key(c_name) references customer

☑ Foreign key(b_name) references Branch

☒ Foreign key(c_name) references Branch

☒ Foreign key(b_name) references Customer

Learning to Write Queries in SQL

Integrity rules:

1- The primary key does not accept duplicate values.

2- The primary key cannot accept null values.

3- The foreign key accepts null values.

Learning to Write Queries in SQL

3) Drop Table Customer

- **DDL Command**
- **When we want to delete a table**
  - ☒ **We cannot delete a table in which we have entered data**
  - ☒ **Not referenced by any other table.**

Learning to Write Queries in SQL

4) Alter Table + Table name

- **DDL Command**
- **This command neither creates nor deletes a table, but rather changes the structure of that table.**

Learning to Write Queries in SQL

4) Alter Table + Table name

Alter Table Branch ADD branch_number char(12)

- ☒ **Null values are entered into the database for as many rows as possible.**

Alter Table Branch Modify branch_number char(12)

- ☒ **There may be an obstacle that prevents the command from being executed.**

Learning to Write Queries in SQL

4) Alter Table + Table name

Alter Table Customer Drop ~~c-city~~ c_name

- ☒ **You will have a table that does not have a primary key.**
- ☒ **Other tables reference this table through this column, which serves as a foreign key for them. If it does not exist, it will give them an error.**

Relational Algebras Operations

Every relational algebra operation consists of two components:

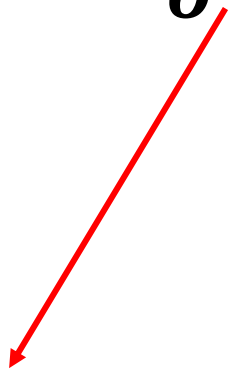
1. **Fundamental Operations**
2. **Additional Operations**

Fundamental Operations

- **Select (σ)**
- **Project (π)**
- **Cartesian Product ($\times$)**
- **Union ($\cup$)**
- **Set Deference ($-$)**
- **Rename (ρ)**

Select

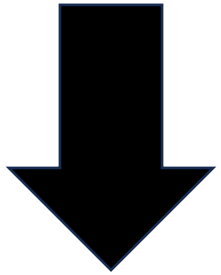
σ (**Relation**) **Output**



Set of conditions

Select

$$R_1 = \{(\alpha, 1, a), (\beta, 2, b), (\alpha, 2, b), (\alpha, 2, a)\}$$



$$R_1 = \{(A, B, C) \mid A \in D_1 \wedge B \in D_2 \wedge C \in D_3\}$$

Select

$(R_2) = \{(\beta, 2, b), (\alpha, 2, b), (\alpha, 2, a)\}$

$\sigma \quad (R_2) = \{(\beta, 2, b), (\alpha, 2, b), (\alpha, 2, a)\}$

$B > 1$

$\sigma \quad (R_2) = \{(\beta, 2, b)\}$

$A = \text{‘}\beta\text{’} \wedge B > 1$

$\sigma \quad (R_2) = \{\}$

$B > 2$

Select

$(R_3) = \{(\alpha, 1, a), (\beta, 1, a), (\alpha, 2, a), (\beta, 2, b)\}$

σ $(R_3) = \{(\alpha, 2, a), (\beta, 2, b)\}$

$B > 1$

$(\sigma$ $(R_2)) = \{(\alpha, 2, a)\}$

σ $B > 1$

$A = ‘\alpha’$

Select

$(R_3) = \{(\alpha, 1, a), (\beta, 1, a), (\alpha, 2, a), (\beta, 2, b)\}$

σ $(R_3) =$ ×

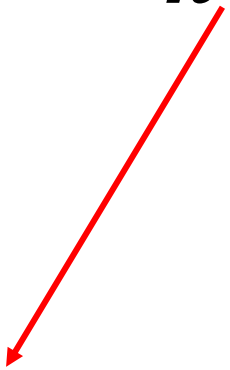
D > 7

σ $(R_3) = \{\}$

A= ‘α’ ∧ B > 3

Project

π (Relation) Output



One or more component names

Project

$(R_3) = \{(\alpha, 1, a), (\beta, 1, a), (\alpha, 2, a), (\beta, 2, b)\}$

π $(R_3) = \{(\alpha, 1), (\beta, 1), (\alpha, 2), (\beta, 2)\}$

A, B

π $(R_3) = \{(\alpha, a), (\beta, a), \sout{(\alpha, a)}, (\beta, b)\}$

A, C

$(\pi$ $(R_3)) = \{\alpha, \beta\}$

π A, B

A

Project

$(R_3) = \{(\alpha, 1, a), (\beta, 1, a), (\alpha, 2, a), (\beta, 2, b)\}$

(π (R3)) = × {(α, 1), (β, 1), (α, 2), (β, 2)}

π A, B

C

(π (R))

π L1

L2

$L_2 \supseteq L_1$

Project

$(R_3) = \{(\alpha, 1, a), (\beta, 1, a), (\alpha, 2, a), (\beta, 2, b)\}$

$\pi_{A,B}(\sigma_{B>1}(R_3)) = \{(\alpha, 2), (\beta, 2)\}$

$\pi_{A,B}(\sigma_{B>1}(R_3)) = \times$

D

$\sigma_{\theta}(\pi_{L}(R))$

Cartesian Product

Relation × Relation = Output

Cartesian Product

$(R_4) = \{(\alpha, 1), (\beta, 2)\}$

$(R_5) = \{(a, 1, \alpha), (b, 2, \beta), (b, 1, \alpha)\}$

$R_4 \times R_5 = \{(\alpha, 1, a, 1, \alpha), (\alpha, 1, b, 2, \beta), (\alpha, 1, b, 1, \alpha), \ldots\}$

Cartesian Product

$(R_4) = \{(\alpha, 1), (\beta, 2)\}$

$(R_5) = \{(a, 1, \alpha), (b, 2, \beta), (b, 1, \alpha)\}$

1) R => n S => m

$$R \times S = n \times m$$

$$R_4 \times R_5 = 2 \times 3 = 6$$

2) R => a S => b

$$R \times S = a + b$$

$$R_4 \times R_5 = 2 + 3 = 5$$

Union

Relation ∪ Relation = Output

Union

$(R_4) = \{(1, a), (2, b), (1, b)\}$

$(R_5) = \{(2, a), (1, b)\}$

$R_4 \cup R_5 = \{(1, a), (2, b), (1, b), (2, a)\}$

Union

Compatibility Conditions:

1. The tuples of the two relations must have the same degree.

2. The corresponding attributes in the tuples of the two relations must belong to the same domain.

Union

$(R_4) = \{(1, a), (2, b), (1, b)\}$

$(R_6) = \{(1, a, \alpha), \{(1, b, \alpha), (1, b, \beta)\}$

$D_1 = \{\alpha , \beta\}$

$D_2 = \{1, 2, 3\}$

$D_3 = \{a, b\}$

Second Degree

Third Degree

$$R_4 \cup R_6 = \{(1, a), (2, b), (1, b), (1, a, \alpha), (1, b, \alpha), \ldots\}$$

Not a relation

The first compatibility condition ✗

Union

$(R_4) = \{(1, a), (2, b), (1, b)\}$

$(R_7) = \{(\alpha, a), (\beta, b)\}$

$D_1 = \{\alpha, \beta\}$

$D_2 = \{1, 2, 3\}$

$D_3 = \{a, b\}$

Not of the same domain

$R_4 \cup R_7 = \{(1, a), (2, b), (1, b), (\alpha, a), (\beta, b)\}$

Not a relation

The second compatibility condition ✗

Set Deference

Relation−Relation = Output

R−S

Set Difference

$(R_4) = \{(1, a), (2, b), (1, b)\}$ $(R_5) = \{(2, a), (1, b)\}$

$(R_7) = \{(\alpha, a), (\beta, b)\}$

$R_5 - R_4 = \{(2, a)\}$

$R_4 - R_5 = \{(1, a), (2, b)\}$

$R_4 - R_7 = \{(1, a), (2, b), (1, b)\}$ ✗

The second compatibility condition

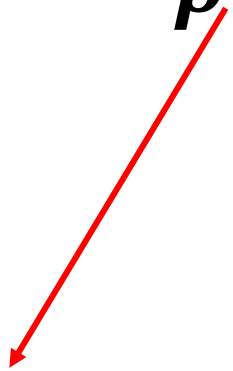
$D_1 = \{\alpha, \beta\}$

$D_2 = \{1, 2, 3\}$

$D_3 = \{a, b\}$

Rename

ρ(Relation) Output



Alias

Rename

$$\rho_{A}(R5)$$

☑ **Each name is a way to access that relation.**

☑ **Sometimes, it is necessary to access a relation through more than one name.**

Additional Operations

- **Intersection (∩)**
- **Assignment (←)**

Assignment

This relational operator is implemented using the basic operator Rename.

Learning to Write Queries in SQL

5) **Insert Into** + Table name

Values + (Value1, Value2, . . .)

Insert Into = Assignment + Union

Insert Into = ← + ∪

Learning to Write Queries in SQL

For example:

Query 1: یک مشتری به نام علی، با آدرس شهر شیراز و خیابان حافظ را به پایگاه داده بانک اضافه نمایید.

- **RA:**

$$Customer \leftarrow Customer \cup \{(\text{‘Ali’}, \text{‘Shiraz’}, \text{‘Hafez’})\}$$

- **SQL:**

```
Insert Into Customer
Values (‘Ali’, ‘Shiraz’, ‘Hafez’)
```

Learning to Write Queries in SQL

Error:

- ❖ Values(‘Ali’, ‘Shiraz’, ‘Hafez’) Values(‘Ali’, ‘Shiraz’)
- ☒ Violation of compatibility conditions

- ❖ Values(‘Ali’, ‘Shiraz’, ‘Hafez’) Values(‘Ali’, ‘Shiraz’,)
- ☒ Null

- ❖ Values(‘Ali’, ‘Shiraz’, ‘Hafez’) Values(,‘Shiraz’, ‘Hafez’)
- ☒ Violation of the second rule of Integrity

Learning to Write Queries in SQL

Error:

- Values(‘Ali’, ‘Shiraz’, ‘Hafez’) Values(‘Ali’, ‘Shiraz’, ‘Hafez’)
- ☒ Duplicate Error

- Values(‘Ali’, ‘Shiraz’, ‘Hafez’) Values(‘Ali’, ‘Shiraz’, 1890)
- ☒ Violation of compatibility conditions

- Insert Into Deposit

  Values(‘Ali’, ‘Kaj’, ‘385’, 2800)
- ☒ Referential Integrity Error

Learning to Write Queries in SQL

Error:

- **Insert Into Deposit**

  Values(‘Mirzaei’, ‘Mirzaei’, ‘385’, 2800)

- **Violation of compatibility conditions**

Learning to Write Queries in SQL

6) **Delete From** + Table name

Where + Column name

Delete From = Assignment + Set Deference + Select

Delete From = $\leftarrow$ + $-$ + σ

Learning to Write Queries in SQL

For example:

Query 2: اطلاعات یک حساب بانکی با شماره حساب ۲۰۰ را حذف نمایید.

- **RA:**

$$\text{Deposit} \leftarrow \text{Deposit} - \sigma_{\text{A-no = '200'}}(\text{Deposit})$$

- **SQL:**

```
Delete From Deposit
Where A-no = '200'
```

Learning to Write Queries in SQL

Error:

- ❖ **Delete From Deposit** ~~**Where A-no = '200'**~~
- ☒ **All rows will be deleted.**

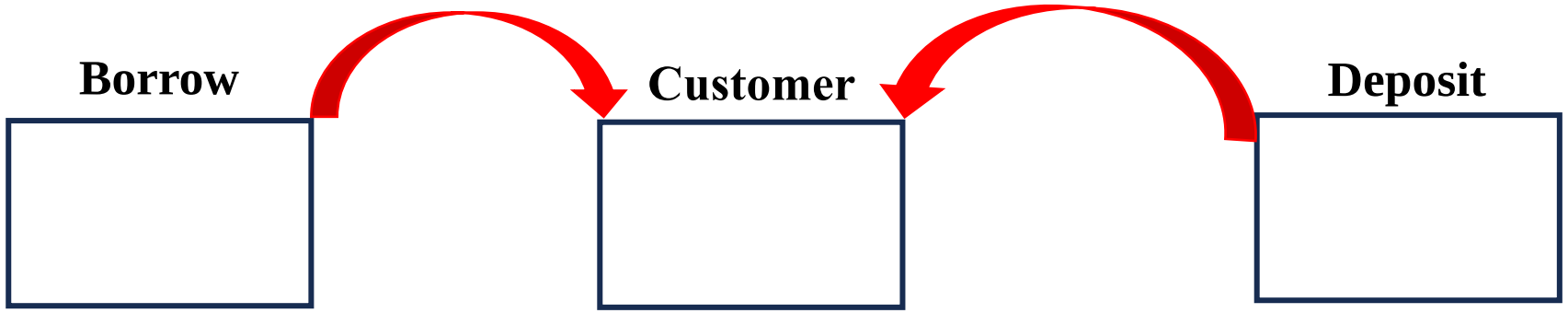
- ❖ **Delete From Deposit** **Where A-no = '~~200~~'**
- ☒ **Noting**

- ❖ **Delete From Deposit** **Where A-no= '200'**
- ☒ **Notting**

Learning to Write Queries in SQL

Error:

- ❖ **Delete From Customer** **Where C-name = ‘Ali’**
- ☒ **Referential Integrity Error**



Learning to Write Queries in SQL

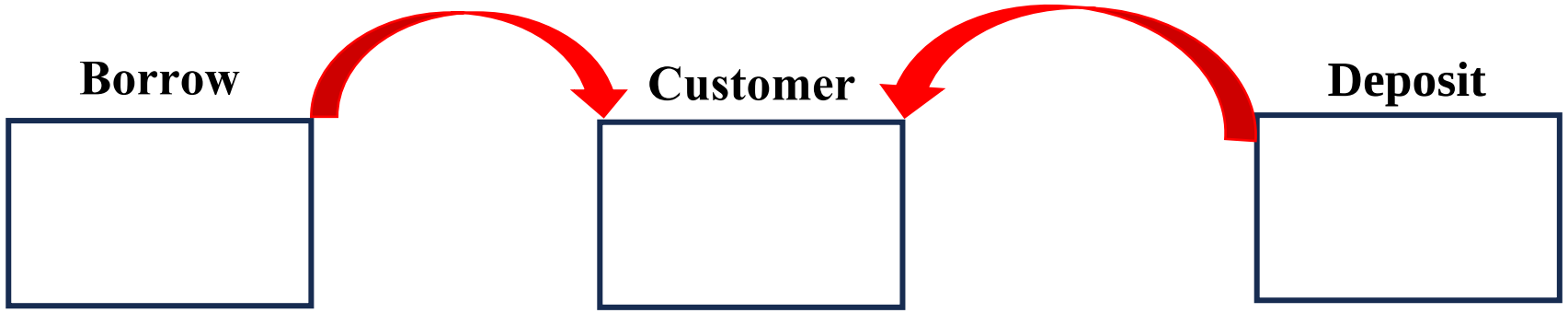
7) **Foreign Key + On Delete Cascade**

Foreign Key(c_name) references Customer **On Delete Cascade**

On Delete Cascade ≠ On Delete Restricted

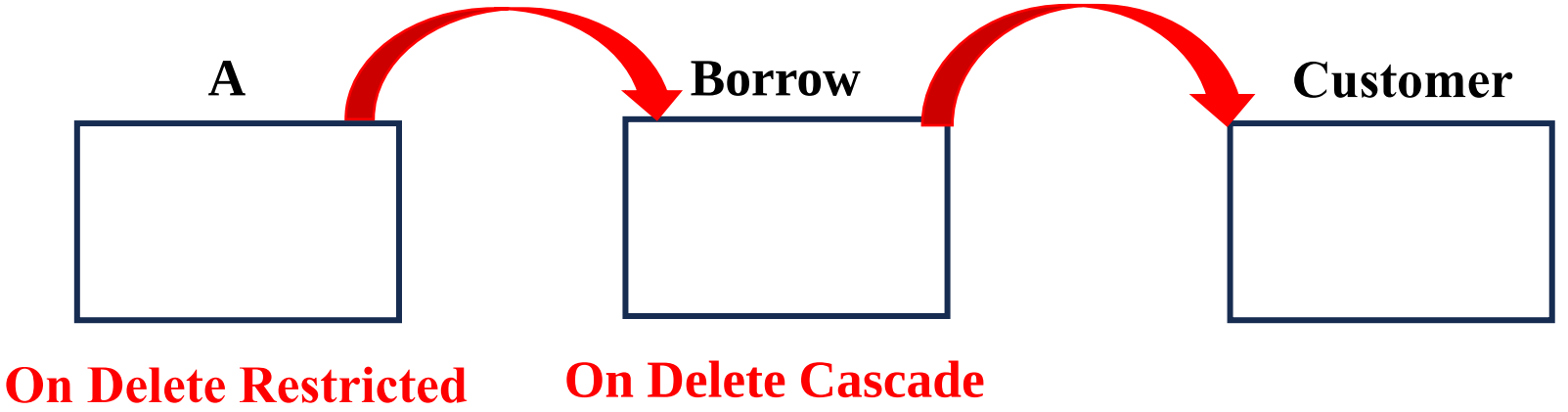
Learning to Write Queries in SQL

⌦ **We are not allowed to delete from the Customer table.**



Learning to Write Queries in SQL

⌦ **We are not allowed to delete from the Customer table.**



Learning to Write Queries in SQL

On Delete Cascade

☑ **Another table must not reference the Customer table.**

☑ **All referencing tables must on Delete Cascade so that data.**

Learning to Write Queries in SQL

8) **Update** + Table name

Set + Column name

Where + Column name

Learning to Write Queries in SQL

For example:

Query 3: اطلاعات شهر مشتری به نام علی را به تهران تغییر دهید.

- **SQL:**

```
Update Customer
Set C-city = 'Tehran'
Where C-name = 'Ali'
```

Learning to Write Queries in SQL

Error:

- **Update Customer** ~~**Where C-name = 'Ali'**~~

  Set C-city = 'Tehran'

- ☒ **Tehran becomes the city of all customers.**

- **Update Customer**

  Set C-city = 'Tehran'

  Where C-name = ~~'Ali'~~

- ☒ **Noting**

Learning to Write Queries in SQL

Error:

- **Update Customer**

  Set C-city = ‘Tehran’

  Where C-name = ‘Ali’

- **Noting**

Learning to Write Queries in SQL

Error:

- ❖ **Update Customer**

  Set C-name = ‘Ali’

  Where C-name = ‘Reza’

- ☒ **Violation of referential integrity**
- ☒ **Duplicate Error**

Learning to Write Queries in SQL

Error:

- **Update Deposit**

  Set C-name = ‘Ali’

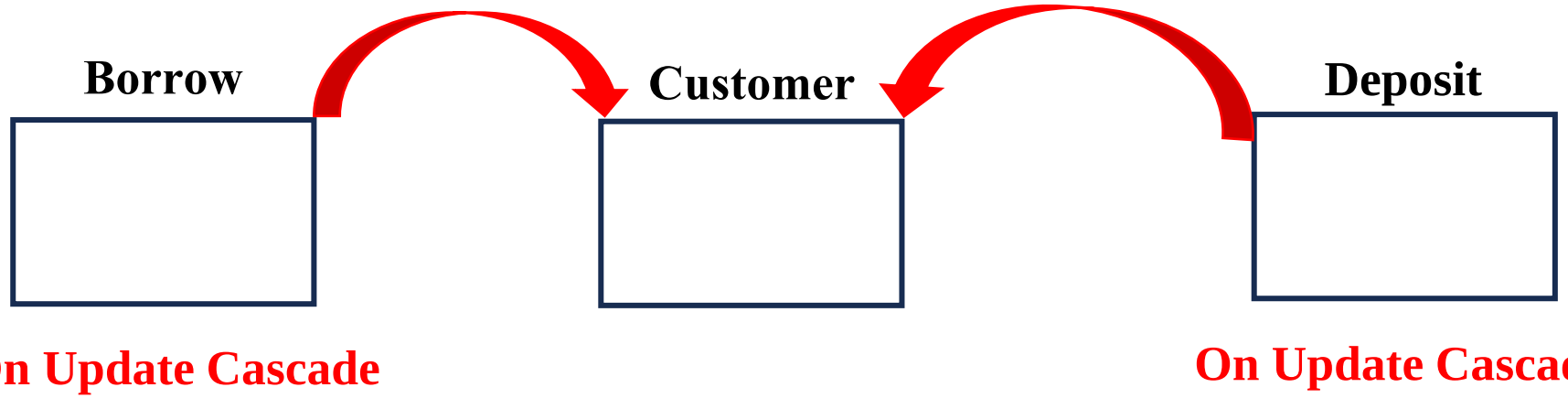
  Where A-no = ‘187’

- **Referential integrity**

Learning to Write Queries in SQL

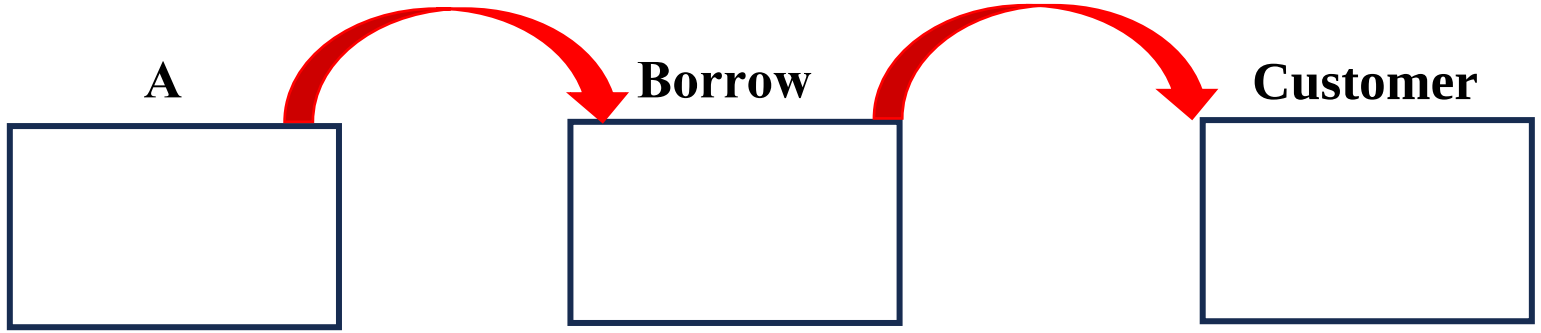
9) Foreign Key + On Update Cascade

Foreign Key(c_name) references Customer **On Update Cascade**



Learning to Write Queries in SQL

⌦ **We are not allowed to update from the Customer table.**



On Update Cascade

Learning to Write Queries in SQL

10) **Select** + Column name
From + Table name
} Mandatory part

Where + Condition
} Optional part

•

•

•

Learning to Write Queries in SQL

For example:

Query4: Query بنویسید که مشخصات کلیه شعب را به ما بدهد.

- **RA:**

  Branch

- **SQL:**

  Select *

  From branch

Learning to Write Queries in SQL

For example:

Query 5: Query بنویسید که نام کلیه شعبه هایی که وام پرداخت کرده اند، را به ما بدهد.

- **RA:**

$$\pi_{\text{b-name}}(Borrow)$$

- **SQL:**

```
Select b-name
From Borrow
```

Learning to Write Queries in SQL

For example:

Query 6: **Query بنویسید که نام مشتریان تهرانی را به ما بدهد.**

- **RA:**

$$\pi_{\text{c-name}}\left(\sigma_{\text{c-city = 'Tehran'}}(Customer)\right)$$

- **SQL:**

```
Select c-name
From Customer
Where c-city = 'Tehran'
```

Learning to Write Queries in SQL

For example:

Query 7: Query بنویسید که نام شعبی که به مشتری به نام علی به مبلغی بیشتر از ۵.۰۰۰ تومان وام داده اند، را به ما بدهد.

- **RA:**

$$\pi_{\text{b-name}}\left(\sigma_{\text{c-name = 'Ali' } \wedge \text{ amount > 5.000}}(Borrow)\right)$$

- **SQL:**

```
Select b-name
From Borrow
Where c-name = 'Ali' and amount >5.000
```

Learning to Write Queries in SQL

For example:

Query 8: Query بنویسید که نام شعبی که تاکنون وام نداده اند، را به ما بدهد.

- **RA:**

$$\pi_{\text{b-name}}(\text{Branch}) - \pi_{\text{b-name}}(\text{Borrow})$$

نام کلیه شعب — نام شعبی که وام داده اند.

- **SQL:**

(Select b-name From Branch)

Except

(Select b-name From Borrow)

نام کلیه شعب

نام شعبی که وام داده اند.

Learning to Write Queries in SQL

11) Select + Distinct + Column name

From + Table name

For example:

Select distinct b-name

From Borrow

For example:

Query 9: بنویسید که نام شعبی که با مشتری به نام علی سروکار ندارد، را به ما بدهد.

- **RA**

$$\pi_{\text{b-name}}(\text{Branch}) - (\pi_{\text{b-name}}(\sigma_{\text{c-name = 'Ali'}}(Deposit)) \cup \pi_{\text{b-name}}(\sigma_{\text{c-name = 'Ali'}}(Borrow)))$$

نام کلیه شعب را می دهد.

نام شعبی که علی در آن حساب دارد.

نام شعبی که به علی وام داده است.

- **SQL:**

```
((Select b-name  From Branch)
Except
((Select b-name  From Deposit where c-name ='Ali')
 Union
(Select b-name from Borrow where c-name = 'Ali')))
```

For example:

Query 10: Query بنویسید که نام و نام شهر مشتریانی که از شعبه کاج وام گرفته اند، را به ما بدهد.

- **RA**

$$\pi_{\text{Borrow. c-name, Customer. c-city}}\left(\sigma_{\text{Borrow. c-name = Customer. c-name} \wedge \textit{Borrow}.\text{ b-name = 'kaj'}}(\textit{Borrow} \times \textit{Customer})\right)$$

شرط ضرب

- **SQL:**

```
Select Borrow. c-name, Customer. c-city
From Borrow, Customer
Where Borrow. c-name = Customer. C-name and Borrow. b-name = 'Kaj'
```

شرط ضرب

Relational Algebra Queries

Borrow

c- name	b-name	L-no	amount
Ali	Kaj	46	1500
Reza	Bahar	61	7500
Hamed	Kaj	53	3000

Customer

c-name	c- city	c- street
Ali	Tehran	Azadi
Reza	Tabriz	Aseman
Hamed	Hamedan	Sina
Ahmad	Mashhad	Noor

Relational Algebra Queries

Customer × Borrow

c-name	c- city	c- street	c- name	b-name	L-no	amount
Ali	Tehran	Azadi	Ali	Kaj	46	1500
Ali	Tehran	Azadi	Reza	Bahar	61	7500
Ali	Tehran	Azadi	Hamed	Kaj	53	3000
Reza	Tabriz	Aseman	Ali	Kaj	46	1500
Reza	Tabriz	Aseman	Reza	Bahar	61	7500
Reza	Tabriz	Aseman	Hamed	Kaj	53	3000
Hamed	Hamedan	Sina	Ali	Kaj	46	1500
Hamed	Hamedan	Sina	Reza	Bahar	61	7500
Hamed	Hamedan	Sina	Hamed	Kaj	53	3000
Ahmad	Mashhad	Noor	Ali	Kaj	46	1500
Ahmad	Mashhad	Noor	Reza	Bahar	61	7500
Ahmad	Mashhad	Noor	Hamed	Kaj	53	3000

Learning to Write Queries in SQL

12) Division

Division = Except + Not Exists

Division

For example:

Query 11: Query بنویسید که نام شعبی که به همه مشتریان آبادانی وام داده اند، را به ما بدهد.

- **RA**

$$\pi_{\text{b-name, c_name}}(\text{Borrow}) \div (\pi_{\text{c-name}}(\sigma_{\text{c-city = 'Abadan''}}(\text{Customer})))$$

نام شعبی که وام داده اند.

نام همه مشتریان آبادانی را می دهد.

{} = نام مشتریانی که شعبه x به آنها وام داده است – نام همه مشتریان آبادانی

N. Mirzaei Chahardeh, Ph.D Candidate Data Base

Division

- **SQL**

```
Select B. b_name
from Branch  B
where Not Exists ((Select c_name
                   from Customer
                   where c_city = 'Abadan')
                   Except
                   (Select D. c_name
                    from Borrow D
                    where B. b_name = D. b_name))
```

نام شعبی که حاصل این تفاضل برای آن تهی است.

نام همه مشتریان آبادانی را به ما می دهد.

نام مشتریانی که از آن شعبه وام گرفته اند.

Special Operations

- ☑ **Not In**
- ☑ **In**
- ☑ **Not Exists**
- ☑ **Exists**

- ☒ **Between**
- ☒ **Not Like**
- ☒ **Like**
- ☒ **Is Null**
- ☒ **Is Not Null (Not Null)**

Between

For example:

Query 12: Query بنویسید که نام مشتریانی که مقدار وام دریافتی آنها بین ۵۰۰ تا ۵.۰۰۰ تومان بوده است، را به ما بدهد.

- **Type 1:**

```
Select c-name
From Borrow
Where amount > 500
And     amount < 5.000
```

- **Type 2:**

```
Select c-name
From Borrow
Where amount between 500 and 5.000
```

Like and Not Like

For example:

Query 13: Query بنویسید که نام شعبی که شهر آن با حرف A شروع می شود، را به ما بدهد.

- **Like:**

```
Select b-name
From Branch
Where b-city is like ‘A%’
```

Like and Not Like

For example:

Query 14: Query بنویسید که نام شعبی که شهر آن با حرف A تمام می شود، را به ما بدهد.

- **Like:**

```
Select b-name
From Branch
Where b-city is like ‘%A’
```

Like and Not Like

For example:

Query 15: Query بنویسید که نام شعبی که در اسم شهر آن حرف A می باشد، را به ما بدهد.

- **Like:**

```
Select b-name
From Branch
Where b-city is like '%A%'
```

Like and Not Like

For example:

Query 16: Query بنویسید که نام شعبی که شهر آن با حرف A شروع نمی شود، را به ما بدهد.

- **Not Like:**

```
Select b-name
From Branch
Where b-city is not like ‘A%’
```

Like and Not Like

For example:

Query 17: Query بنویسید که نام شعبی که در اسم شهر آن حرف A نیست، را به ما بدهد.

- **Not Like:**

```
Select b-name
From Branch
Where b-city is not like '%A%'
```

Is Null and Is Not Null

For example:

Query 18: Query بنویسید که نام شعبی که شهر آن ها معلوم نیست، را به ما بدهد.

- **Is Null:**

```
Select b-name
From Branch
Where b-city is Null
```

Is Null and Is Not Null

For example:

Query 19: Query بنویسید که نام شعبی که برای آن شهری ثبت شده است، را به ما بدهد.

- **Is Not Null:**

```
Select b-name
From Branch
Where b-city is not Null
```

- **Is Not Null:**

```
Select b-name
From Branch
Where b-city = b-city
```

In and Not In

For example:

Query 20: Query بنویسید که نام مشتریانی که از شعب شهر آبادان وام گرفته اند و در شعبه کاج حساب دارند، را به ما بدهد.

Select c-name from Borrow where b-name in ← نام مشتریانی که وام گرفته اند.

(Select b-name from Branch where b-city = 'Abadan') ← نام شعب شهر آبادان را میدهد.

And c-name in

(Select c-name from Deposit where b-name = 'Kaj') ← در شعبه کاج حساب دارند.

Nested In

For example:

Query 21: Query بنویسید که نام مشتریان زاهدانی که در شعب شهر شیراز حساب دارند، را به ما بدهد.

Select c-name from Customer where c-city = ‘Zahedan’

نام مشتریان زاهدانی که اسم آنها در لیست است.

And c-name in

(Select c-name from Deposit where b-name in

نام مشتریانی که در شعب شهر شیراز حساب دارند، و اسم آنها در فهرست است.

(Select b-name from Branch where b-city = ‘Shiraz’))

نام شعبه های شهر شیراز را می دهد.

Nested In and not In

For example:

Query 22: Query بنویسید که نام مشتریان زاهدانی که در شعب شهر شیراز حساب ندارند، را به ما بدهد.

```
Select c-name from Customer where c-city = 'Zahedan'
And c-name not in
(Select c-name from Deposit where b-name in
(Select b-name from Branch where b-city = 'Shiraz'))
```

نام مشتریان زاهدانی که اسم آنها در لیست نیست.

نام مشتریانی که در شعب شهر شیراز حساب دارند.

نام شعبه های شهر شیراز را می دهد.

Nested In and not In

For example:

Query 23: Query بنویسید که نام مشتریان زاهدانی که در شعب غیر شهر شیراز حساب دارند، را به ما بدهد.

Select c-name from Customer where c-city = ‘Zahedan’ ← نام مشتریان زاهدانی که اسم آنها در لیست است.

And c-name in

(Select c-name from Deposit where b-name not in ← نام مشتریانی که در شعب غیر شهر شیراز حساب دارند، شاید در شعب شیراز هم حساب داشته باشند.

(Select b-name from Branch where b-city = ‘Shiraz’) ← نام شعبه های شهر شیراز را می دهد.

Nested Not In and Not In

For example:

Query 24: Query بنویسید که نام مشتریان زاهدانی که در شعب غیر شهر شیراز حساب ندارند، را به ما بدهد.

Select c-name from Customer where c-city = ‘Zahedan’ ← نام مشتریان زاهدانی که نام آنها در این لیست نیست.

And c-name not in

(Select c-name from Deposit where b-name not in ← نام مشتریانی که در شعب غیر شهر شیراز حساب دارند،

(Select b-name from Branch where b-city = ‘Shiraz’) ← نام شعبه های شهر شیراز را می دهد.

Exist, Not Exist, Unique

13) Select + Column name

From + Table name

Where + Exist (Select + Column name

Not Exist From + Table name

Unique .

.)

Exist, Not Exist, Unique

For example:

Query 25: Query بنویسید که نام مشتریانی را که فقط یک حساب دارند، را به ما بدهد.

```
Select C.c-name
from Customer  C
where unique (Select *
                    from Deposit D
                    where C.c-name = D.c-name)
```

Exist, Not Exist, Unique

For example:

Query 26: Query بنویسید که نام مشتریانی را که حداقل یک حساب دارند، را به ما بدهد.

```
Select C.c-name
from Customer  C
where Exist     (Select *
                 from Deposit D
                 where C.c-name = D.c-name)
```

Exist, Not Exist, Unique

For example:

Query 27: Query بنویسید که نام مشتریانی را که هیچ حساب بانکی ندارند، را به ما بدهد.

```
Select C.c-name
from Customer  C
where Not Exist (Select *
                 from Deposit D
                 where C.c-name = D.c-name)
```

Data Dependency

Data dependency classification:

1) **Functional Dependency(FD)**
2) **Multi-Value Dependency(MVD)**
3) **Join Dependency(JD)**

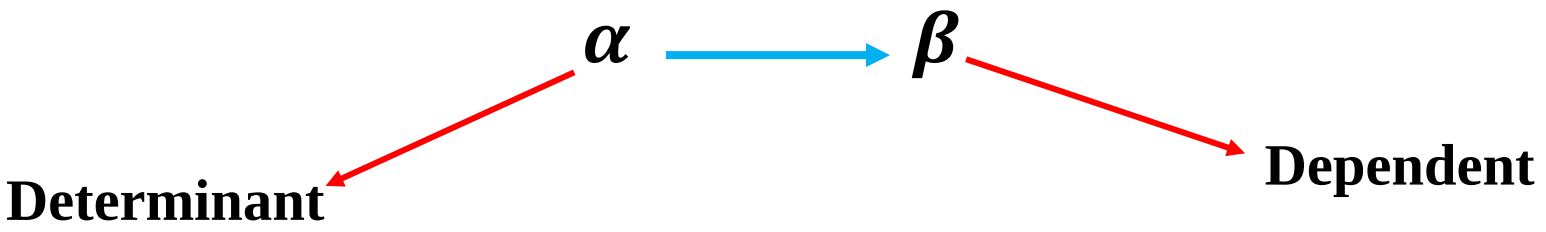
Functional Dependency(FD)

اگر α ← یک یا چند خصیصه، یک یا چند ستون

اگر β ← یک یا چند خصیصه، یک یا چند ستون

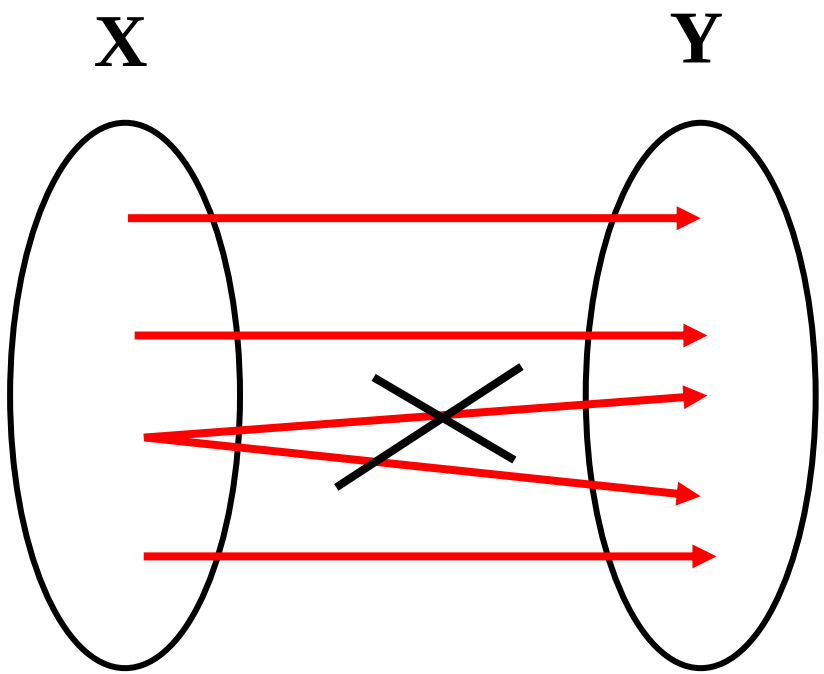
اگر γ ← یک یا چند خصیصه، یک یا چند ستون

Functional dependency symbol: →

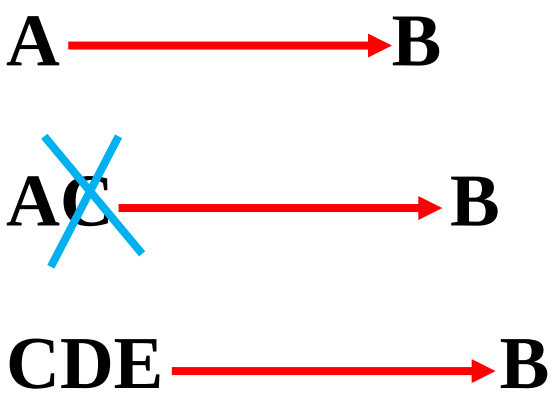


Functional Dependency(FD)

For Example: **St-no** → **St-name**



Full Functional Dependency(FFD)



Closure of a Set of FD

Closure of set of Functional Dependencies Symbol: F^+

$F^+ = F \cup$ {همه وابستگی های تابعی قابل استنتاج از F}

For Example:

R(A B C)

F = { A ⟶ B, B ⟶ C }

F^+ = {A ⟶ B, B ⟶ C, A ⟶ C}

این همان وابستگی قابل استنتاج از F بوده است.

Closure of a Set of FD

For Example:

R(A B C)

F = { A ⟶ B, B ⟶ C }

G = { A ⟶ B, B ⟶ C, A ⟶ C }

F^+ = {A ⟶ B, B ⟶ C, A ⟶ C}

G^+ = {A ⟶ B, B ⟶ C, A ⟶ C}

$F^+ = G^+$

Armstrong's Rules

1) Reflexivity Rule

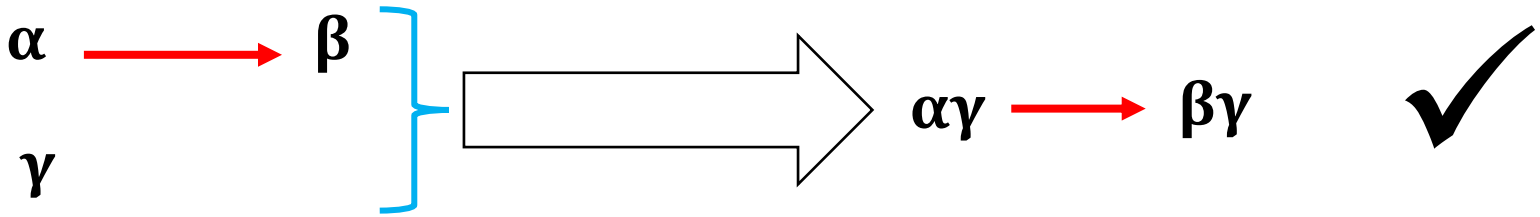
$\beta \subseteq \alpha$ $\qquad$ $\alpha \longrightarrow \beta$

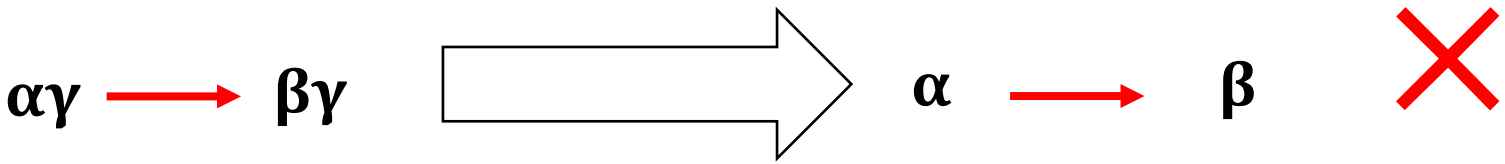
$A \longrightarrow A$

$AB \longrightarrow B$

Armstrong's Rules

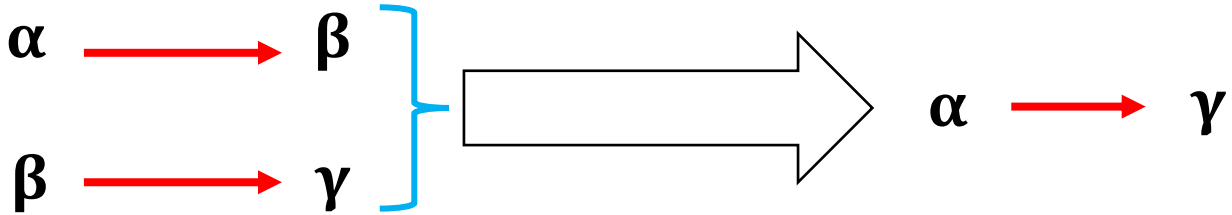
2) Augmentation Rule





Armstrong's Rules

3) Transitivity Rule



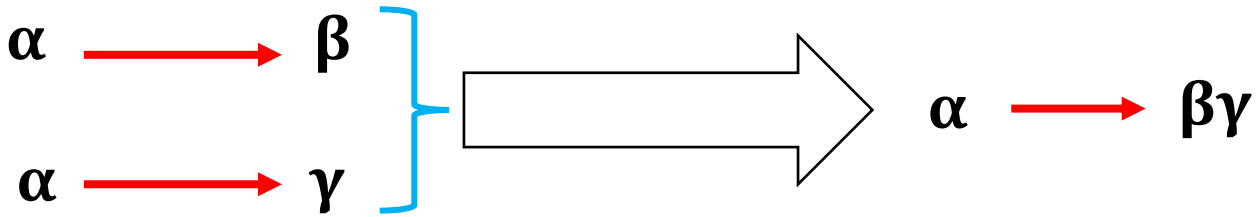
Derived Rules of Armstrong

1. Reflexivity Rule

$$\alpha \longrightarrow \alpha$$

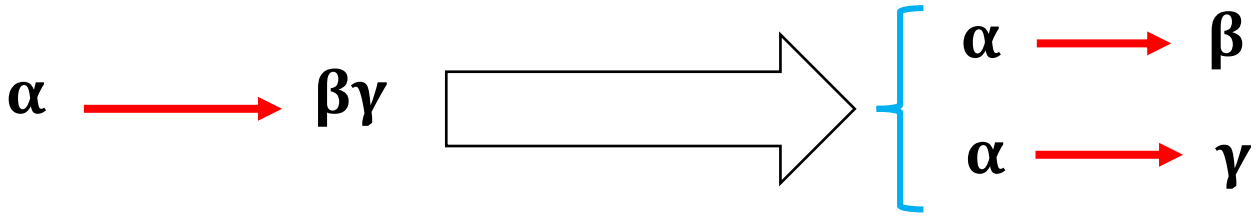
Derived Rules of Armstrong

2. Union Rule



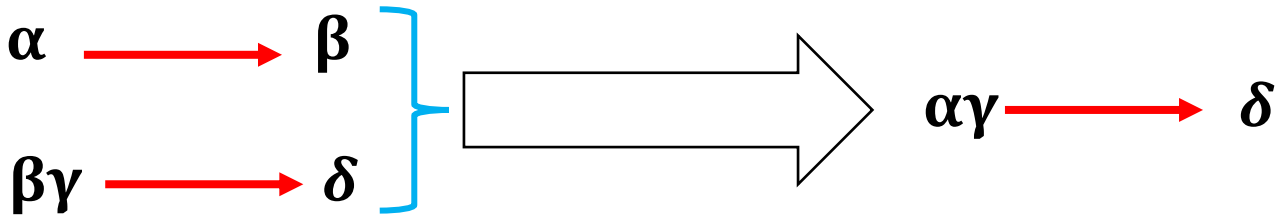
Derived Rules of Armstrong

3. Decomposition Rule



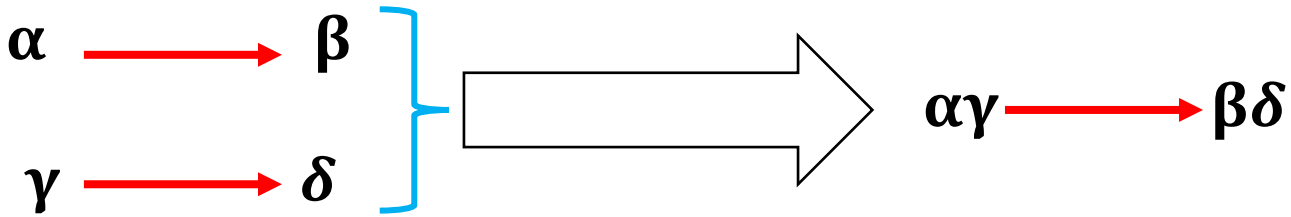
Derived Rules of Armstrong

4. Pseudo-Transitivity Rule



Derived Rules of Armstrong

5. Composition Rule



Closure of a Set of Attributes

For Example:

R(A B C D)

FD = { A ⟶ B, B ⟶ C}

$\{A\}^+$

$\{A\}^+ = A \cup$ {همه خصیه های قابل دسترسی از طریق A}

Closure of a Set of Attributes

For Example:

R (A B C D)

FD = { A ⟶ B, B ⟶ C}

$\{A\}^+ = \{A, B, C\}$

$\{B\}^+ = \{B, C\}$

$\{C\}^+ = \{C\}$

$\{B, C\}^+ = \{B, C\}$

چون A تمامی اعضا را به ما می دهد، پس کلید کاندید ما محسوب می شود.