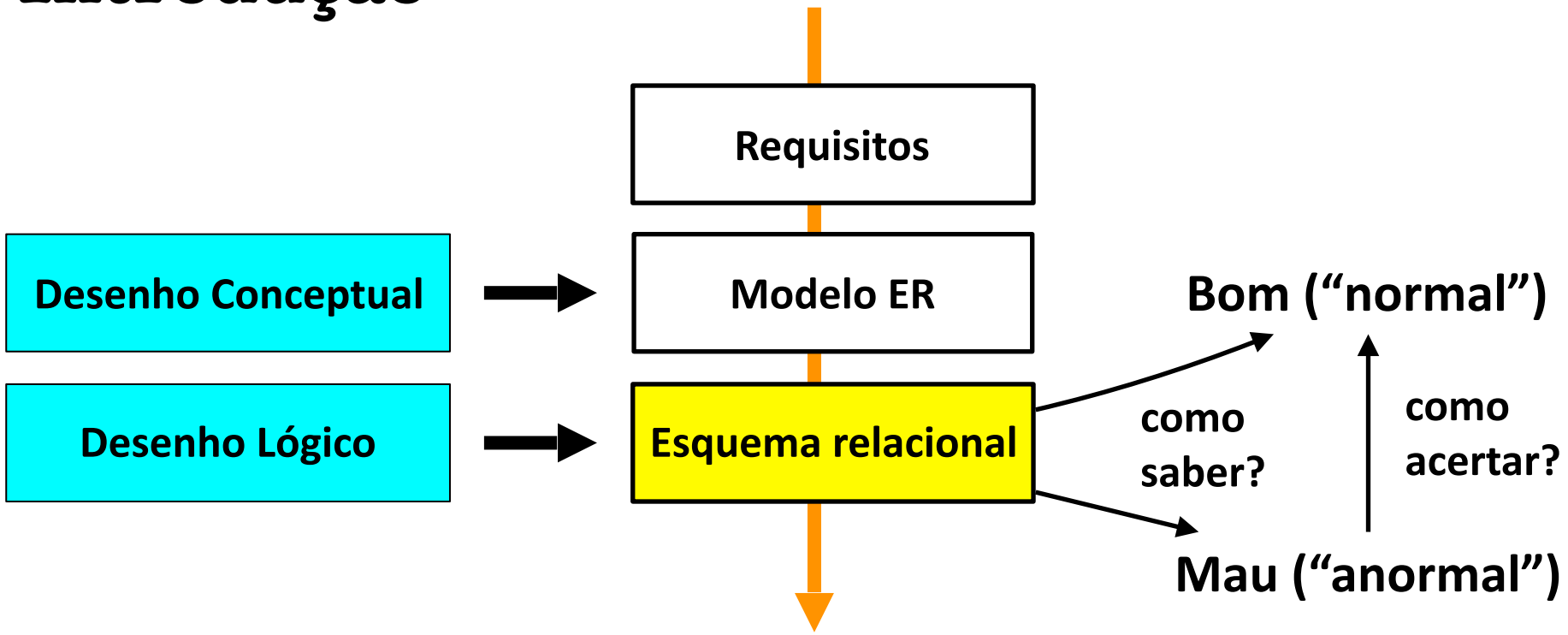
Normalização de Bases de Dados

Bases de Dados (CC2005)
Departamento de Ciência de Computadores
Faculdade de Ciências da Universidade do Porto

Eduardo R. B. Marques — DCC/FCUP

Introdução

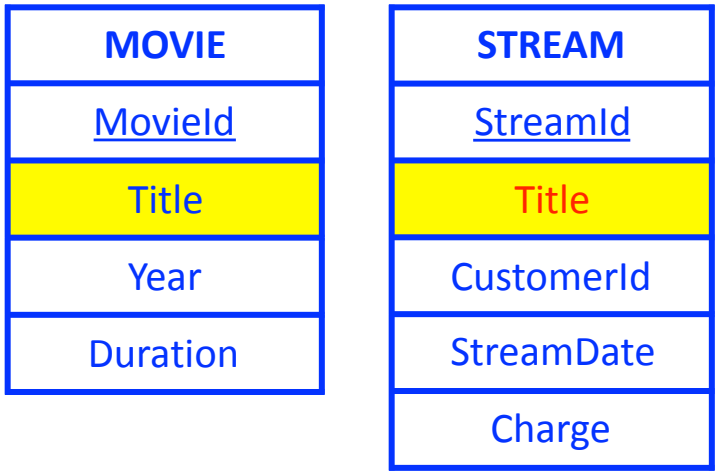


- Qual é o critério para aferir se **o desenho de uma BD**, expresso por um esquema relacional, é "bom" ou "mau"? E e for "mau", como acertá-lo?
- Um "mau" desenho tem **"anomalias" (é "anormal")** que deveremos conseguir detectar de forma objectiva, e estas devem poder ser eliminadas por um processo de **normalização** do esquema.

Princípios gerais para um bom desenho

- **Princípios gerais:**
  - **Uma tabela deve representar apenas um conceito do universo da BD,** em correspondência a uma entidade ou um relacionamento entre entidades, e manter a informação completa associada a este.
  - **Informação de uma tabela não deve ser duplicada em outra,** com excepção da chave primária via referências por chaves externas.
  - As tabelas devem relacionar-se **apenas** também via ligações **chave-externa/chave primária** e não por outros atributos.
  - Devemos evitar atributos que sejam **NULL** em um grande número de casos.
- A violação destes princípios leva geralmente a:
  - a um **significado semântico pouco claro** do esquema da BD;
  - **anomalias** em operações de dados e/ou **inconsistências** resultantes das mesmas
  - **desperdício e redundância** no armazenamento;

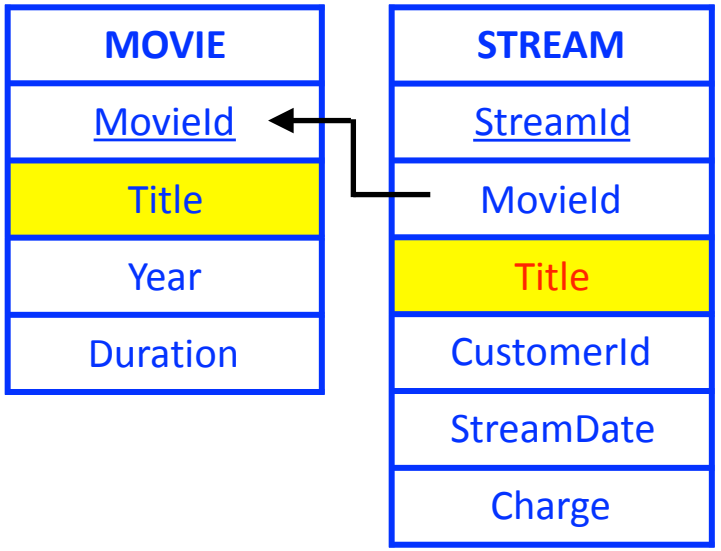
“Maus exemplos”



- Mau desenho:
  - Relação entre **STREAM** e **MOVIE** feita através de **TITLE** sem relação expressa no esquema.
  - Embora **TITLE** seja um atributo chave em **MOVIE**, a chave primária é **MovieId**.
  - **Possível ambiguidade semântica**: será que **STREAM** tem um título afinal e não se relaciona com **MOVIE** ?!

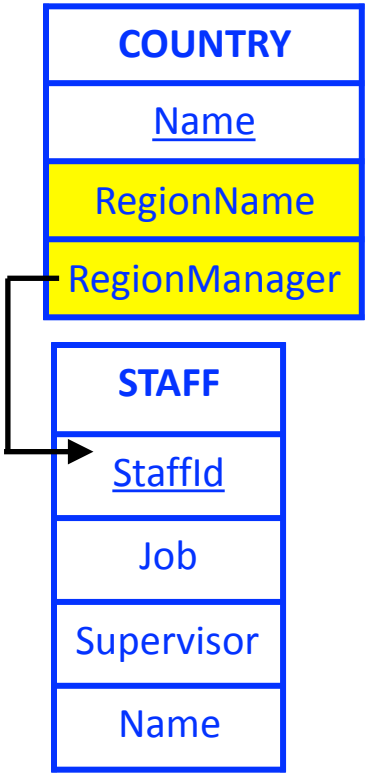
- **Anomalias na manipulação de dados**: podemos por exemplo alterar isoladamente **MOVIE** (actualizando o título, removendo entradas) ou **CUSTOMER** (usando um título de filme não existente) levando a inconsistência de dados e perda de informação.
- Duplicação do atributo **Title.**

“Maus exemplos” (cont.)



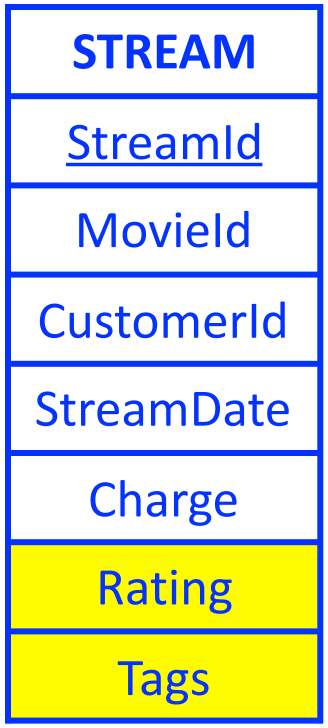
- Mau desenho:
  - Relação via chave externa agora OK, mas **TITLE** duplicado
  - Redundância de informação e anomalias do exemplo anterior ainda possíveis, excepto no caso da remoção de um filme.
  - Possível ambiguidade semântica mantém-se.

“Maus exemplos” (cont.)



- **Significado semântico do esquema não é de todo claro!** **COUNTRY** corresponde a “duas entidades”: **REGION** “existe escondida” na tabela **COUNTRY.** Além disso, **RegionManager** **depende** na prática de **RegionName** mas o **esquema não captura essa dependência.**
- Anomalias na manipulação de dados: podemos alterar **RegionName** sem qualquer restrição ou **RegionManager** de forma independente de **RegionName** levando a inconsistência dos dados.
- Mesmo que a consistência da BD de alguma forma sobreviva às “intempéries” o esquema é também pouco económico em termos de espaço: **RegionName e RegionManager** são repetidos para vários países (ao invés de um único atributo **RegionId** no esquema “bom”).

“Maus exemplos” (cont.)



- Vamos supor que:
  - um cliente poderia atribuir opcionalmente uma valorização a um filme reflectindo se gostou/não gostou de um filme ex. numa escala de 0 a 5 e também várias “tags” a um filme por exemplo em texto separado por vírgulas como em **'classic, Hitchcock, thriller'.**
  - que estes dados são gravados na tabela **STREAM**
  - que grande parte dos clientes não atribuem “ratings” ou “tags” deixando esses atributos a **NULL**.
- Problemas:
  - Demasiados registos com entradas **NULL** levam a desperdício de espaço.
  - Dados de “ratings” e “tags” parecem depender do “stream” em vez do cliente o que seria mais natural: faz pouco sentido que o mesmo cliente possa dar vários “ratings” diferentes para o mesmo filme.
  - “Ratings” e “tags” podem ser vistas como entidades que merecem existência concreta na BD. Tags é também implicitamente um atributo multi-valor. Como definiríamos as correspondentes tabelas ?

Normalização / formas normais

- Processo que visa corrigir deficiências de uma esquema de BD por forma a transformá-lo numa **forma normal**.
- Uma forma normal tem associados:
  - um conjunto de **restrições** que a definem;
  - um **processo de transformação** de um esquema que não verifica a forma normal, recorrendo a decomposição e/ou transformação de relações.

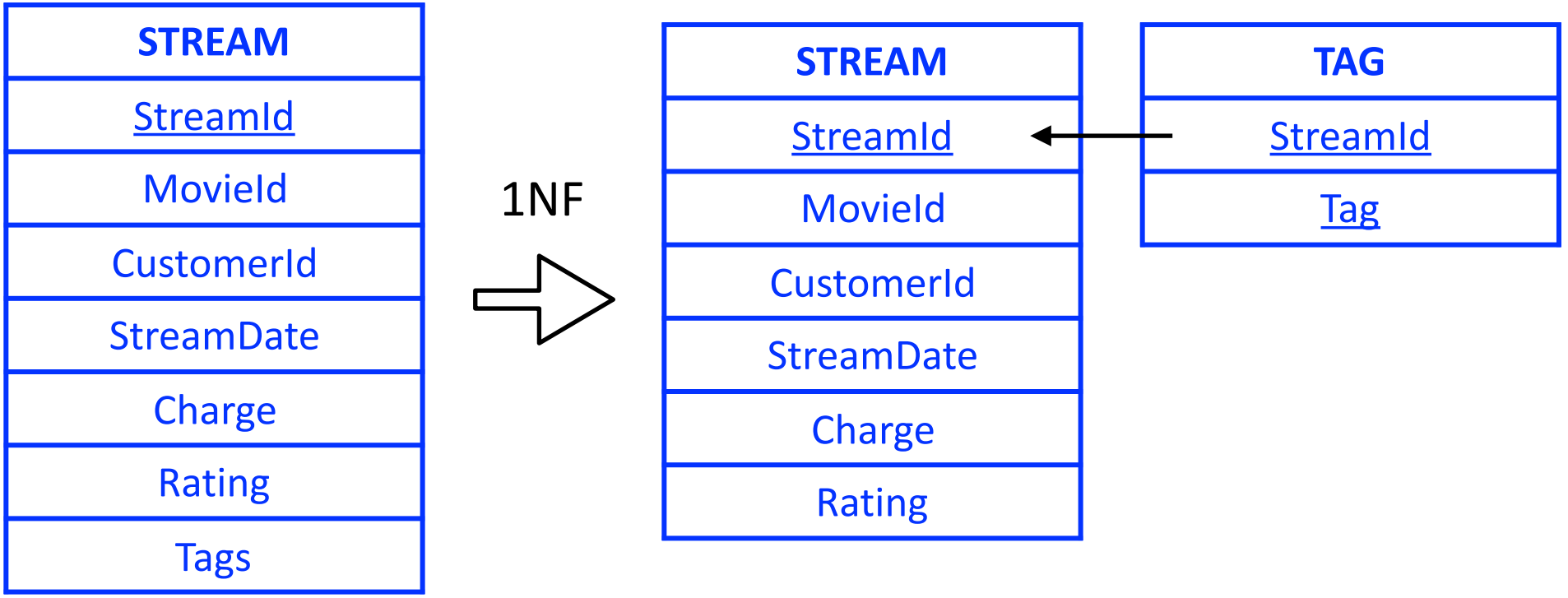
Formas normais (cont.)

- Formas normais, com restrições progressivamente mais fortes:
  - **1NF**: 1ª forma normal ("1st Normal Form")
  - **2NF**: 2ª forma normal ("2nd Normal Form")
  - **3NF**: 3ª forma normal ("3rd Normal Form")
  - **BCNF**: Forma normal de Boyce–Codd ("Boyce-Codd Normal Form")
- Formas normais mais fortes que BCNF (4NF, 5NF, 6NF, 7NF!) são normalmente pouco consideradas/práticas.
- Esquemas 3NF são quase sempre também BCNF.
- Iremos apenas falar de 1NF, 2NF e 3NF.

1NF - 1ª forma normal

- Uma relação diz-se na 1ª forma normal se:
  - tiver uma **chave primária** identificada;
  - e todos os atributos de relações forem **atómicos.**
    - Não são permitidos atributos que implicitamente codificam sub-atributos (atributos compostos) ou atributos multi-valor.
- **Conversão para 1NF**:
  - Uma das chaves candidatas é escolhida para chave primária.
  - Atributos multi-valor implícitos convertidos em nova relação com chave externa referindo a chave primária da tabela original.
  - Cada atributo composto implícito é mapeado em vários sub-atributos atómicos.

Normalização 1NF — exemplo 1



- Relação original decomposta em duas em função do atributo "multi-valor" implícito.

Normalização 1NF — exemplo 2

Suponha que numa empresa os empregados e respectiva dedicação a projectos em nº de horas é representado por uma única relação

STAFF(SId, SName, { Project(PId, PName, Hours) })

STAFF
SId
SName
{ Project(PId, PName, Hours) }

Quanto um atributo multi-valor é também composto, o atributo representada aquilo que se chama uma "relação imbricada".

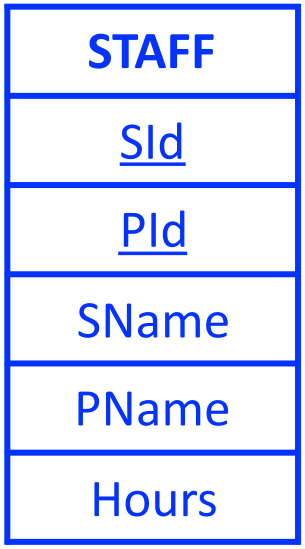
Exemplo de registos:

(1, 'John Doe', { (1,'Proj A', 2), (2,'Proj B', 10) })
(2, 'Maria Silva', { (1,'Proj A', 4), (3,'Proj C', 15) })
(3, 'Manuel Silva', { (1, 'Proj B', 10) })
(4, 'Alberto Silva', { (1,'Proj B', 4) })

Normalização 1NF — exemplo 2

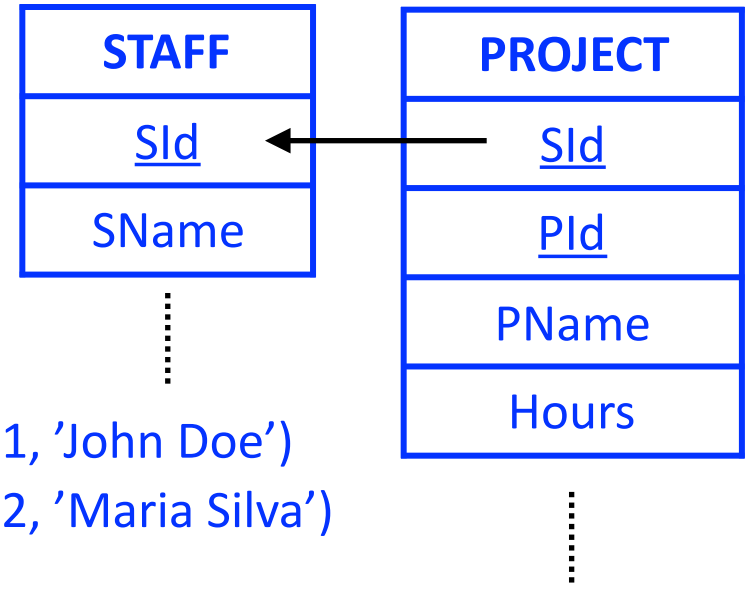
As seguintes 2 alternativas a STAFF(SId, SName, { Project(PId, PName, Hours) }) estão na 1ª forma normal

(1)



(1, 1, 'John Doe','Proj A', 2)
(1, 2, 'John Doe', 'Proj B', 10)
(2, 1, 'Maria Silva', 'Proj A', 4)
(2, 3, 'Maria Silva', 'Proj C', 15)
...

(2)



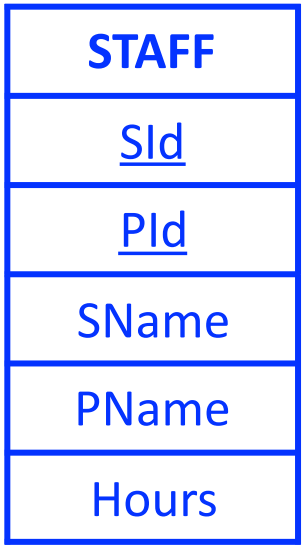
(1, 'John Doe')
(2, 'Maria Silva')
...

(1, 1,'Proj A', 2)
(1, 2, 'Proj B', 10)
(2, 1, 'Proj A', 4)
(2, 3, 'Proj C', 15)
...

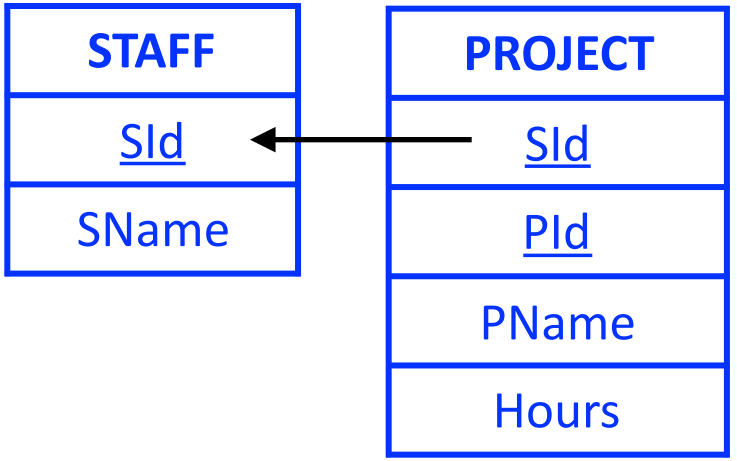
Normalização 1NF — exemplo 2

As seguintes 2 alternativas a STAFF(SId, SName, { Project(PId, PName, Hours) }) estão na 1ª forma normal

(1)



(2)



Nenhuma é inteiramente satisfatória:

- Haverá muita redundância de informação nos dois casos.
- Em **(1)** uma só tabela mistura dados de duas entidades implícitas: empregado e projecto;
- Em **(2)** projecto não existe de forma independente da entidade funcionário — não conseguimos por exemplo definir um projecto sem empregados associados.

Dependência funcional

- Para uma relação R, sendo X e Y sub-conjuntos não-vazios de atributos de R, dizemos que Y **depende funcionalmente de** X ou que X **determina** Y, se:

$$X \to Y \quad \stackrel{\text{def}}{\equiv} \quad \forall r_1, r_2 \in R, \quad r_1[X] = r_2[X] \Longrightarrow r_1[Y] = r_2[Y]$$

- Caso particular — uma chave (primária ou não) K de R determina sempre qualquer sub-conjunto de atributos, isto é,

$$\forall \emptyset \neq X \subseteq \text{Attrs}(R), \quad K \to X$$

Dependências parciais, completas e transitivas

- Uma dependência $X \rightarrow Y$ é **parcial** se pudermos remover algum atributo A de X e o valor Y continuar funcionalmente dependente de $X - A$

$$\exists \emptyset \neq A \subset X : \quad (X - A) \rightarrow Y$$

- Caso contrário a dependência diz-se **completa**:

$$\forall \emptyset \neq A \subset X : \quad (X - A) \nrightarrow Y$$

- Uma dependência $X \rightarrow Y$ é **transitiva** se existir um atributo não-chave Y' tal que

$$X \rightarrow Y' \wedge Y' \rightarrow Y$$

Dependências funcionais — exemplos

(1)

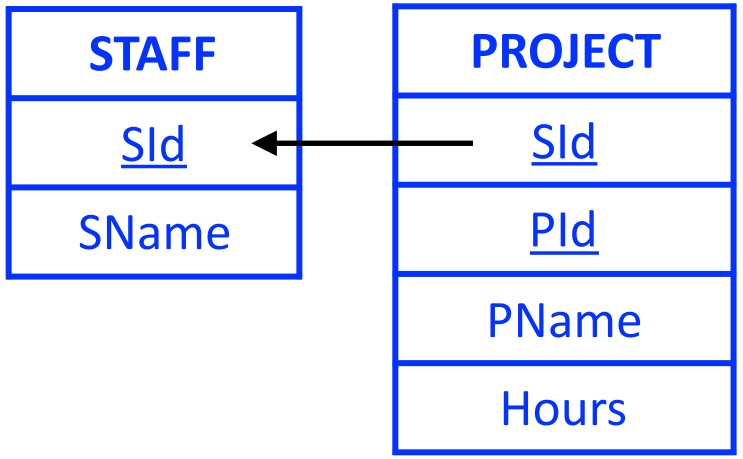
STAFF
SId
PId
SName
PName
Hours

SId → SName

PId → PName

{SId,PId} → Hours

(2)



STAFF.SId → STAFF.SName

PROJECT.PId → PROJECT.PName

{ PROJECT.SId, PROJECT.PId}
→ PROJECT.Hours

Acima detalhamos apenas as dependências funcionais completas. Dependências parciais são irrelevantes na análise p/normalização 2NF e 3NF discutido a seguir, ex:

{ PROJECT.SId, PROJECT.PId} → { PROJECT.PName }

Dependências funcionais — exemplos (cont.)

STREAM
StreamId
MovieId
CustomerId
StreamDate
Charge
Rating
Tags

Assuma que para cada par filme-cliente não deverá haver mais do que um rating ou conjunto de "tags" correspondentes, mesmo que um filme possa ser visto mais do que uma vez por um cliente.

Temos as seguintes **dependências funcionais completas**:

StreamId → { MovieId, CustomerId, StreamDate, Charge}

{CustomerId, MovieId} → {Rating, Tags}

... e a **dependência transitiva**:

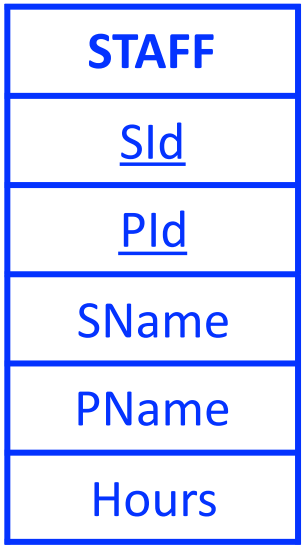
StreamId → {CustomerId, MovieId} → {Rating, Tags}

2ª Forma Normal

- Uma relação *R* está na **2ª forma normal (2NF)** se:
  - **1.** *R* estiver na 1ª forma normal;
  - **2.** nenhum atributo não-chave depende funcionalmente de uma **chave parcial,** um subconjunto estrito de uma chave da tabela (primária ou candidata).
- **Normalização => decomposição em relações tal que os atributos não-chave dependam apenas de chaves primárias.**

Violação da 2NF — exemplos

(1)

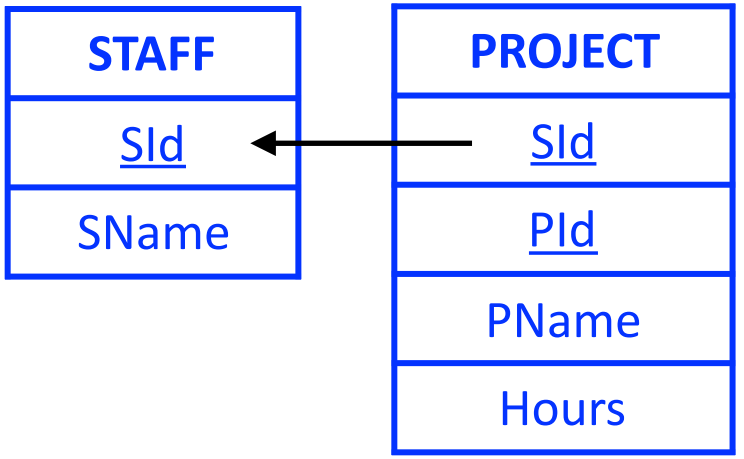


SId → SName

PId → PName

{SId,PId} → Hours

(2)



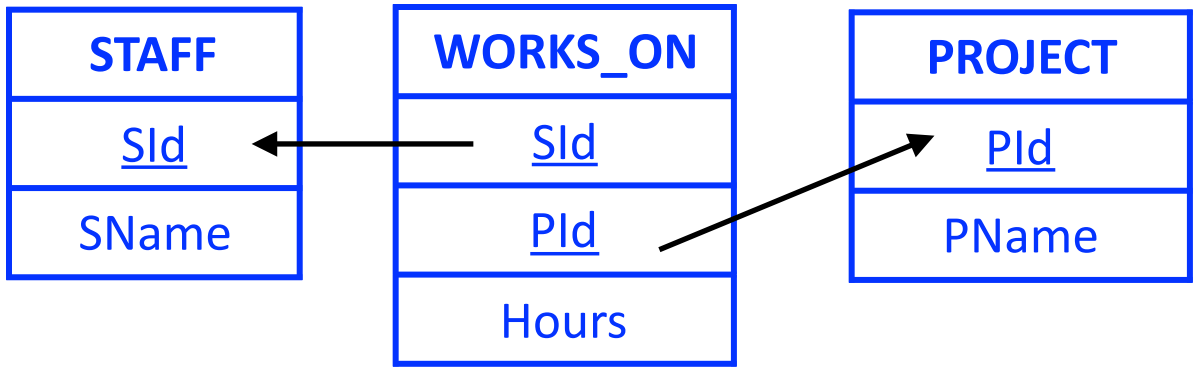
STAFF.SId → STAFF.SName

PROJECT.PId → PROJECT.PName

{ PROJECT.SId, PROJECT.PId}
→ PROJECT.Hours

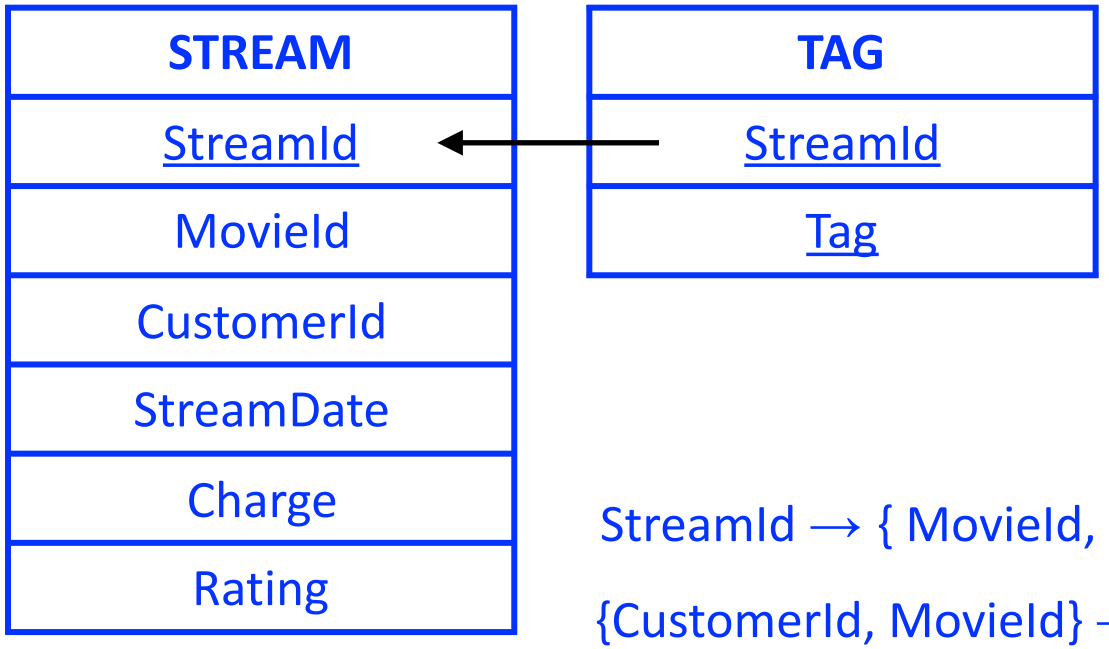
- Em ambos os casos temos uma chave parcial a determinar um atributo não chave — as dependências assinaladas a **vermelho**. Em **(2)** note que **STAFF** está na 2ª forma normal mas **PROJECT** não.

Normalização 2NF — exemplo



- Decompondo **PROJECT** apropriadamente todas as relações ficam na 2ª forma normal. Note também que todas as dependências funcionais do esquema de partida são preservados.

Normalização 2NF – exemplos (cont.)



StreamId → { MovieId, CustomerId, StreamDate, Charge}

{CustomerId, MovieId} → {Rating}

- O esquema é 2NF, mas não expressa que o "rating" e as "tags" deveriam depender apenas de { **CustomerId , MovieId }**

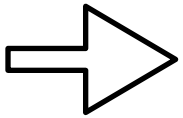
3ª Forma Normal (3NF)

- Uma relação está na **3ª forma normal (3NF)** se:
  - **1.** estiver na 2ª forma normal;
  - **2.** nenhum atributo não-chave depender **transitivamente** da chave primária.
- Exemplo 2NF anterior: **STREAM** não está na 3ª forma normal pois StreamId → { MovieId, CustomerId } → Rating
- Por outras palavras, em uma relação na 3ª forma normal todos os atributos dependem única e exclusivamente da chave primária.
- **Normalização => decomposição em relações tal que nenhum atributo não-chave dependa transitivamente da chaves primária.**

Normalização 3NF — exemplo

STREAM
StreamId
MovieId
CustomerId
StreamDate
Charge
Rating
Tags

3NF



STREAM
StreamId
MovieId
CustomerId
StreamDate
Charge

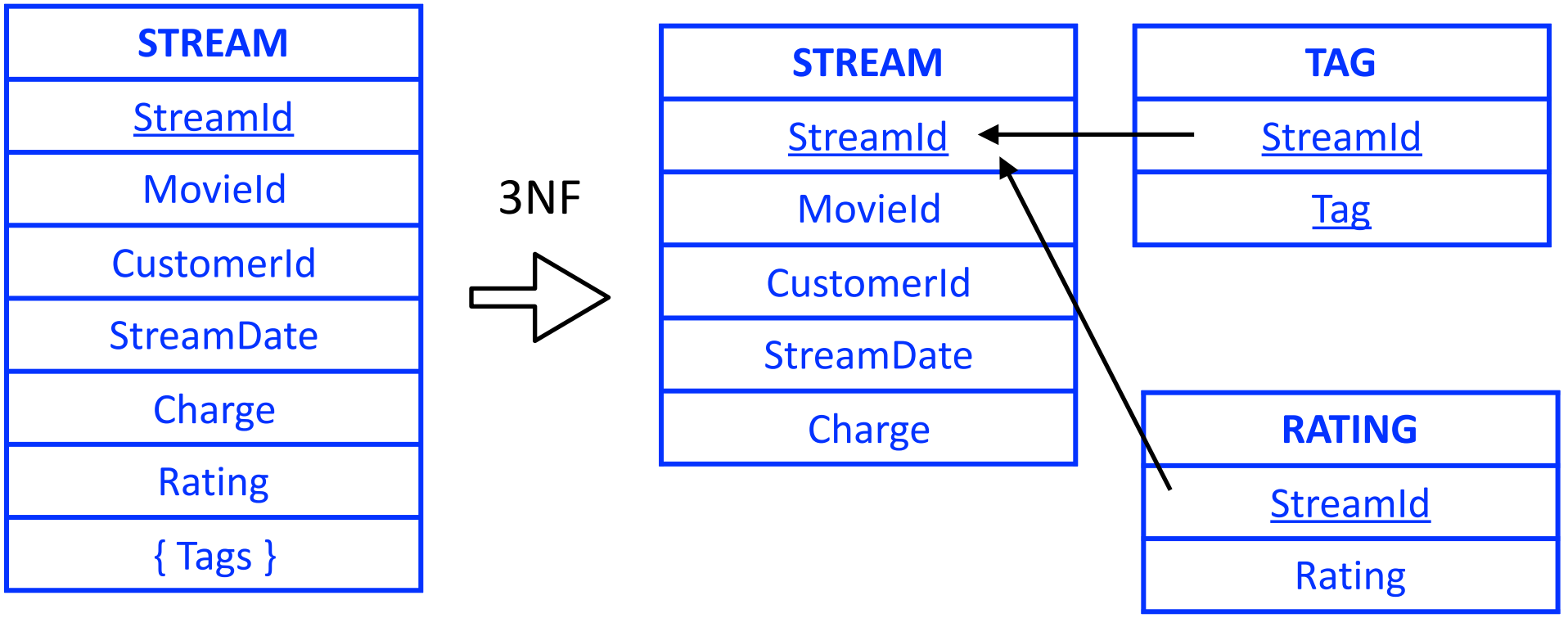
TAG
CustomerId
MovieId
Tag

RATING
CustomerId
MovieId
Rating

StreamId → { MovieId, CustomerId, StreamDate, Charge}

{CustomerId, MovieId} → {Rating, Tags}

Normalização 3NF — "mau exemplo"



StreamId → { MovieId, CustomerId, StreamDate, Charge}

{CustomerId, MovieId} → {Rating, Tags}

Esquema 3NF não expressa a dependência funcional:

{CustomerId, MovieId} → {Rating, Tags}

Exemplos

Exemplo 1 (adaptado do exame de 22-06-2019)

CONCERTO(CId,CArt,CData,CHora,SId,SNome,SLugares,FId,FNome,FPapel)

CId	CArt	CData	CHora	SId	SNome	SLugares	FId	FNome	FPapel
1	Caetano Veloso	12-06-2019	21:00	1	Coliseu	3000	1	Sérgio Abreu	Promotor
1	Caetano Veloso	12-06-2019	21:00	1	Coliseu	3000	2	Maria Menezes	Relações públicas
1	Caetano Veloso	12-06-2019	21:00	1	Coliseu	300	3	Carlos Roberto	Técnico de som
2	Antónia Variations	25-06-2019	21:00	2	Maus Hábitos	300	1	Sérgio Abreu	Relações públicas
2	Antónia Variations	25-06-2019	21:00	2	Maus Hábitos	300	4	Filipa Marques	Promotor
2	Antónia Variations	25-06-2019	21:00	2	Maus Hábitos	300	3	Carlos Roberto	Técnico de som
3	Iggy Carvalho	25-06-2019	23:00	2	Maus Hábitos	300	4	Filipa Marques	Promotor
3	Iggy Carvalho	25-06-2019	23:00	2	Maus Hábitos	300	3	Carlos Roberto	Técnico de som

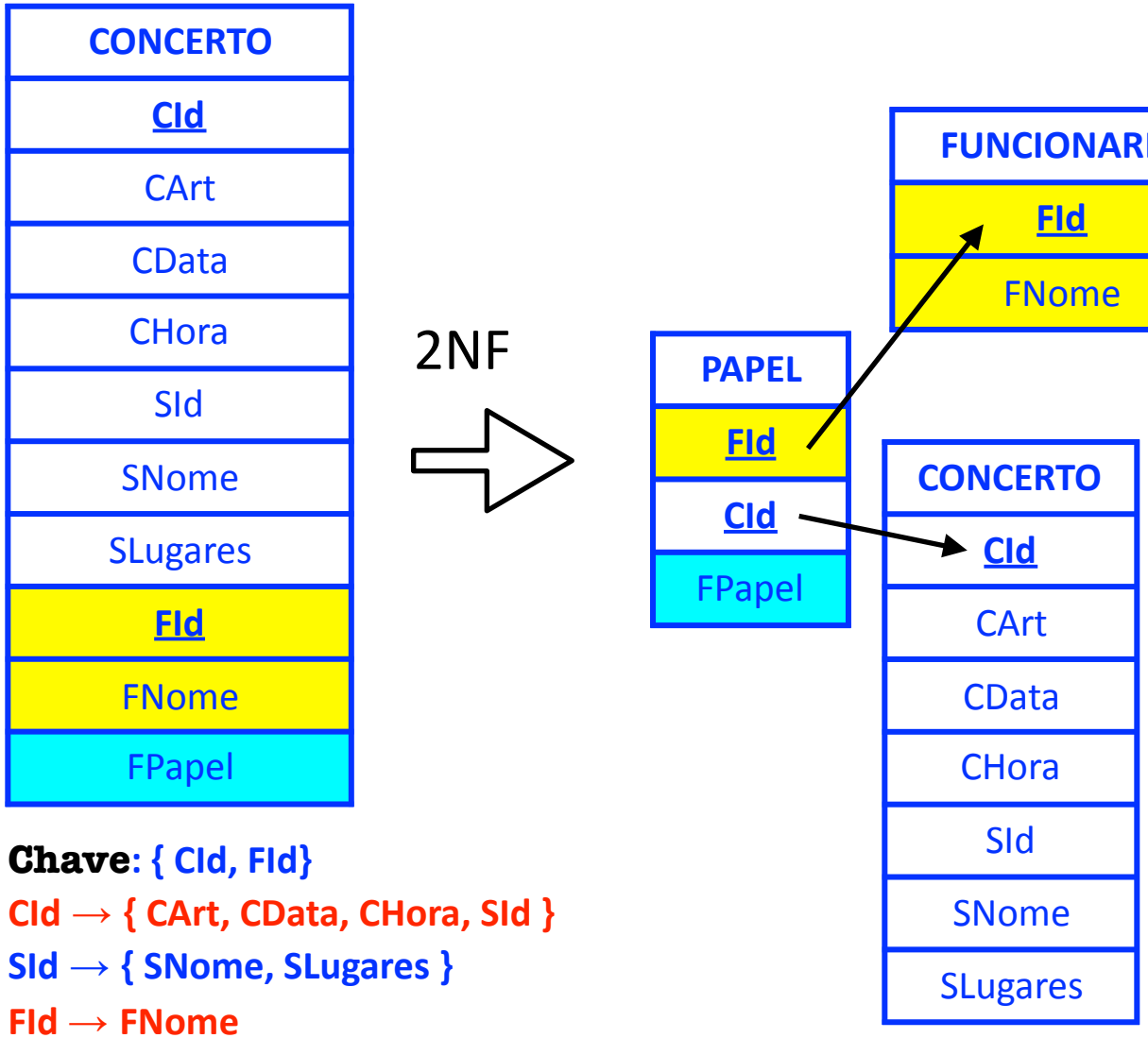
- Considere a BD ilustrada acima para uma empresa de organização de concertos de música.
- Um **concerto** de música é caracterizado por um identificador único, nome do artista ou grupo, data e hora. Um concerto tem lugar numa única **sala de espetáculos**, em que cada sala é caracterizada por um identificador único, um nome, e um número de lugares. Um concerto pode ter associado vários funcionários da empresa com papéis (responsabilidades) diferentes, em que um **funcionário** é caracterizado o um identificador único e um nome. Um funcionário desempenha apenas um papel em cada concerto mas o papel pode ser distinto para concertos diferentes (veja por ex. as entradas envolvendo **Sérgio Abreu** acima).

CONCERTO(CId,CArt,CData,CHora,SId,SNome,SLugares,FId,FNome,FPapel)

CId	CArt	CData	CHora	SId	SNome	SLugares	FId	FNome	FPapel
1	Caetano Veloso	12-06-2019	21:00	1	Coliseu	3000	1	Sérgio Abreu	Promotor
1	Caetano Veloso	12-06-2019	21:00	1	Coliseu	3000	2	Maria Menezes	Relações públicas
1	Caetano Veloso	12-06-2019	21:00	1	Coliseu	300	3	Carlos Roberto	Técnico de som
2	Antónia Variations	25-06-2019	21:00	2	Maus Hábitos	300	1	Sérgio Abreu	Relações públicas
2	Antónia Variations	25-06-2019	21:00	2	Maus Hábitos	300	4	Filipa Marques	Promotor
2	Antónia Variations	25-06-2019	21:00	2	Maus Hábitos	300	3	Carlos Roberto	Técnico de som
3	Iggy Carvalho	25-06-2019	23:00	2	Maus Hábitos	300	4	Filipa Marques	Promotor
3	Iggy Carvalho	25-06-2019	23:00	2	Maus Hábitos	300	3	Carlos Roberto	Técnico de som

- Está na 1ª forma normal ? Não há atributos multi-valor ou compostos implícitos, mas deverá haver uma chave primária. **As dependências funcionais (apenas as completas) são:**
  - **CId → { CArt, CData, CHora, SId }**
  - **SId → { SNome, SLugares }**
  - **FId → FNome**
  - **{ CId, FId } → FPapel**
- **Chave primária**: **{CId, FId}** determina todos os outros atributos. **Não está na segunda forma normal** porque temos as dependências com chaves parciais: **FId → FNome** **e** **CId → { CArt, CData, CHora, SId }.**

Exemplo 1 — conversão para 2NF



Chave: { CId, FId}

CId → { CArt, CData, CHora, SId }

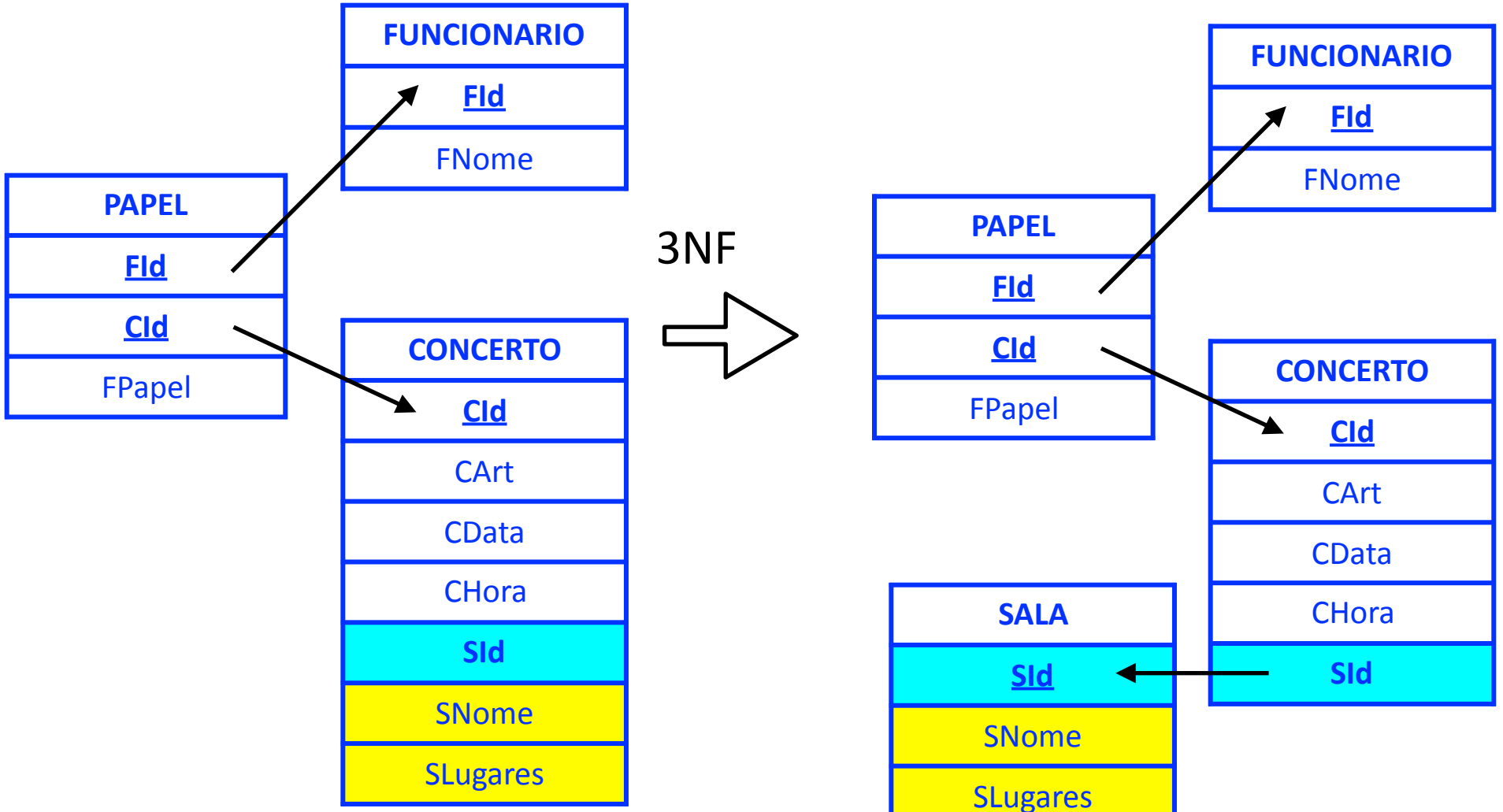
SId → { SNome, SLugares }

FId → FNome

{ CId, FId } → FPapel

- **Normalização para 2NF**: decomposição tendo em conta (apenas) dependências de chaves parciais.
- **Esquema obtido não está na 3ª forma normal**. Observe que na tabela **CONCERTO** ficamos ainda com dependências transitivas : CId → SId → { SNome, SLugares }

Exemplo 1 — conversão para 3NF



CId → SId → { SNome, SLugares }

Exemplo 2 (adaptado do exame de 04-07-2019)

PLAYLIST(ArtId,ArtNome, AlbId, AlbTítulo, AlbAno, FNum, FTítulo, FDuração)

ArtId	ArtNome	AlbId	AlbTítulo	AlbAno	FNum	FTítulo	FDuração
1	Caetano Veloso	1	Abraçaço	2012	1	Um abraçaço	3:50
1	Caetano Veloso	1	Abraçaço	2012	2	Estou triste	5:13
1	Caetano Veloso	1	Abraçaço	2012	8	Vinco	4:38
1	Caetano Veloso	2	Caetanear	1984	8	Sampa	3:17
2	Sonic Youth	3	Goo	1990	8	Mildred Pierce	2:12
2	Sonic Youth	4	Sister	1987	8	Hot wire my heart	3:47
3	Lou Reed	5	Best of	1984	1	Perfect day	3:47
3	Lou Reed	5	Best of	1984	2	Walk on the wild side	4:15
4	Leonard Cohen	6	Best of	1975	4	Bird on the wire	3:27
5	Johnny Cash	7	American Recordings	1994	8	Bird on the wire	4:02

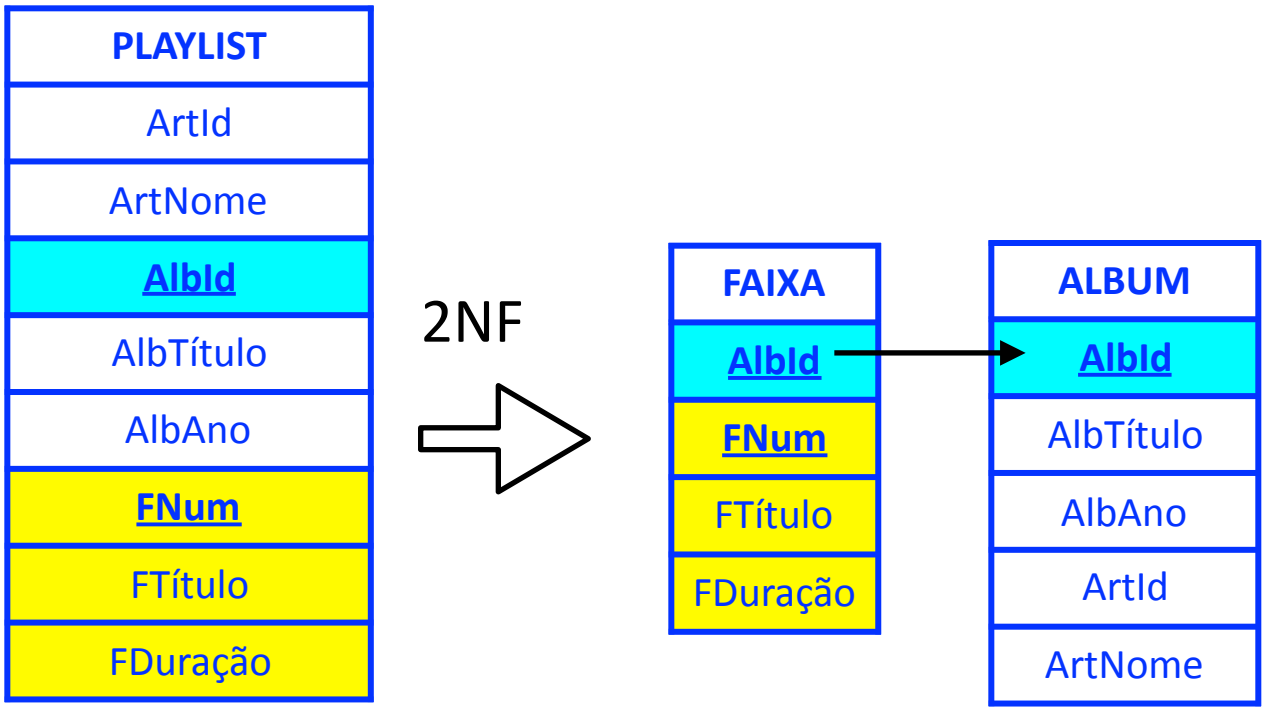
- Considere a BD ilustrada acima para uma "play list" de música.
- Um **artista** é caracterizado por um identificador único e um nome.
- Um **álbum** de um artista tem um identificador único, um título, e ano de edição.
- Uma **faixa de música** tem associado o número de ordem no álbum (único apenas por álbum), um título, e uma duração.

PLAYLIST(ArtId,ArtNome, AlbId, AlbTítulo, AlbAno, FNum, FTítulo, FDuração)

ArtId	ArtNome	AlbId	AlbTítulo	AlbAno	FNum	FTítulo	FDuração
1	Caetano Veloso	1	Abraçaço	2012	1	Um abraçaço	3:50
1	Caetano Veloso	1	Abraçaço	2012	2	Estou triste	5:13
1	Caetano Veloso	1	Abraçaço	2012	8	Vinco	4:38
1	Caetano Veloso	2	Caetanear	1984	8	Sampa	3:17
2	Sonic Youth	3	Goo	1990	8	Mildred Pierce	2:12
2	Sonic Youth	4	Sister	1987	8	Hot wire my heart	3:47
3	Lou Reed	5	Best of	1984	1	Perfect day	3:47
3	Lou Reed	5	Best of	1984	2	Walk on the wild side	4:15
4	Leonard Cohen	6	Best of	1975	4	Bird on the wire	3:27
5	Johnny Cash	7	American Recordings	1994	8	Bird on the wire	4:02

- Está na 1ª forma normal ? Não há atributos multi-valor ou compostos implícitos, mas deverá haver uma chave primária. **As dependências funcionais (completas) são:**
  - **ArtId → ArtNome**
  - **AlbId → { AlbTítulo, AlbAno, ArtId}**
  - **{ AlbId, FNum } → { FTítulo, FDuração}**
- **Chave primária**: **{AlbId, FNum}** determinam todos os outros atributos.
- Não está na segunda forma normal porque temos uma dependência acima que envolve uma chaves parcial: **AlbId → { AlbTítulo, AlbAno, ArtId } .**

Exemplo 2 — conversão para 2NF



Chave: { ArtId, AlbId, FNum}

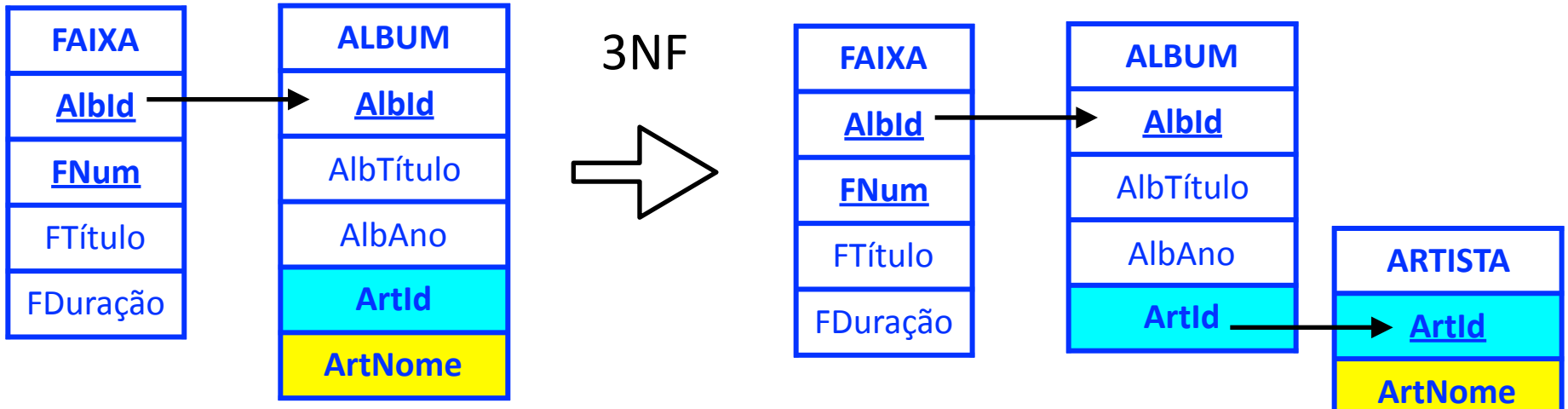
ArtId → ArtNome

AlbId → { AlbTítulo, AlbAno, ArtId}

{ AlbId, FNum }→ { FTítulo, FDuração}

- **Normalização para 2NF**: decomposição tendo em conta (apenas) dependências de chaves parciais.
- **Esquema obtido não está na 3ª forma normal**. Observe que na tabela ALBUM ficamos ainda com dependências transitivas : AlbId → ArtId → ArtNome.

Exemplo 2 — conversão para 3NF



AlbId → ArtId → ArtNome