الأكاديمية العربية الدولية
Arab International Academy

الأكاديمية العربية الدولية

المقررات الجامعية

www.aiacademy.info | care@aiacademy.info

الفصل الثاني:
النموذج العلائقي

العنوان	الصفحة
1. مقدمة	3
2. بنى المعطيات في النموذج العلائقي	3
3. خصائص الجداول	5
4. العلاقات والمفاتيح	5
5. قواعد التكامل	7
6. معالجة البيانات	7
1.6 الاجتماع (Union)	7
2.6 التقاطع (Intersection)	8
3.6 الفرق (Difference)	8
4.6 الجداء (Product)	9
5.6 الاختيار (Selection)	9
6.6 الإسقاط (Projection)	10
7.6 الضم (Join)	10
8.6 التقسيم (Division)	10
7. المراجع	11

الكلمات المفتاحية

علاقة، جدول، تسجيلة، عمود، حقل، نمط بيانات، مفتاح أساسي، مفتاح مستورد، تكامل المعطيات، التنظيم، المنظور المنطقي، المنظور الفيزيائي.

ملخص

يركز هذا الفصل على التعريف بنموذج البيانات العلائقي، والمفاهيم الخاصة به.

الأهداف التعليمية

يهدف هذا الفصل إلى:

- تعريف المفاهيم الخاصة بالنموذج العلائقي للبيانات
- خصائص الجداول في النموذج العلائقي
- دراسة العلاقات وطرق تنفيذها في النموذج العلائقي
- تعريف تكامل المعطيات ومعالجتها
- مقدمة عن التنظيم

1. مقدمة

تم التقديم للنموذج العلائقي في تصميم قواعد البيانات عام 1970 من قبل Dr.E.F.Codd، وقد تطور منذ ذلك الحين من خلال سلسلة من المقالات والكتابات، إلى أن أخذ شكلاً مستقراً حالياً.

تتألف قاعدة البيانات المبنية على النموذج العلائقي، من مجموعة جداول ثنائية البعد، يمثل كل جدول منها كياناً (شخص، مكان، شيئ، حدث ...) له مجموعة من الموصّفات، أو يمثل علاقة بين أكثر من كيان.

ويعني المنظور المنطقي لقاعدة البيانات Logical view النظر للقاعدة كمجموعة من الجداول والعلاقات بينها.

بينما يعني المنظور الداخلي Internal view النظر لقاعدة البيانات كمجموعة من الملفات الفيزيائية وطريقة تخزين على الأقراص الصلبة.

فيما يلي سنعرض أهم المفاهيم الخاصة بالنموذج العلائقي لتصميم قواعد المعطيات، مع شرح مفصل عنها، من خلال الفقرات التالية:

- بنى المعطيات في النموذج العلائقي
- خصائص الجداول
- العلاقات والمفاتيح
- قواعد التكامل
- معالجة البيانات

2. بنى المعطيات في النموذج العلائقي

قاعدة المعطيات العلائقية هي مجموعة من الجداول.

الجدول هو بنية ثنائية البعد تتألف من أعمدة وأسطر. لكل عمود اسم وحيد ونمط معطيات ويمثل العمود موصِّفة للكيان الذي يعبر عنه الجدول أو للعلاقة التي نتج عنها الجدول.

السطر في الجدول يمثل ورود لأحد عناصر الكيان، فمثلاً إذا كان الجدول يحمل بيانات موظفي شركة، فكل عمود يمثل موصِّفة للموظف (اسمه، تاريخ توظيفه، منصبه،...) وكل سطر يمثل بيانات موظف محدد.

تقاطع السطر والعمود يمثل قيمة موصِّفة لأحد عناصر الكيان (تقاطع عمود تاريخ التوظيف مع السطر الخاص بالموظف Scott يمثل تاريخ توظيف Scott في الشركة).

الشكل التالي يظهر التصميم العلائقي لقاعدة بيانات تخص مجموعة من الكتب وعلاقتها بالمؤلفين ودور النشر:

A Relational Data Base

AUTHOR

au_id	au_lname	au_fname	address	city	state
172-32-1176	White	Johnson	10932 Bigge Rd.	Menlo Park	CA
213-46-8915	Green	Marjorie	309 63rd St. #411	Oakland	CA
238-95-7766	Carson	Cheryl	589 Darwin Ln.	Berkeley	CA
267-41-2394	O'Leary	Michael	22 Cleveland Av. #14	San Jose	CA
274-80-9391	Straight	Dean	5420 College Av.	Oakland	CA
341-22-1782	Smith	Meander	10 Mississippi Dr.	Lawrence	KS
409-56-7008	Bennet	Abraham	6223 Bateman St.	Berkeley	CA
427-17-2319	Dull	Ann	3410 Blonde St.	Palo Alto	CA
472-27-2349	Gringlesby	Burt	PO Box 792	Covelo	CA
486-29-1786	Locksley	Charlene	18 Broadway Av.	San Francisco	CA

TITLE

title_id	title	type	price	pub_id
BU1032	The Busy Executive's Database Guide	business	19.99	1389
BU1111	Cooking with Computers	business	11.95	1389
BU2075	You Can Combat Computer Stress!	business	2.99	736
BU7832	Straight Talk About Computers	business	19.99	1389
MC2222	Silicon Valley Gastronomic Treats	mod_cook	19.99	877
MC3021	The Gourmet Microwave	mod_cook	2.99	877
MC3026	The Psychology of Computer Cooking	UNDECIDED		877
PC1035	But Is It User Friendly?	popular_comp	22.95	1389
PC8888	Secrets of Silicon Valley	popular_comp	20	1389
PC9999	Net Etiquette	popular_comp		1389
PS2091	Is Anger the Enemy?	psychology	10.95	736

PUBLISHER

pub_id	pub_name	city
736	New Moon Books	Boston
877	Binnet & Hardley	Washington
1389	Algodata Infosystems	Berkeley
1622	Five Lakes Publishing	Chicago
1756	Ramona Publishers	Dallas
9901	GGG&G	München
9952	Scootney Books	New York
9999	Lucerne Publishing	Paris

AUTHOR_TITLE

au_id	title_id
172-32-1176	PS3333
213-46-8915	BU1032
213-46-8915	BU2075
238-95-7766	PC1035
267-41-2394	BU1111
267-41-2394	TC7777
274-80-9391	BU7832
409-56-7008	BU1032
427-17-2319	PC8888
472-27-2349	TC7777

الجدول التالي يظهر مجموعة من المصطلحات المتداولة في النموذج العلائقي، مع المرادفات المستخدمة لكل منها:

In This Document	Formal Terms	Many Database Manuals
Relational Table	Relation	Table
Column	Attribute	Field
Row	Tuple	Record

يمكن عرض تصميم قاعدة البيانات السابقة من خلال العبارات التالية (بدون المعطيات المتضمنة في الجداول وبدون تحديد أنماط الحقول):

AUTHOR	(au_id, au_lname, au_fname, address, city, state, zip)
TITLE	(title_id, title, type, price, pub_id)
PUBLISHER	(pub_id, pub_name, city)
AUTHOR_TITLE	(au_id, title_id)

الجدول AUTHOR_TITLE يعبر عن العلاقة بين المؤلفين والكتب، بينما تعبر الجداول الأخرى عن الكيانات: مؤلف، كتاب، دار نشر.

3. خصائص الجداول

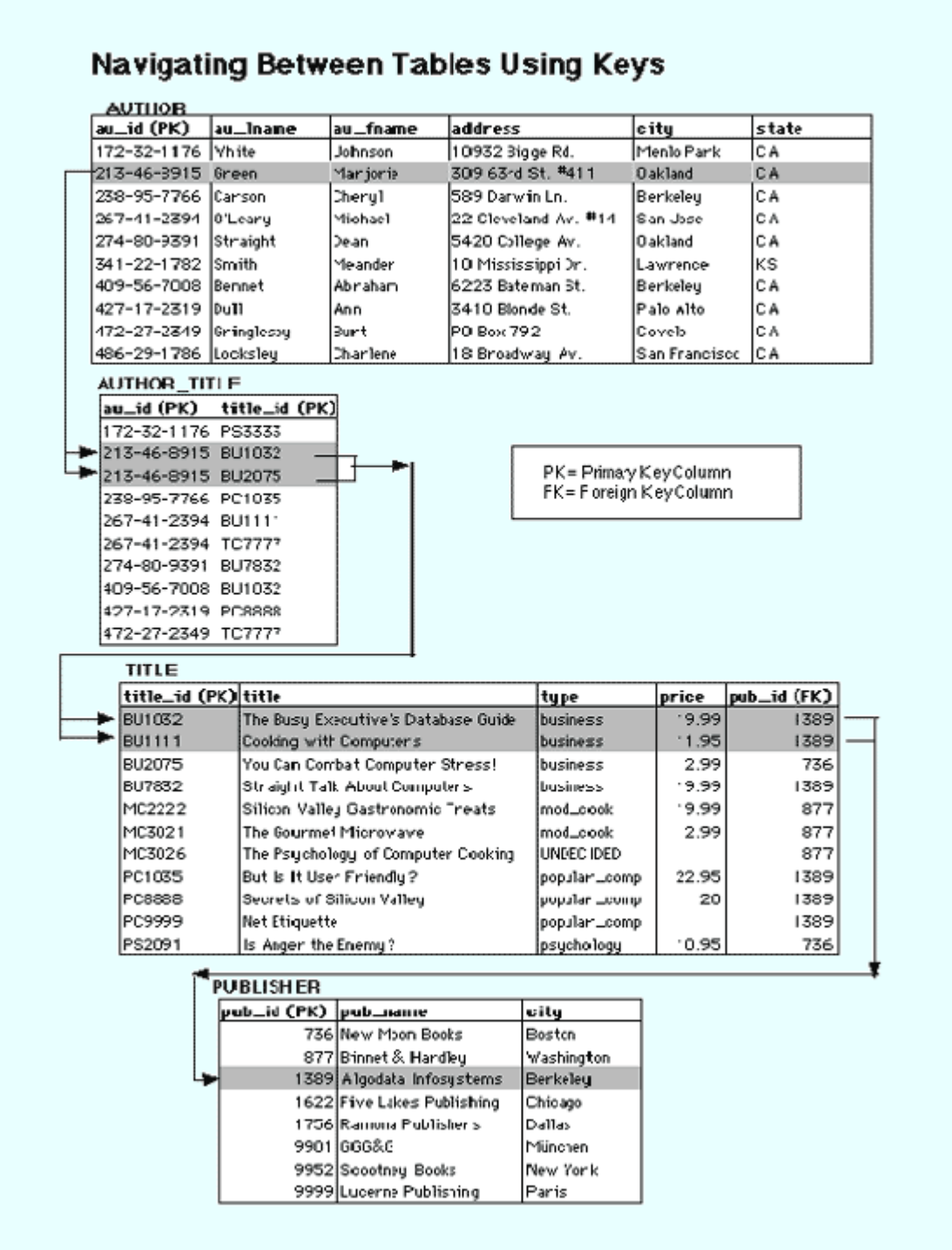
- بنية ثائية مؤلفة من أعمدة وأسطر
- يمثل كل سطر (تسجيلة) كياناً واحداً من مجموعة الكيانات
- يمثل كل حقل في الجدول واصفة، وله اسم مميز
- تمثل تقاطعات الأسطر والأعمدة قيمة معطيات واحدة
- ينبغي أن تطابق جميع القيم في حقل صيغة معطيات واحدة، كأن تكون كلها أعداد صحيحة أو أن تكون كلها من نمط تاريخ
- لكل عمود مجال محدد من القيم يعرف باسم مجال الواصفات
- ترتيب الأسطر والأعمدة غير مهم بالنسبة لنظام إدارة قواعد البيانات، ويمكن استرجاعها بالترتيب المطلوب
- يجب أن يحوي كل جدول على واصفة أو مجموعة واصفات تميز كل سطر عن غيره (مفتاح أساسي)

4. العلاقات والمفاتيح

العلاقة Relationship هي الرابط بين جدولين أو أكثر، يعبر عنها في قاعدة المعطيات من خلال المفتاح الأساسي Primary key والمفتاح المستورد Foreign key.

المفتاح الأساسي Primary key هو حقل أو مجموعة حقول تميز بمجموعها كل تسجيلة في الجدول.

المفتاح المستورد Foreign Key هو حقل قيمته تطابق حتماً قيمة مفتاح أساسي في جدول آخر (ان لم تكن NULL)، ويمكن النظر للمفتاح المستورد على أنه نسخة من قيمة مفتاح أساسي في جدول آخر، تحدد التسجيلة المرتبطة من الجدول الآخر بالتسجيلة الحاوية على المفتاح المستورد من هذا الجدول.



Au_Id في الجدول Author_Title هي مفتاح مستورد من المفتاح الأساسي Au_id في الجدول Author.

5. قواعد التكامل

يضمن التكامل للمستخدم التجوال والمعالجة الصحيحة للبيانات في جداول قاعدة المعطيات، ويقصد به نوعين من التكامل:

- تكامل المعطيات: وهو يعني أن تكون قيم المفتاح الأساسي فريدة، وألا يكون أي جزء من المفتاح الأساسي Null، وذلك لضمان أن يكون لكل كيان هوية مميزة، ولضمان أن تكون قيم المفاتيح المستوردة تشير بشكل صحيح إلى تسجيلات محتواة في الجدول الأساسي.
- التكامل المرجعي: يقصد به أن تكون قيمة المفتاح المستورد إما Null أو قيمة موجودة في حقل المفتاح الأساسي للجدول الذي تم الاستيراد منه.

6. معالجة البيانات

الجداول هي مجموعات عناصرها التسجيلات، والعمليات الممكن إجراؤها على المجموعات يمكن إجراؤها أيضاً على الجداول، هذه العمليات هي:

الاجتماع (Union)، التقاطع (Intersection)، الفرق (Difference)، الجداء (Product)، الاختيار (Selection)، الإسقاط (Projection)، الضم (Join)، التقسيم (Division)

الاجتماع (Union)

تجمع هذه العملية كافة الأسطر من الجدولين، دون تكرار الأسطر الموجودة في الجدولين. لتطبيق هذه العملية يجب أن يتطابق الجدولين في ترتيب وأنماط الأعمدة.

A

K	X	Y
1	A	2
2	B	4
3	C	6

B

K	X	Y
1	A	2
4	D	8
5	E	10

A UNION B

K	X	Y
1	A	2
2	B	4
3	C	6
4	D	8
5	E	10

التقاطع (Intersection)

ينتج عن هذه العملية جدول يضم الأسطر المشتركة بين الجدولين الأساسيين، يجب أن يكون الجدولان المطبق عليهما هذه العملية منسجمان من حيث عدد الأعمدة وترتيبها وأنماطها.

A

K	X	Y
1	A	2
2	B	4
3	C	6

B

K	X	Y
1	A	2
4	D	8
5	E	10

A INTERSECT B

K	X	Y
1	A	2
2	B	4
3	C	6
4	D	8
5	E	10

الفرق(Difference)

ينتج عن هذه العملية جدول يتضمن الأسطر التي تظهر في الجدول الأول ولا تظهر في الجدول الثاني.

A

K	X	Y
1	A	2
2	B	4
3	C	6

B

K	X	Y
1	A	2
4	D	8
5	E	10

A – B

K	X	Y
2	B	4
3	C	6

B – A

K	X	Y
4	D	8
5	E	10

الجداء (Product):

ينتج عن هذه العملية كافة أزواج التسجيلات الممكنة من كلا الجدولين.

A

K	X	Y
1	A	2
2	B	4
3	C	6

B

K	X	Y
1	A	2
4	D	8
5	E	10

A TIMES B

AK	AX	AY	BK	BX	BY
1	A	2	1	A	2
1	A	2	4	D	8
1	A	2	5	E	10
2	B	4	1	A	2
2	B	4	4	D	8
2	B	4	5	E	10
3	C	6	1	A	2
3	C	6	4	D	8
3	C	6	5	E	10

الاختيار (Selection)

ترجع هذه العملية مجموعة جزئية من أسطر الجدول، المجموعة الجزئية تحقق شرط معين.

A

K	X	Y
1	A	2
2	B	4
3	C	6

SELECT BASED ON A CONDITION

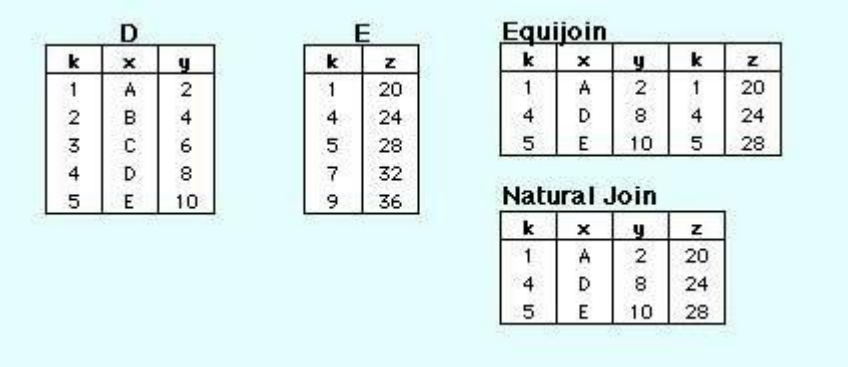
K	X	Y
2	B	4
3	C	6

الإسقاط (Projection)

يرجع مجموعة جزئية من أعمدة الجدول.

الضم (Join)

تسمح هذه العملية بجمع الواصفات من جدولين أو أكثر، هذه العملية هي من أهم ميزات نظم قواعد المعطيات العلائقية، إذ أنها تسمح بربط جداول مستقلة عن بعضها من خلال واصفات مشتركة.



D

k	x	y
1	A	2
2	B	4
3	C	6
4	D	8
5	E	10

E

k	z
1	20
4	24
5	28
7	32
9	36

Equijoin

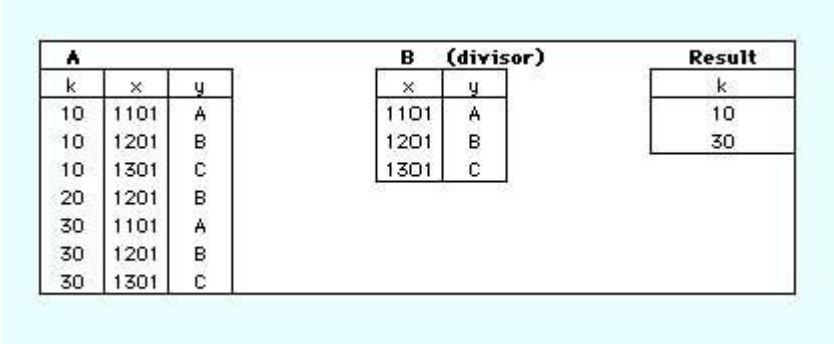
k	x	y	k	z
1	A	2	1	20
4	D	8	4	24
5	E	10	5	28

Natural Join

k	x	y	z
1	A	2	20
4	D	8	24
5	E	10	28

التقسيم (Division)

ينتج عن هذه العملية جدول بقيم أعمدة متممها من أعمدة الجدول الأول موجود كأسطر في الجدول الثاني.



A

k	x	y
10	1101	A
10	1201	B
10	1301	C
20	1201	B
30	1101	A
30	1201	B
30	1301	C

B (divisor)

x	y
1101	A
1201	B
1301	C

Result

k
10
30

7. المراجع:

- http://database.ittoolbox.com
- http://www.utexas.edu/its/windows/database/datamodeling/Rm/

الفصل الثالث:
تصميم قواعد المعطيات

العنوان	الصفحة
1. مقدمة	3
2. دورة حياة النظام المعلوماتي	3
1.2 التخطيط	3
2.2 التحليل	4
3.2 التصميم	4
4.2 التنفيذ	4
5.2 الصيانة	4
3. دورة حياة قاعدة المعطيات	5
1.3 الدراسة الأولية لقاعدة المعطيات	5
2.3 تصميم قاعدة المعطيات	6
3.3 التنفيذ والتحميل	8
4.3 الاختبار والتقييم	8
5.3 التطبيق	9
6.3 الصيانة والتطوير	9
4. دورة حياة نظام المعلومات ودورة حياة قاعدة المعطيات	9

ملخص

يشرح هذا الفصل مراحل بناء نظام معلوماتي، ومراحل بناء قاعدة معطيات مع دراسة التقاطع بين بناء قاعدة المعطيات وبناء النظام المعلوماتي المعتمد عليها.

الأهداف التعليمية

يتضمن هذا الفصل:

- دورة حياة النظام المعلوماتي
- دورة حياة قاعدة المعطيات
- العلاقة بين مراحل بناء النظام المعلوماتي ومراحل بناء قاعدة المعطيات

1. مقدمة

يعتبر تصميم قاعدة المعطيات جزءاً من عمل أشمل هو تصميم نظام معلوماتي، غايته تحويل المعطيات المخزنة في القاعدة إلى معلومات مفيدة. وبالتالي فإن بناء قاعدة المعطيات هو جزء من دورة حياة النظام المعلوماتي.

يقصد بتحويل المعطيات إلى معلومات مفيدة، عرض المعطيات بطريقة معالجة ومقروءة كأن تعرض على شكل مخططات أو تقارير أو تلخص على شكل إجماليات يمكن استنتاج معلومات أخرى منها ويمكن الاعتماد عليها في اتخاذ القرارات.

أما النظام المعلوماتي فيقصد به مجموعة من الأدوات المساعدة على جمع المعطيات وتخزينها في قاعدة بيانات ومن ثم استرجاعها وعرضها بطرائق مختلفة. ويتكون من مجموعة من العتاد المادي والبرمجيات الأساسية ومعطيات وتطبيقات مختلفة ومترابطة.

يعتمد أداء نظام المعلومات على:

1. تصميم وبناء قواعد البيانات.
2. تصميم وتحقيق التطبيقات.
3. الإجراءات الإدارية.

2. دورة حياة النظام المعلوماتي

تتألف دورة حياة النظام المعلوماتي من المراحل الخمس التالية:

1. التخطيط
2. التحليل
3. التصميم
4. التنفيذ
5. الصيانة

1.2 التخطيط

يتم في هذه المرحلة بناء توصيف عام للمؤسسة وتتم فيه الإجابة على الأسئلة التالية:

- هل يوجد نظام يتم العمل عليه حالياً، وفي حال وجود نظام هل ينبغي الاستمرار باستخدامه
- هل ينبغي إجراء تعديلات على النظام الحالي في حال وجوده
- هل ينبغي استبدال النظام الحالي بنظام آخر

وفي حال تمت الإجابة على الأسئلة السابقة لاستبدال النظام الموجود أو بناء نظام جديد، فعندها يجب أن تشمل مرحلة التخطيط ما يلي:

- المتطلبات التقنية للنظام الجديد (العتاد والبرمجيات المطلوبة)
- الكلفة التقريبية للنظام الجديد

2.2 التحليل

يتم في هذه المرحلة دراسة محتويات المرحلة السابقة بتفصيل أكثر، وتحديد احتياجات المؤسسة واحتياجات الأفراد، كما يتم أيضاً تحديد العتاد المطلوب بدقة والبرمجيات الأساسية المطلوبة كنظام إدارة واستثمار الشبكات ونظام الاستثمار على الطرفيات وأي برمجيات أخرى مطلوبة. وينتج عن هذه المرحلة تقرير مفصل عن المتطلبات الوظيفية للنظام المعلوماتي المطلوب.

يمكن تلخيص محتويات التقرير الأساسية بما يلي:

1. توصيف وظائف النظام المطلوب.
2. توصيف الورقيات الناتجة أو المستخدمة للإدخال، مع مخطط تدفق الوثائق.
3. البنية المفاهيمية لقاعدة البيانات التي يمكن تخزين (ERD) فيها (المعطيات المطلوبة لتحقيق وظائف النظام)
4. المتطلبات المادية والبرمجية اللازمة لتحقيق النظام.

(مخطط تدفق الوثائق هو مخطط تظهر فيه الوثائق المتداولة في المؤسسة مع شرح محتوياتها واتجاه حركتها بين فعاليات المؤسسة المختلفة).

3.2 التصميم

يتم في هذه المرحلة تصميم قاعدة البيانات[1]، ويتم تصميم المكونات الأخرى للنظام المعلوماتي[2].

4.2 التنفيذ

يتم في هذه المرحلة بناء قاعدة المعطيات واختبارها، ومن ثم بناء التطبيقات المطلوبة واختبارها مع المستخدمين.

5.2.الصيانة

يتنبه المستثمرون عادةً بعد اختبار النظم المطورة لمجموعة من التعديلات، قد تفرض هذه التعديلات تغييراً في بنية قاعدة البيانات مع تغييرات على البرمجيات المطورة، يتم في هذه المرحلة تعديل النظام المعلوماتي بما يتلاءم مع احتياجات المؤسسة، ويمكن تمييز ثلاثة أنواع من الصيانة في هذه المرحلة هي:

---

1 (بنية منطقية وفيزيائية للقاعدة)

2 (مخطط تدفق الشبكة المعلوماتية في حال وجودها، المجتزءات المكونة للتطبيقات والإجراءات المكونة للبرمجيات المطلوبة)

- صيانة علاجية، يتم تطبيقها عند وقوع أخطاء معينة
- صيانة تطويرية، تفرضها تغييرات في بيئة العمل
- صيانة كمالية، تفيد في تحسين وتطوير النظام من حيث الأداء والوظائف

3. دورة حياة قاعدة المعطيات

تتألف دورة حياة قاعدة المعطيات من المراحل التالية:

وفيما يلي شرح مفصل لكل من هذه المراحل.

1. الدراسة الأولية.
2. تصميم قاعدة المعطيات.
3. التنفيذ والتحميل.
4. الاختبار والتقييم.
5. التطبيق.
6. الصيانة والتطوير.

1.3 الدراسة الأولية لقاعدة المعطيات

يتم في هذه المرحلة:

- تحليل وضع المؤسسة
- تعريف وتحديد المشاكل والقيود
- تحديد الأهداف
- تحديد المجال والحدود

تحليل وضع المؤسسة

تتضمن هذه المرحلة توصيف بنية المؤسسة وهرميتها الإدارية، مع تقسيمها إلى مواقع عمل نمطية مع شرح لعلاقة هذه المواقع ببعضها. ويعني موقع العمل مجموعة وظائف مترابطة منطقياً تترجم عند بناء النظام المعلوماتي إلى تطبيق وحيد يتم تنصيبه على حاسب طرفي أو مجموعة حواسب حسب حجوم الإدخال وعدد المستثمرين الذي تتطلبه هذه الإدخالات.

تعريف وتحديد المشاكل والقيود

يتم في هذه المرحلة تحديد المشاكل الفنية التي قد تنشأ عن العلاقة بين الوظائف المختلفة، والمشاكل التي يمكن أن تنتج عن الترابط مع أنظمة موجودة مسبقاً، كما يتم تحديد القيود الزمنية والمالية التي يجب أن يتم التطوير ضمن حدودها.

تحديد الأهداف

هنا يجب على مصمم قاعدة البيانات التأكد من مطابقة أهداف المستثمرين من النظام لأهداف المؤسسة وتوصيف هذه الأهداف للتأكد من إمكانية تحقيقها، والتأكد من تحقيقها بعد إتمام النظام.

تحديد المجال والحدود

ويقصد بذلك تحديد المجال الذي يغطيه النظام المطلوب[3] من وظائف المؤسسة. هذا التحديد يفيد في تقدير حجوم المدخلات وتحديد بنى المعطيات المناسبة وشكل وعدد كيانات القاعدة وحجم القاعدة التي يجب تصميمها.

2.3 تصميم قاعدة المعطيات

يتم في هذه المرحلة توصيف كامل عناصر قاعدة البيانات المطلوبة لتحقيق النظام المعلوماتي، بما في ذلك من كيانات وارتباطات بينها، مع خصائص كل علاقة بين كيانين وواصفات كل كيان ونمط كل واصفة ومجال تعريفها، ويتم التصميم على أربعة مراحل هي:

- التصميم المفهومي (Conceptual)
- اختيار نظام إدارة قواعد البيانات
- التصميم المنطقي (Logical)
- التصميم الفيزيائي (Physical)

التصميم المفهومي

يكون التصميم في هذه المرحلة مستقلاً عن العتاد والأدوات البرمجية التي سيتم استخدامها، وهو يمثل وصفاً للمتطلبات التي ستحققها قاعدة البيانات.

يبدأ التصميم المفهومي بتحليل المعطيات والمتطلبات، حيث تمثل المعطيات مدخلات النظام الواجب تخزينها في قاعدة المعطيات، والمتطلبات هي المعلومات التي سيتم استخلاصها من المعطيات لتساعد في اتخاذ القرار، وهي مخرجات النظام.

يتم في هذه المرحلة وصف:

- مدخلات النظام ومخرجاته
- مستخدمي المعلومات
- حجوم المعطيات وأنماطها والعلاقة فيما بينها
- مصادر المعلومات وكيفية استخلاصها من المعطيات

3 هل يشمل النظام وبالتالي قاعدة البيانات جزءاً محدداً من فعاليات المؤسسة أو مجموعة أجزاء؟ وما هي هذه الأجزاء؟

ويستخلص محلل النظم المعلومات السابقة من:

- مراقبة النظام الحالي المعمول به
- التعاون مع المستثمرين
- جمع المعطيات وتنسيقها وتحليل العلاقات فيما بينها

بعد تحليل المعطيات والمتطلبات، ينتقل محلل النظم إلى سرد قواعد العمل التي توصف بلغة بسيطة قواعد الحساب (مثال: قيمة بند في فاتورة تساوي جداء السعر بالكمية، قيمة الفاتورة تساوي مجموع قيم بنودها) وقواعد الضبط (مثال: لا يمكن لزبون سحب دفعة من حسابه، إذا كان رصيد الحساب لا يغطي المبلغ المطلوب).
وتؤمن معرفة قواعد العمل وفهمها من قبل مصمم قاعدة البيانات فهماً للمؤسسة وكيفية إدارة أعمالها.

تفيد معرفة قواعد العمل في:

- عرض معطيات المؤسسة بأسلوب قياسي
- تشكل صلة الوصل بين المستخدمين والمصممين
- تسمح للمصمم بتشكيل فكرة عن طبيعة ودور ومجال المعطيات
- فهم إجراءات العمل المستخدمة
- تسمح للمصمم بوضع علاقات مناسبة بين الكيانات (تعرف العلاقات الإجبارية والاختيارية ووجود الربط بين كيانين أو عدم وجوده)

الخطوة التالية في التصميم المفهومي لقاعدة البيانات هي وضع مخطط الكيانات – ارتباطات (ERD) الذي يتم إنشاؤه وفق الخطوات التالية:

- عزل الكيانات المطلوبة
- وضع العلاقات بين الكيانات
- تحديد واصفات الكيانات والمفاتيح الأساسية والمفاتيح المستوردة فيها
- تنظيم الكيانات
- تدقيق المخطط الناتج مع المستثمرين للتأكيد على شموله لمتطلبات المستثمرين
- تعديل المخطط كنتيجة لتدقيقه مع المستثمرين

المرحلة الأخيرة في التصميم المفهومي، هي التحقق من النموذج الذي تم بناؤه. وذلك من خلال اختبارات تشمل المتطلبات والقيود التي تفرضها قواعد العمل، وغالباً ما يؤدي ذلك إلى تعديل في بنية قاعدة المعطيات.
في حال تصميم قاعدة بيانات موزعة، ينبغي على المصمم دراسة اعتبارات خاصة تناسب منطق قواعد المعطيات الموزعة، خاصة فيما يتعلق بتوزيع المعطيات واستراتيجيات التحصيص المتبعة.

في حال التصميم الموزع لقاعدة البيانات، يتم بناء القاعدة في عدة مواقع فيزيائية، ويمكن للإجراءات أن تقوم بالولوج إلى أقسام القاعدة في المواقع المختلفة.

اختيار نظام إدارة قواعد المعطيات

يكون المستوى المفهومي في تصميم قاعدة البيانات مستقل عن نظام إدارة قواعد البيانات المستخدم، على خلاف المستوى المنطقي كونه ينتهي بنص بناء قاعدة المعطيات. ولذلك فقبل وضع التصميم المنطقي لقاعدة المعطيات يجب اختيار نظام إدارة قواعد المعطيات، ويتم ذلك على أساس:

- كلفة الشراء والصيانة والدعم الفني والتدريب على الاستخدام والإدارة
- النموذج المبني على أساسه (هرمي، شبكي، علائقي، غرضي...)
- المتطلبات العتادية لتنصيب وتشغيل النظام (ذاكرة، أقراص صلبة...)

التصميم المنطقي لقاعدة المعطيات

تتم في هذه المرحلة ترجمة عناصر القاعدة التي تظهر في التصميم المفهومي إلى مقابلاتها في نظام إدارة قواعد البيانات المستخدم.

يتضمن التصميم المنطقي لقاعدة معطيات علائقية، كافة التوصيفات للجداول والفهارس والمناظير والمناقلات وحقوق وسماحيات الولوج إلى المعطيات.

التصميم الفيزيائي لقاعدة المعطيات

يتم في هذه المرحلة وصف بنية القاعدة من حيث الملفات الفيزيائية المكونة لها، مع حجومها البدائية وطريقة تجميعها وتوزيعها على الأقراص الصلبة، واستراتيجيات النسخ الاحتياطي لها، مع بعض المعاملات الأخرى التي تتعلق بنظام إدارة قواعد البيانات المستخدم، والتي تفيد في تحسين أداء قاعدة البيانات (مثل File groups في حال SQL Server، و Tablespace في حال Oracle).

3.3 التنفيذ والتحميل

يتم في هذه المرحلة بناء نموذج عن قاعدة المعطيات، وتحميل بعض البيانات الثابتة اللازمة لتشغيل الأنظمة المطلوبة.

4.3 الاختبار والتقييم

يقع عبء هذه المرحلة على مدير قاعدة المعطيات، حيث يقوم فيها باختبار أداء القاعدة وتكامل معطياتها ويحدد قيود الأمان وطرق الولوج للقاعدة.

تبدأ هذه المرحلة على التوازي مع برمجة التطبيقات.

قد يتم في هذه المرحلة التعديل على البنية الفيزيائية للقاعدة في حال ظهور ضعف في الأداء، وقد تفرض هذه المرحلة تعديلات على البنية المنطقية للقاعدة أو تغيير نظام إدارة قواعد البيانات المستخدم.

5.3 التطبيق

بعد أن تتجاوز قاعدة المعطيات مرحلة الاختبار، يمكن اعتبارها جاهزة للتطبيق، وأن هذا الجزء من النظام المعلوماتي قد انتهى، وينصب التركيز بدءاً من هذه المرحلة على تطوير التطبيقات المطلوبة.

6.3 الصيانة والتطوير

ينبغي على مدير قاعدة المعطيات أن يكون مستعداً للقيام بالعديد من إجراءات الصيانة على قاعدة المعطيات، ويمكن أن تتضمن أعمال الصيانة النقاط التالية:

- أعمال صيانة وقائية (نسخ احتياطي)
- أعمال صيانة علاجية (تصحيح بعض الأخطاء)
- أعمال صيانة تطويرية (تحسينات في الأداء أو إضافة كيانات أو واصفات جديدة)
- تعديل في سماحيات الولوج للمستخدمين
- إجراء عمليات إحصائية ومراقبة للمستخدمين وعمليات الولوج، بهدف تحسين الأداء
- إجراء تدقيقات أمنية دورية اعتماداً على المعلومات الإحصائية التي تم جمعها
- إجراء عمليات وحسابات دورية للنظام بالكامل اعتماداً على معايير محددة

4. دورة حياة نظام المعلومات ودورة حياة قاعدة المعطيات

يوجد العديد من الأنشطة التي يتم تنفيذها على التوازي ما بين دورة حياة نظام المعلومات ودورة حياة قاعدة المعطيات، بحيث تمثل دورة حياة نظام المعلومات الإطار الخارجي الذي تتم فيه أنشطة دورة حياة قاعدة المعطيات.

الفصل الرابع:

مخطط الكيانات-العلاقات (ERD) (1)

العنوان	الصفحة
1. مقدمة	3
2. الكيان ومجموعات الكيانات (Entities and entity sets)	3
3. الواصفات وأنواعها (Attributes)	4
1.3 أنواع الواصفات	4
4. العلاقات ومجموعات العلاقات (Relationships and Relationship sets)	5
5. طرق ترميز الكيانات والعلاقات والواصفات في مخططات ER	6
6. المفاتيح (Keys)	9
7. طرق الترميز الأخرى	10
8. المراجع	11

الكلمات المفتاحية

علاقة، كيان، واصفة، كيان ضعيف، كيان حقيقي، واصفة متعددة القيم، واصفة مشتقة، واصفة مركبة، مفتاح، درجة العلاقة، CHEN,CROW'S FOOT, REIN85, IDEFIX.

ملخص

يركز هذا الفصل على تعريف المصطلحات وطرق الترميز في مخططات علاقة–كيان (ERD).

الأهداف التعليمية

يهدف هذا الفصل التعريف بالمفاهيم التالية:

- الكيان ومجموعات الكيانات (Entities and entity sets)
- الواصفات وأنواعها (Attributes)
- العلاقات ومجموعات العلاقات (Relationships and Relationship sets)
- طرق ترميز الكيانات والعلاقات والواصفات في مخططات ER
- المفاتيح (Keys)
- طرق ترميز أخرى

1. مقدمة

مخططات ERD هي طريقة لتوصيف مشكلة في العالم الحقيقي، بطريقة يمكن شرحها وفهمها من قبل المستثمر النهائي، وهي تنطلق من فكرة أن أي مشكلة يمكن توصيفها بواسطة كيانات، علاقات وواصفات. حيث الكيان هو أي شيئ له وجود حقيقي أو له وجود مفهومي في عقل الشخص (conceptual)، ويمكن تمييزه من خلال واصفات تعبر عنه وتميزه عن غيره ويرتبط مع غيره من الكيانات بمجموعة علاقات، فمثلاً الموظف له وجود حقيقي ويمكن تمييزه من خلال اسم وتاريخ ميلاد وتاريخ توظيف وغيرها، ويرتبط مع قسم في شركة بعلاقة انتماء فكل موظف ينتمي لأحد أقسام الشركة.

عناصر ERD هي ثلاثة: الكيانات، العلاقات، الواصفات، بالإضافة إلى مجموعة ترميزات تساعد على التفريق بين أنواع الكيانات والعلاقات، وفيما يلي شرحها.

2. الكيان ومجموعات الكيانات (Entities and entity sets)

يمكن النظر إلى قاعدة البيانات العلائقية على أنها:

- مجموعة من الكيانات
- روابط (علاقات) بين هذه الكيانات.

الكيان (Entity) هو غرض موجود ومميز عن غيره من خلال قيم لمجموعة من الواصفات (شخص، شركة، كوكب، منتج،..)
يمكن أن يعبر الكيان عن شيء واقعي (شخص، كتاب ..)، أو عن شيء مجرد (عطلة، قرار ...).

مجموعة الكيانات (Entity Set) هي مجموعة تضم كيانات متشابهة من حيث الواصفات (من نمط واحد). (الأشخاص، الشركات، الكواكب، المنتجات،...)
يرمز لمجموعة الكيانات في مخططات ER بمستطيل.

يمكن التميز بين نوعين من الكيانات:
الكيان الحقيقي (Strong Entity set)، وهو الكيان الذي له وجود بمعزل عن وجود الكيانات الأخرى.
الكيان الضعيف (Weak Entity set) يعتمد وجوده على وجود كيان آخر، كقرار ترفيع موظف (لا وجود له إذا لم يكن هناك كيان للموظف)، يرمز لهذا النوع من الكيانات بمستطيل مزدوج الإطار.
أهم ميزات الكيان الضعيف، عدم وجود مفتاح أساسي طبيعي (Actual Key)، وعادةً يتم انتخاب مفتاح أساسي له من تركيب المفتاح المستورد من الكيان الأب مع إحدى واصفات هذا الكيان (رقم الموظف وتاريخ الترفيع مثلاً)

3. الواصفات وأنواعها (Attributes)

تتميز كل مجموعة كيانات بمجموعة من الواصفات، ويتميز كيان عن غيره ضمن المجموعة نفسها بقيم الواصفات.

أمثلة عن مجموعات كيانات:

- Customer (name, ssn, city, street)
- Account (number, Balance)

أمثلة عن كيانات:

- (جمال، 6103، دمشق، مزة) هو كيان من مجموعة الكيانات Customer
- (محمد، 2130، دمشق، برامكة) هو كيان آخر من المجموعة Customer

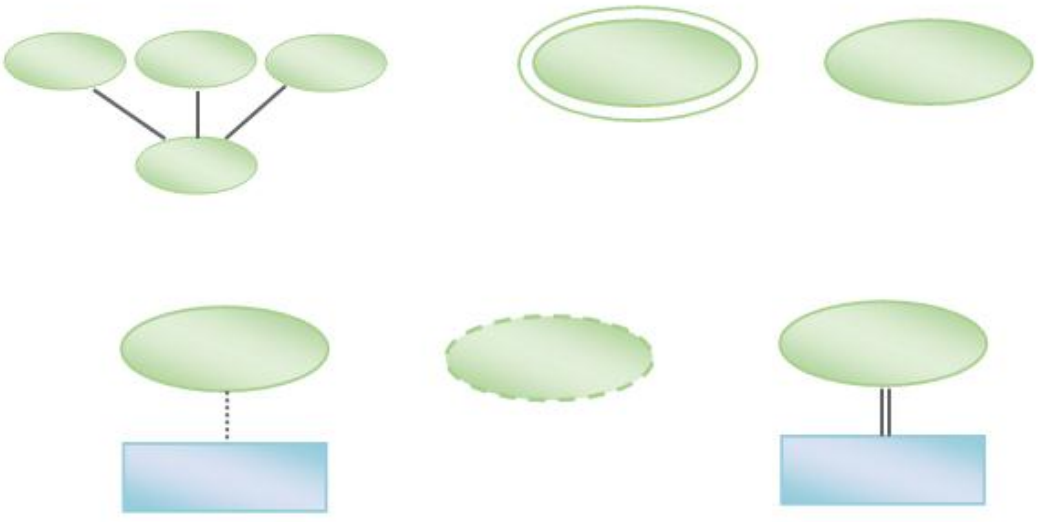
مجال التعريف (Domain): القيم المتاحة لكل واصفة.

مثال: القيم المتاحة للواصفة city هي: (دمشق – حلب – حمص – ... – السويداء)

أنواع الواصفات

- (بسيطة، مركبة) الاسم هو واصفة مركبة من ثلاث واصفات بسيطة هي: الاسم الأول، الأب، الكنية
- مشتقة (العمر هو واصفة مشتقة من تاريخ الميلاد)
- (وحيدة القيمة، متعددة القيم) رقم الهاتف هو واصفة متعددة القيم (هاتف منزل، هاتف عمل، خليوي ...)

يرمز للواصفة في مخططات ER بشكل بيضوي، إذا كانت الواصفة متعددة القيم يرمز لها بشكل بيضوي مزدوج الإطار، ويرمز للواصفة المركبة بمجموعة أشكال بيضوية مرتبطة بخطوط، أو بشكل بيضوي يرتبط مع الكيان بخط مزدوج، يرمز للواصفة المشتقة بشكل بيضوي منقط الإطار أو اتصاله مع الكيان يتم بخط منقط.



ملاحظات:

1. الواصفة المركبة تترجم عند الإنتقال إلى قاعدة البيانات، إلى عدة حقول في جدول الكيان، فمثلاً الاسم يترجم في جدول Employee إلى ثلاثة حقول هي (First_name, Father, Last_name).
2. الواصفة المشتقة لا تترجم إلى حقل في قاعدة البيانات، لأنه يمكن استخلاصها من واصفة أو مجموعة واصفات أخرى، فمثلاً العمر يمكن استخلاصه من طرح تاريخ الميلاد من التاريخ الحالي.
3. عند ترجمة مخطط الكيانات إلى قاعدة بيانات، تترجم الواصفة متعددة القيم إلى عدة حقول في جدول الكيان إذا كان العدد الأعظمي لها محدد، أو إلى جدول يرتبط مع الجدول الأساسي، فمثلاً الواصفة رقم الهاتف، يمكن ترجمتها إلى ثلاثة حقول في جدول الموظف (Employee) هي (Home_phone, Work_phone, Mobile)، أو إلى جدول جديد هو Contact (Employee_id, Contact_type, Contact_value).

4. العلاقات ومجموعات العلاقات (Relationships and Relationship sets)

العلاقة هي الرابط بين أكثر من كيان، يمكن أن يكون للعلاقة واصفات تماماً كالكيان، فمثلاً العلاقة بين قرار إجازة وموظف يمكن أن يكون لها تاريخ بدء ومدة وغير ذلك من الواصفات.

درجة العلاقة (Degree of a relationship): تشير إلى عدد الكيانات المشاركة في العلاقة 1، 2، أو أعلى.

أمثلة:

- (يحضر الطالب محاضرة لمدرس في قاعة معينة، فالعلاقة بين المدرس والطالب والقاعة والمحاضرة هي 4).
- (حساب مصرفي، زبون، فرع مصرف) يمكن ربطها بعلاقة ثلاثية.
- يمكن أن تكون العلاقة أحادية مثل (موظف، مدير) وفي هذه الحالة تدعى العلاقة بالعلاقة العودية (recursive).

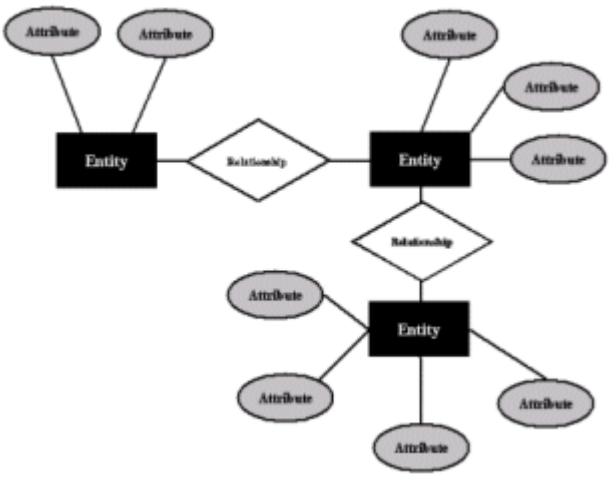
يمكن أن تكون العلاقة من أحد ثلاثة أنماط:

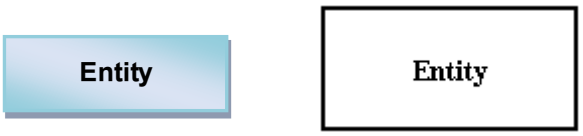
- واحد لواحد (One to one): وتدل على أن كل كيان من الطرف الأول يرتبط بكيان وحيد من الطرف الثاني والعكس بالعكس، فمثلاً لكل زوج زوجة ولكل زوجة زوج وحيد.
- واحد لعدة (One to many): وتدل على أن الكيان من الطرف الأول يمكن أن يرتبط بأكثر من كيان من الطرف الثاني، بينما كل كيان من الطرف الثاني يرتبط بكيان وحيد من الطرف الأول، فمثلاً يضم القسم أكثر من موظف، بينما يعمل الموظف في قسم واحد من الشركة.
- عدة لعدة (many to many): وتدل على أن الكيان من أحد الطرفين يرتبط بأكثر من كيان من الطرف الآخر، فمثلاً كل فيلم يضم عدة ممثلين، وكل ممثل يمثل في أكثر من فيلم.

ملاحظات:

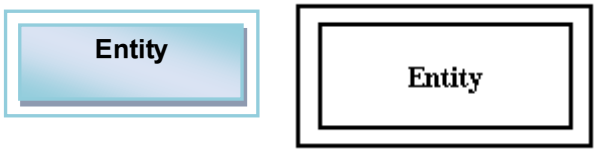
1. تترجم العلاقة واحد لواحد عند الانتقال إلى تصميم قاعدة البيانات بدمج الكيانين في كيان واحد يجمع واصفات الكيانين، إلا في حالات خاصة بداعي رفع الأداء عند استرجاع البيانات.
2. تترجم العلاقة واحد لعدة، بتصدير مفتاح الأب للإبن، فمثلاً علاقة القسم بالموظف تترجم بتصدير مفتاح القسم، كواصفة للموظف.
3. تترجم العلاقة عدة لعدة، بإنشاء كيان جديد تصدر له مفاتيح الكيانين المرتبطين، فمثلاً العلاقة بين الفيلم والممثل، تترجم بإنشاء كيان جديد (Cast) يضم مفتاح الفيلم والممثل (movie_id, actor_id).

5. طرق ترميز الكيانات والعلاقات والواصفات في مخططات ER





Entity: غرض نريد تخزين بيانات عنه



Weak Entity: وجوده يعتمد على وجود كيان آخر

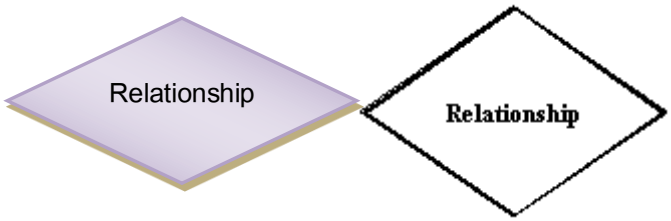
Attributes: خصائص أو مميزات الكيان أو العلاقة

Key attribute: خاصة مميزة لكل كيان عن غيره ضمن مجموعة الكيانات نفسها (مثل رقم هوية الشخص)

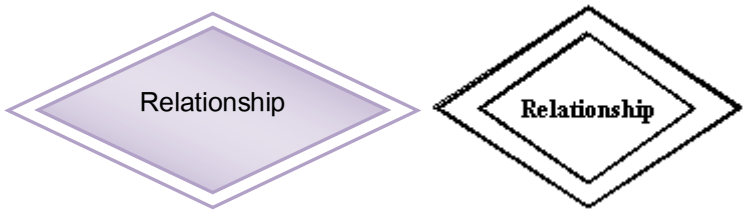
Multivalued attribute: واصفة للكيان يمكن أن يكون لها أكثر من قيمة (رقم الهاتف)

Derived attribute: واصفة يمكن استنتاج قيمتها بالإعتماد على قيم واصفات أخرى (العمر انطلاقاً من تاريخ الميلاد)

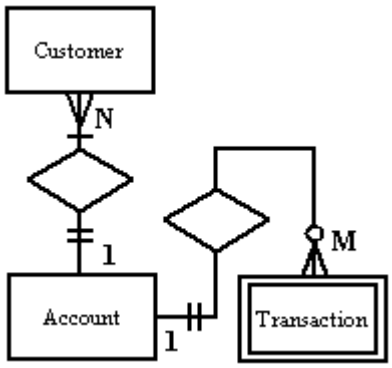
Composite attribute: واصفة مركبة يمكن تجزيئها لأكثر من واصفة (الاسم=اسم أول + اسم أب + كنية)



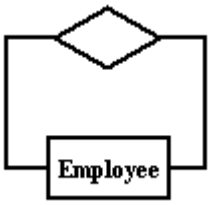
Relationships: علاقة بين أكثر من كيان



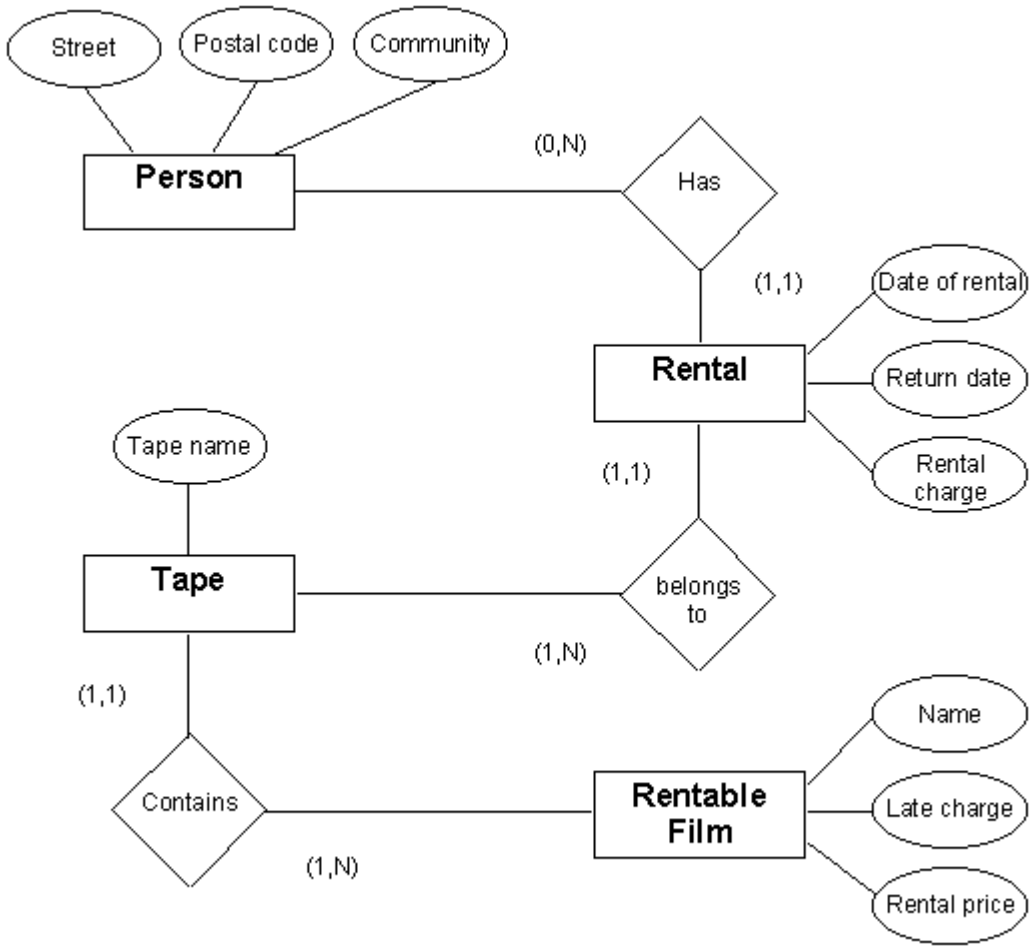
Weak relationship: علاقة بين كيان ضعيف وكيانات أخرى



Cardinality: تحدد عدد الكيانات المرتبطة من مجموعة كيانات مع كيان أو أكثر من مجموعة كيانات أخرى



Recursive relationship: علاقة أحادية بين كيان وكيانات أخرى من نفس مجموعة الكيانات



6. المفاتيح (Keys)

هي واصفة أو مجموعة واصفات تميز كيان من مجموعة كيانات عن غيره من أعضاء نفس المجموعة، ويمكن التمييز بين ثلاثة أنواع من المفاتيح:

- مفتاح أعظمي (Super key):

هي واصفة أو مجموعة واصفات تميز قيمها بشكل وحيد كل كيان في مجموعة الكيانات (الاسم واسم الأب والكنية للموظف)، ويمكن إيجاد أكثر من مفتاح أعظمي لنفس الكيان.

- مفتاح مرشح (Candidate key):

هي المفاتيح الأعظمية الأصغر في الكيان (رقم الضمان الاجتماعي، الرقم الذاتي، رقم الهوية)، يمكن أن يكون لنفس الكيان أكثر من مفتاح مرشح، يتم انتقاء أحدها ليكون المفتاح الأساسي للكيان.

- مفتاح أساسي (Primary key):

أحد المفاتيح المرشحة يتم اختياره ليكوّن مفتاح أساسي ويرمز له في ER بخط تحت الواصفة.

7. طرق الترميز الأخرى

هناك عدة نماذج لمخططات ER تختلف عن بعضها بمستوى التفصيلات الممثّلة بالمخطط، وترميز عناصر المخطط، إلا أن أشهر النماذج وأكثرها استخداماً هو نموذج CHEN الذي عرضنا ترميزاته ومحتوياته في هذا الفصل.

من النماذج الأخرى:

- CHEN
- CROW'S FOOT
- REIN85
- IDEFIX

8. المراجع

- http://database.ittoolbox.com
- http://www.utexas.edu/its/windows/database/datamodeling/Rm/

الفصل الخامس:

مخطط الكيانات-العلاقات (ERD) (2)

العنوان	الصفحة
1. مقدمة	3
2. التعميم (Generalization)	5
3. ربط العلاقات (Aggregation)	6
4. خطوات بناء نموذج ERD	7
5. تطبيق	8
1.5 الخطوة الأولى: عزل الكيانات	8
2.5 الخطوة الثانية: إيجاد العلاقات	9
3.5 الخطوة الثالثة: رسم ERD مبدئي	9
4.5 الخطوة الرابعة: وضع الحدود الدنيا والعليا لأطراف العلاقات	10
5.5 الخطوة الخامسة: انتخاب المفاتيح الأساسية	10
6.5 الخطوة السادسة: تحسين مخطط ERD	12
7.5 الخطوة السابعة: تحديد الواصفات	12
8.5 الخطوة الثامنة: ربط الواصفات بالكيانات	13
9.5 الخطوة التاسعة: رسم ERD النهائي	14
10.5 الخطوة العاشرة: التحقق من صحة المخطط	14
6. المراجع	15

الكلمات المفتاحية

علاقة، كيان، واصفة، كيان وهمي، كيان حقيقي، واصفة متعددة القيم، واصفة مشتقة، واصفة مركبة، مفتاح، درجة العلاقة، CHEN,CROW'S FOOT, REIN85, IDEFIX.

ملخص

يركز هذا الفصل على تعريف المصطلحات وطرق الترميز في مخططات علاقة–كيان (ERD).

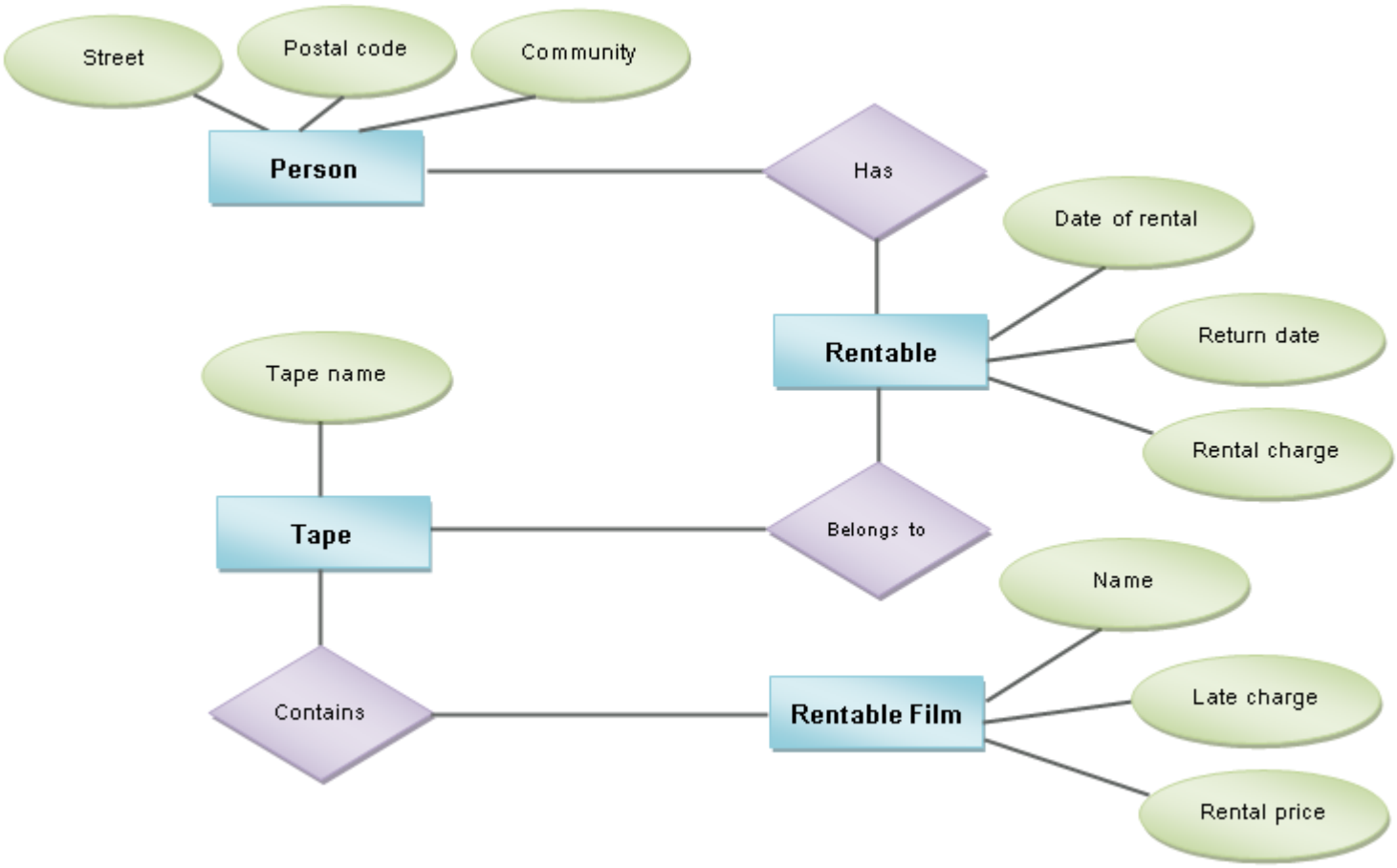
الأهداف التعليمية

يتضمن هذا الفصل تتمة المفاهيم المتعلقة بمخطط الكيانات والارتباطات، الذي بدأناه في الفصل السابق:

- مثال عن ترميز Chen وترميز Crow's Foot.
- التعميم (Generalization)
- ربط العلاقات (Aggregation)
- خطوات بناء نموذج ERD
- تطبيق

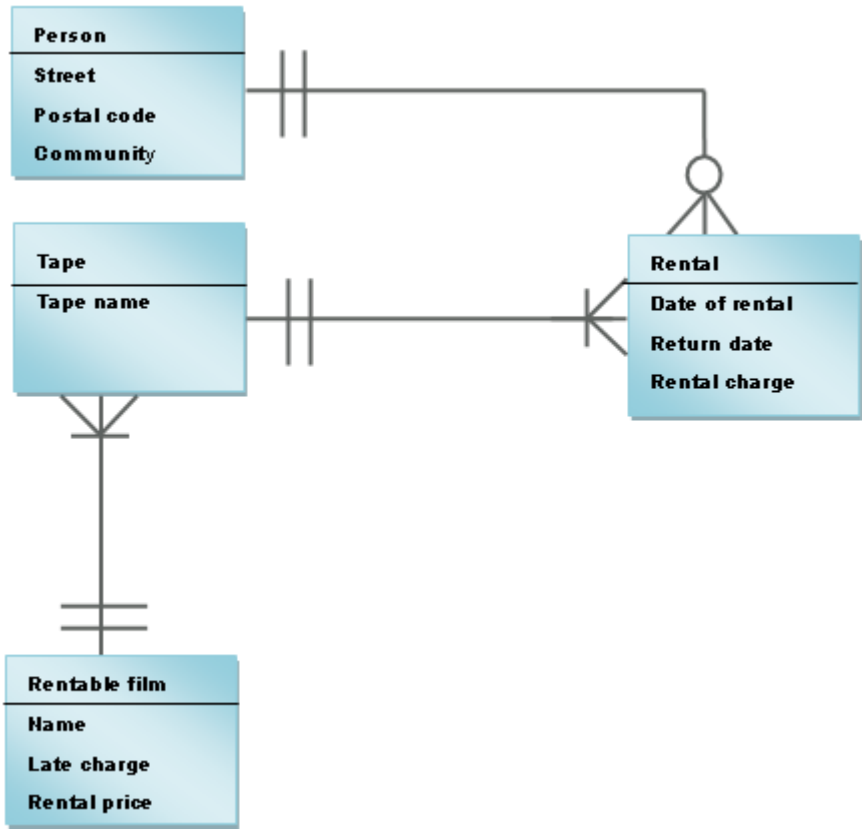
1. مقدمة

عرضنا في الفصل السابق مخططاً يمثل قاعدة بيانات مركز لإعارة أشرطة الفيديو، فيما يلي سنقوم بشرح العلاقات المتضمنة فيه، ومن ثم سنعيد المخطط مستخدمين ترميز Crow's Foot.



A	0	يمكن أن لا يكون للشخص حركات إستعارة (مشترك مسجل لم يقم بحركات إستعارة)
	N	كل شخص يمكن أن يكون له أكثر من حركة استعارة.
B	1	حركة الإعارة تخص شخص على الأقل.
	1	حركة الإعارة تخص شخص على الأكثر.
C	1	حركة الإعارة تخص شريط على الأكثر
	1	حركة الإعارة تخص شريط على الأكثر.
D	1	الشريط يجب أن تتم عليه حركة إعارة واحدة على الأقل (يتم شراؤه عند طلبه).
	N	يمكن أن تتم على الشريط أكثر من حركة إعارة.
E	1	الشريط يتضمن فيلم على الأقل.
	1	الشريط يتضمن فيلم على الأكثر (لا يمكن تسجيل أكثر من فيلم على نفس الشريط).
F	1	الفيلم يسجل على شريط واحد على الأقل.
	N	يمكن أن يسجل الفيلم على أكثر من شريط (مدة الفيلم طويلة).

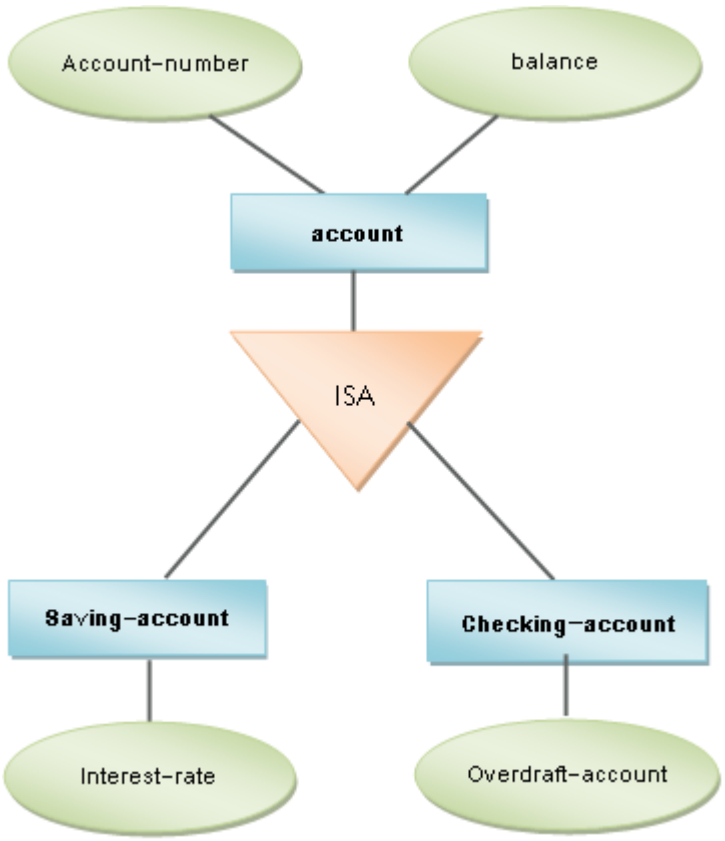
الشكل التالي يمثل المخطط السابق باستخدام ترميز Crow's Foot.



2. التعميم (Generalization)

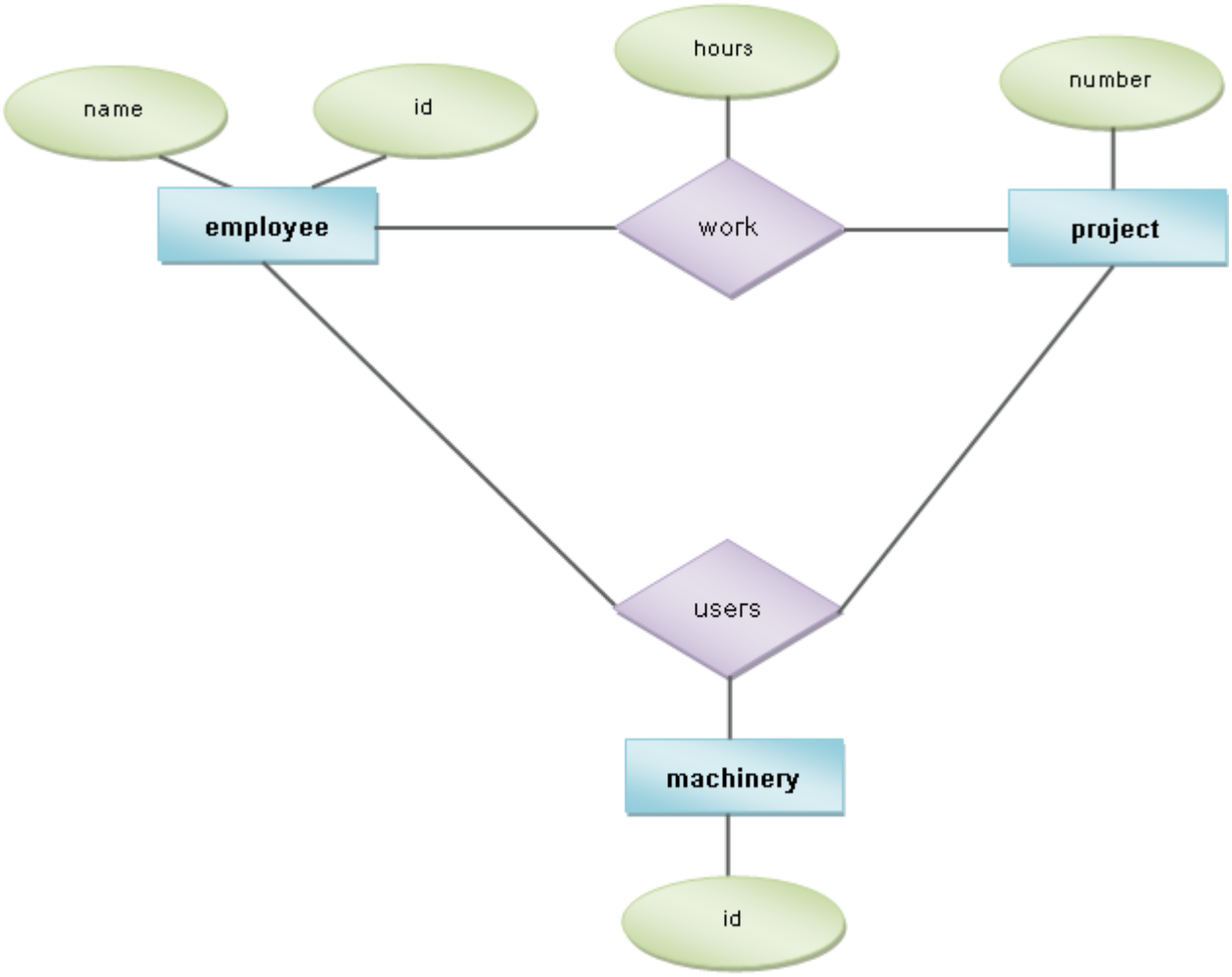
يمكن فهم التعميم من خلال المثال التالي:

يمكن التمييز بين نوعين من الحسابات المصرفية، حساب جاري وحساب ادخار، لكل منهما ميزاته الخاصة، إلا أنهما يتقاطعان في مجموعة واصفات تمثل الحساب المصرفي بصرف النظر عن نوعه. يمكن من خلال التعميم التعبير عن فكرة توريث الواصفات، فلكل من نوعي الحسابات في مثالنا مجموعة واصفات تخصه (مقدار الفائدة أو مقدار الرصيد السالب المسموح) بالإضافة إلى واصفات الحساب بشكل عام (رقم الحساب ورصيده).



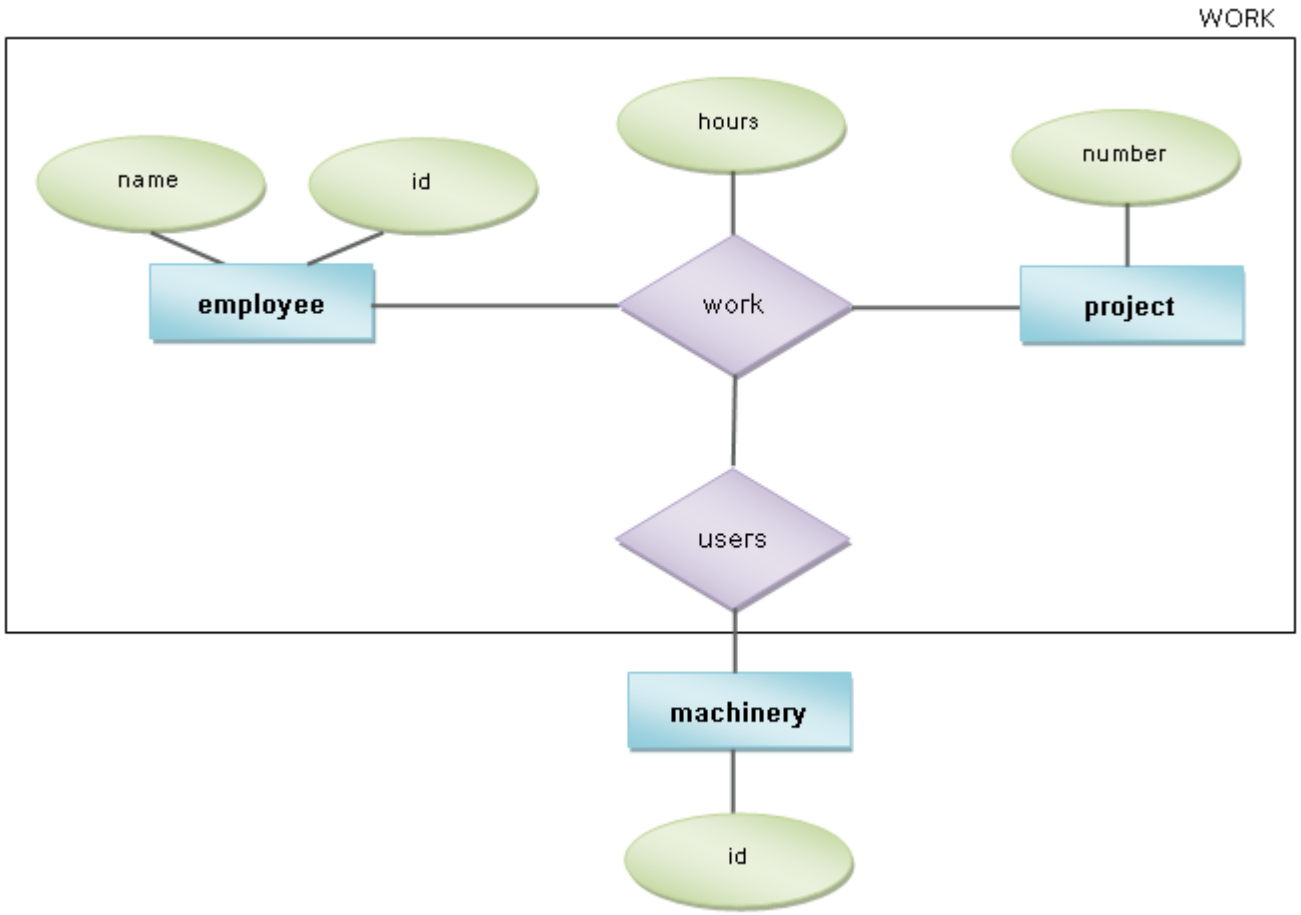
3. ربط العلاقات (Aggregation):

يعبر مفهوم العلاقة في مخططات ER عن ارتباط بين أكثر من كيان، في مخططات ER الموسّعة يمكن أن تربط العلاقة مع علاقات أخرى، وقد نحتاج لهذه الميزة في بعض الأحيان كما سنرى في المثال التالي:



يعبر المخطط التالي عن قاعدة بيانات لموظفين يعملون في مشاريع ويستخدمون آلات لإنجاز العمل.
مجموعة العلاقات work و uses يمكن دمجهما في مجموعة واحدة، إلا أن ذلك يجعل من الصعب قراءة العلاقة بشكل صحيح (قد يتطلب العمل أكثر من آلة).

الحل البديل هو اعتبار العلاقة work كيان وربطها مع machinery بعلاقة uses كما يوضح المخطط التالي:



4. خطوات بناء نموذج ERD

تعتبر المنهجية التالية مجموعة خطوات بسيطة يمكن اتباعها لبناء النموذج المنطقي انطلاقاً من توصيف المسالة، تتألف المنهجية من عشر خطوات هي بالترتيب:

- **عزل الكيانات**

تحديد الكيانات المحتملة، وهي أية أشياء ملموسة أو أحداث أو أدوار أو حتى أشياء منطقية، يتم تخزين بيانات عنها (واصفات)، في النظام المطلوب.

- **إيجاد العلاقات بينها**

ايجاد العلاقة الطبيعية بين كل زوج من الكيانات المعزولة في الخطوة السابقة.

- **رسم مخطط ERD مبدئي**

رسم الكيانات في مستطيلات والعلاقات بينها بخطوط (فقط بين الكيانات المرتبطة بعلاقة).

- **وضع الحدود الدنيا والعليا لأطراف العلاقات (cardinality)**

لكل علاقة نضع الحد الأدنى والأعلى للكيانات المشاركة في العلاقة من طرفيها.

- **انتخاب المفاتيح الأساسية**

من ضمن المفاتيح المرشحة لكل كيان، نختار المفتاح الأصغري وندل على ذلك بوضع خط تحت الواصفة المشاركة في المفتاح.

- **تحسين مخطط ERD**

نحذف العلاقات عدة لعدة (many-to-many)، بإنشاء كيان وسيط وتحويل كل علاقة منها إلى علاقتين واحد لعدة (one-to-many) بتصدير مفتاح الكيانين إلى الكيان الوسيط.

- **تحديد الواصفات**

تحديد واصفات كل كيان ونوع كل منها.

- **ربط الواصفات بالكيانات**

كل واصفة تخص كيان.

- **رسم مخطط ERD النهائي**

رسم الواصفات في مخطط ERD حسب الترميزات المناسبة لكل منها.

- **التحقق من صحة المخطط**

التحقق من مخطط ERD بتطبيقه على أمثلة من المعطيات الحقيقية.

5. تطبيق

سنقوم بتطبيق الخطوات العشر السابقة على المسألة التالية:

"تضم الشركة عدة أقسام، لكل منها مشرف وفي كل منها موظف على الأقل، يمكن أن يعمل الموظف في أكثر من قسم، وممكن أن يكون الموظف في اجازة وغير مسند له أي مشروع، بينما يجب أن يكون في كل مشروع موظف واحد على الأقل. البيانات المطلوب تخزينها هي أسماء الأقسام، المشاريع، المشرفين والموظفين، بالإضافة إلى أرقام تعريف للموظفين والمشرفين ورموز للمشاريع."

الخطوة الأولى: عزل الكيانات

لعزل الكيانات يجب أولاً البحث عن الكيانات المحتملة، وأبسط طريقة لفعل ذلك هي وضع خط تحت كل اسم أو حدث أو مفهوم يمكن توصيفه من خلال مجموعة واصفات، ومن ثم انتقاء الكيانات الفعلية من الكيانات المرشحة، أي التي سيخزن النظام بيانات عنها لتمييزها عن غيرها من الكيانات في نفس المجموعة.

بتطبيق ذلك على المسألة السابقة نحصل على التالي:

"تضم الشركة عدة أقسام، لكل منها مشرف وفي كل منها موظف على الأقل، يمكن أن يعمل الموظف في أكثر من قسم، وممكن أن يكون الموظف في اجازة وغير مسند له أي مشروع، بينما يجب أن يكون في كل مشروع موظف واحد على الأقل. البيانات المطلوب تخزينها هي أسماء الأقسام، المشاريع، المشرفين والموظفين، بالإضافة إلى أرقام تعريف للموظفين والمشرفين ورموز للمشاريع."

الكيانات المرشحة هي: شركة، قسم، مشرف، موظف، مشروع، إجازة.

- الشركة ليست كيان حقيقي كونها وحيدة ولا داعي لتمييزها عن غيرها من نفس المجموعة (مجموعة الكيانات شركة تتضمن ورود وحيد single occurrence)
- الإجازة ليست كيان لعدم وجود واصفات لها.
- الكيانات الفعلية هي: قسم، مشرف، موظف، مشروع.

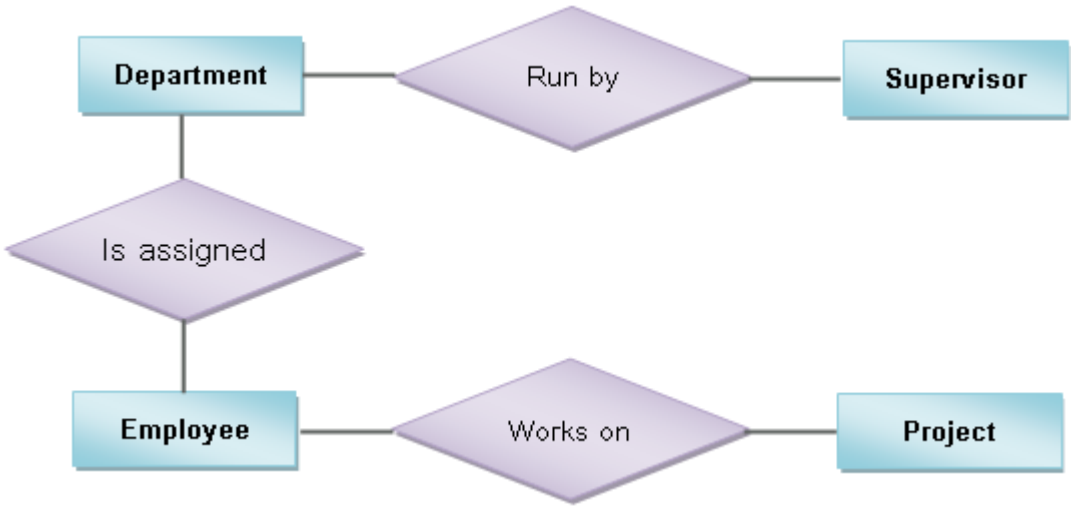
الخطوة الثانية: إيجاد العلاقات

لإيجاد العلاقة بين أزواج الكيانات، نبدأ برسم مصفوفة العلاقات:

	Department	Employee	Supervisor	Project
Department		IS assigned	Run by	
Employee	Belongs to			Works on
Supervisor	Runs			
Project		Uses		

العبارات الموضوعة في خلايا المصفوفة هي وصف للعلاقة بين الكيانين في الصف والعمود الموافقين.

الخطوة الثالثة: رسم ERD مبدئي



الخطوة الرابعة: وضع الحدود الدنيا والعليا لأطراف العلاقات

حدود العلاقة هي عدد الكيانات الممكن أن تشارك في العلاقة من أحد الأطراف مع كيان وحيد من الطرف الآخر، مثلاً كم عدد الموظفين الذين لهم مشرف وحيد، أو كم مشرف يمكن أن يكون لنفس الموظف.

- المشرف (Supervisor):
  - كل قسم له مشرف وحيد.
- القسم (Department):
  - لكل مشرف قسم وحيد.
  - كل موظف ينتمي لقسم أو أكثر.
- الموظف (Employee):
  - كل قسم يضم موظف على الأقل، ويمكن أن يضم أكثر من موظف.
  - كل مشروع يضم موظف على الأقل، ويمكن أن يضم أكثر من موظف.
- المشروع (Project):
  - لكل موظف أكثر من مشروع، ويمكن أن لا يسند له أي مشروع.

وهكذا تكون الأنواع المحتملة لحدود العلاقة هي:

- واحد وواحد فقط

مثل العلاقة بين المشرف والقسم: لكل قسم يوجد مشرف وحيد، ولكل مشرف قسم وحيد.

- واحد أو أكثر

مثل علاقة القسم بالموظف: كل قسم يضم موظف على الأقل، ويمكن أن يضم أكثر من موظف.

- صفر أو أكثر

مثل العلاقة بين المشروع والموظف: لكل موظف أكثر من مشروع، ويمكن ألا يسند له أي مشروع.

- صفر أو واحد على الأكثر

لا يوجد مثال عن هذا النوع في هذه المسألة، ولكنها كأن نقول: يمكن أن يعمل في كل مشروع موظف واحد على الأكثر، ويمكن ألا يسند للمشروع أي موظف.

الخطوة الخامسة: انتخاب المفاتيح الأساسية

المفتاح هو واصفة أو مجموعة واصفات تحدد بشكل وحيد كل كيان من مجموعة كيانات، فمثلاً الاسم هو مفتاح بالنسبة للأقسام لعدم وجود أكثر من قسم في الشركة بنفس الاسم، بينما لا يمكن أن يكون الاسم هو مفتاح للموظف لإمكانية وجود أكثر من موظف باسم John مثلاً.

من المفاتيح المرشحة يجب انتقاء مفتاح أساسي، وهو عادة الأصغر من المفاتيح المرشحة.

في مثالنا يمكن اختيار المفاتيح الأساسية التالية:

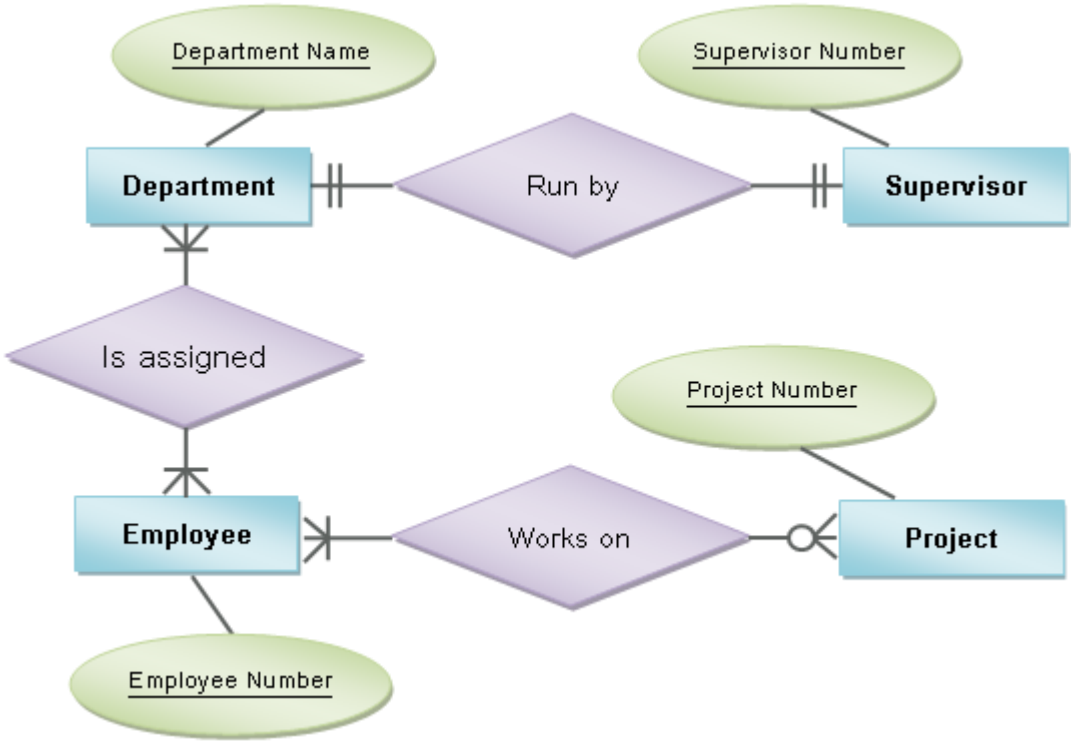
القسم – الاسم

المشرف – الرقم

الموظف – الرقم

المشروع – الرقم

الخطوة السادسة: تحسين مخطط ERD

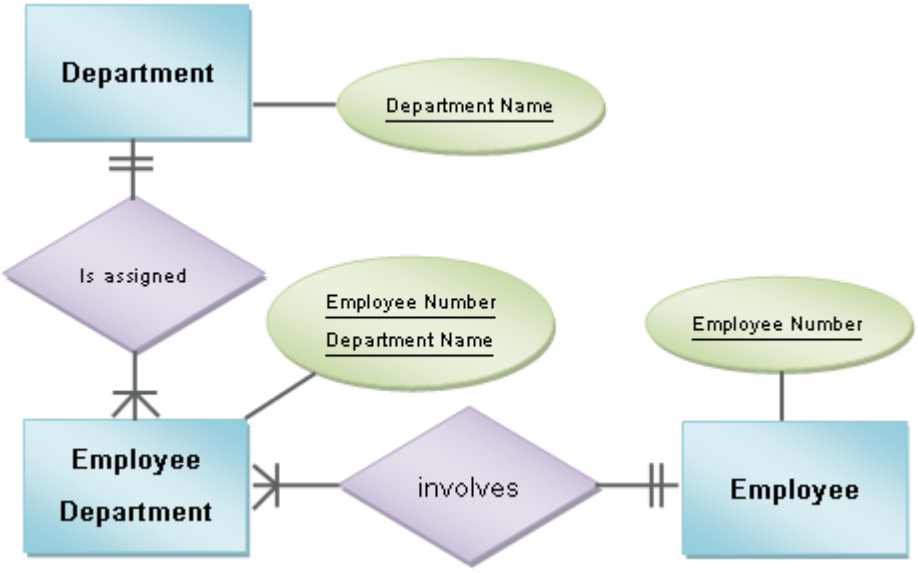


بعد وضع المفاتيح الأساسية للكيانات، يجب كسر العلاقات من نمط عدة-لعدة (many-to-many)، إلى علاقات واحد لعدة (one-to-many)، يتم ذلك بإنشاء كيان وسيط وتصدير مفاتيح الكيانات الأساسية له.

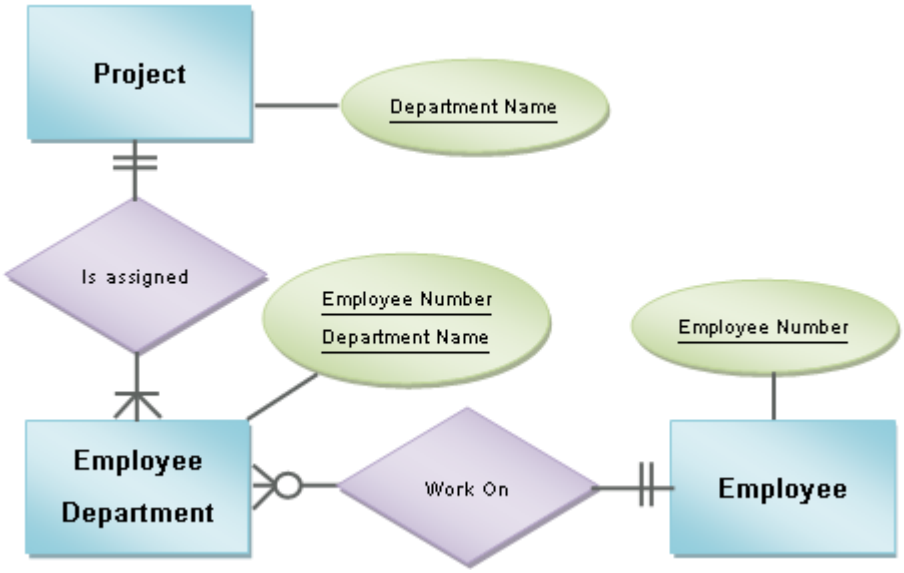
الخطوة السادسة: تحسين مخطط ERD

في حالتنا توجد علاقتين من نمط عدة–لعدة هما Is assigned و works on، فتتحول إلى الشكل التالي:

علاقة القسم بالموظفين:



علاقة المشروع بالموظفين:



الخطوة السابعة: تحديد الواصفات

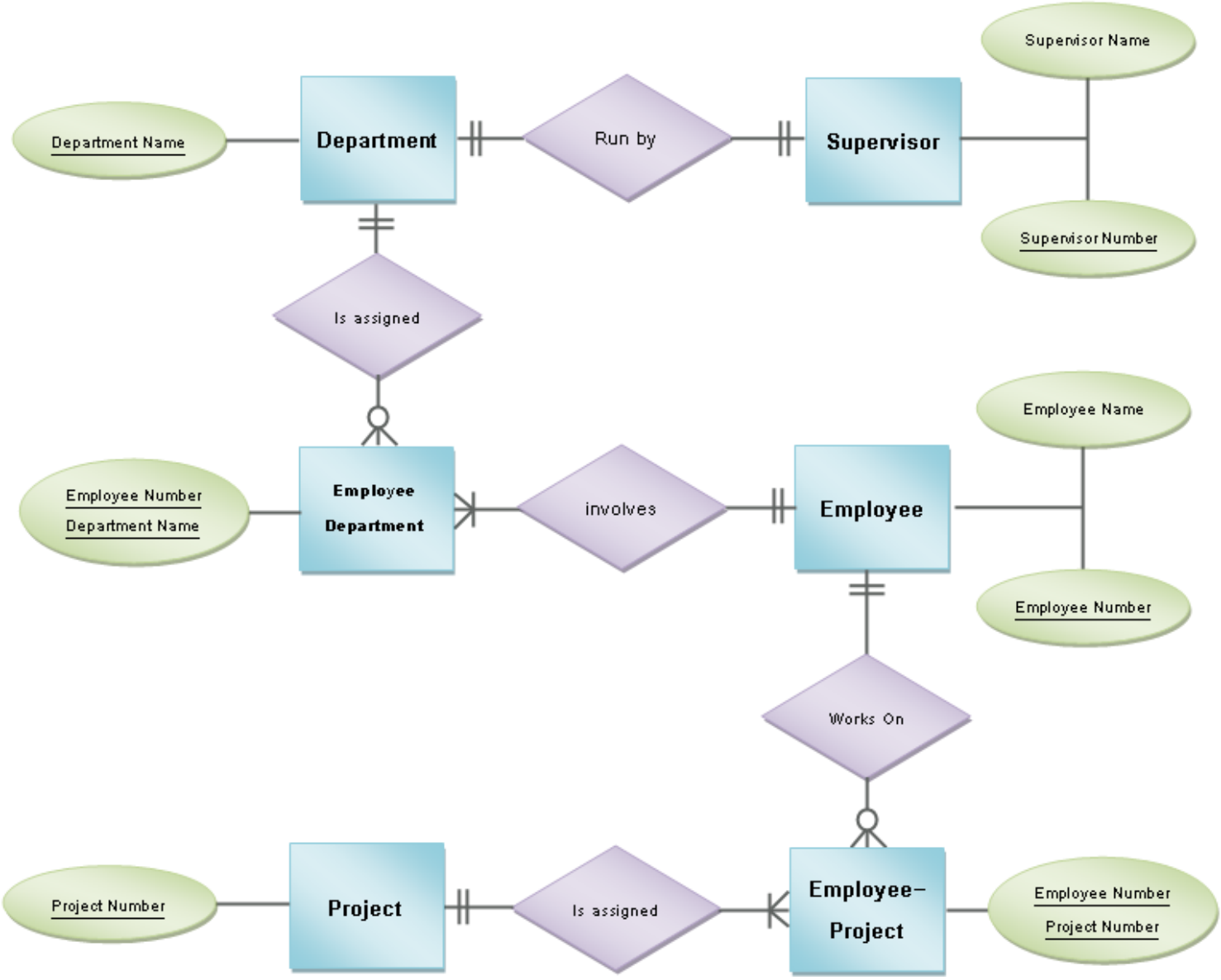
لكل كيان نحدد الواصفات المطلوب تخزينها في النظام، وهي في حالتنا اسم القسم، ولكل من الكيانات المتبقية اسم ورقم.

الخطوة الثامنة: ربط الواصفات بالكيانات

Attribute	Entity
Department name	Department
Employee number	Employee
Employee name	
Supervisor number	Supervisor
Supervisor name	
Project number	Project
Project name	

الخطوة التاسعة: رسم ERD النهائي

نقوم بوضع جميع الواصفات وتحديد المفاتيح على مخطط ERD السابق ليكون نسختنا النهائية من النموذج المنطقي لقاعدة البيانات:



الخطوة العاشرة: التحقق من صحة المخطط

هنا يجب مراجعة مخطط ERD من وجهة نظر المستخدم النهائي للنظام، يتم ذلك بقراءة أجزاء المخطط وشرحها للمستخدم ويجب أن يعبر المخطط عن فهم صحيح لمتطلبات النظام والبيانات التي يجب تخزينها وحفظها عن كل كيان فيه.

6. المراجع

- http://database.ittoolbox.com
- http://www.utexas.edu/its/windows/database/datamodeling/Rm/

الفصل السادس:

التنظيم (Normalization) (1)

العنوان	الصفحة
1. مقدمة	3
2. إشكاليات التصميم السيئ لقاعدة البيانات	3
3. مستويات التنظيم	6
1.3 الشكل النظامي الأول (1NF)	7
2.3 الشكل النظامي الثاني (2NF)	10
3.5 الشكل النظامي الثالث (3NF)	13
4. تدريبات	15

ملخص

يركز هذا الفصل على تعريف إشكاليات التصميم السيئ لقاعدة البيانات، وطرق تنظيم القاعدة.

الأهداف التعليمية

يتضمن هذا الفصل:

- تعريف إشكاليات التصميم السيئ لقاعدة البيانات
  - اشكالية الادخال (Insertion anomaly).
  - اشكالية الحذف (Deletion anomaly).
  - اشكالية التعديل (Update anomaly).
- الشكل النظامي الأول (1NF)
- الشكل النظامي الثاني (2NF)
- الشكل النظامي الثالث (3NF)

1. مقدمة

تعرفنا سابقاً على مستويات التجريد الثلاثة التي يمر بها بناء قاعدة البيانات، وهي المستوى المفهومي والمستوى المنطقي والمستوى الفيزيائي، ورأينا أن المستوى المفهومي ينتهي ببناء مخطط ERD ويتضمن ما يلي:

- عزل الكيانات.
- تحديد الواصفات ومجالات تعريفها
- تحديد العلاقات بين الكيانات
- تحديد المفاتيح المرشحة والمفاتيح الأساسية للكيانات
- إنشاء مخطط ERD
- تدقيق مخطط ERD من قبل مصمم قاعدة البيانات بالتعاون مع المستثمرين

يبدأ المستوى المنطقي في بناء قاعدة المعطيات بعملية التنظيم، ويقصد بالتنظيم (Normalization) تصميم جداول قاعدة البيانات بحيث نتحكم بتكرار المعطيات ونتجنب حالات الشذوذ التي يمكن أن تنتج عن عمليات الإضافة والحذف والتعديل على المعطيات.

سنقوم بداية بدراسة الإشكاليات التي يعاني منها التصميم السيئ لقاعدة البيانات، ومن ثم سندرس مستويات التنظيم وخصائص كل منها.

2. إشكاليات التصميم السيئ لقاعدة البيانات

بفرض أن شركة ما تضم مجموعة من مسؤولي المبيعات يعملون في عدة مواقع، وتريد الإدارة تخزين البيانات المتعلقة بموظفيها وبيانات مستودعاتها في قاعدة معطيات. المحاولة الأولى لبناء القاعدة نتج عنها الجدول التالي:

Id	Name	Address	Title	Store	Store_Address	Store_Phone
Sr1	Jane	E1	Sales Rep	A1	S1	011-1234567
Sr2	Fred	E2	Sales Rep	A1	S1	011-1234567
Sr3	Ed	E3	Manager	A1	S1	011-1234567
Sr4	Ann	E4	Sales Rep	A2	S2	011-7891234
Sr5	Jone	E5	Sales Rep	A2	S2	011-7891234
Sr6	Smith	E6	Manager	A2	S2	011-7891234

يتضمن هذا الجدول[1] بيانات الموظفين وبيانات المستودعات، ومن الواضح وجود تكرار في بيانات المستودعات. إذا أردنا إدخال بيانات موظف جديد فيجب إدخال بيانات موقعه (المستودع) أيضاً، وللحفاظ على عدم تناقض البيانات في القاعدة يجب إدخال بيانات المستودع بدقة وبشكل مطابق للقيم المدخلة سابقاً (لأحد الموظفين السابقين في نفس المستودع)، هنا تظهر مشكلة تكرار البيانات بالإضافة طبعاً لحجم التخزين المهدور.

1. بفرض أننا نريد إدخال بيانات مستودع جديد قبل توظيف أحد فيه، هذا يتطلب إدخال قيم Null في بيانات الموظف ومن ضمنها الحقل المفتاح (Id) مما يخرق شرط تكامل البيانات. هذا ما يدعى بإشكالية الإدخال (Insertion anomaly).

Id	Name	Address	Title	Store	Store_Address	Store_Phone
Sr1	Jane	E 1	Sales Rep	A1	S1	011-1234567
Sr2	Fred	E 2	Sales Rep	A1	S1	011-1234567
Sr3	Ed	E 3	Manager	A1	S1	011-1234567
Sr4	Ann	E 4	Sales Rep	A2	S2	011-7891234
Sr5	Jone	E 5	Sales Rep	A2	S2	011-7891234
Sr6	Smith	E 6	Manager	A2	S2	011-7891234
Null	Null	Null	Null	A3	S3	011-9123456

2. بفرض أننا حذفنا تسجيلة الموظف الأخير في أحد المستودعات، سيؤدي ذلك إلى فقدان بيانات مستودع موجود في الشركة. هذا ما يدعى بإشكالية الحذف (Deletion anomaly).

Id	Name	Address	Title	Store	Store_Address	Store_Phone
Sr1	Jane	E 1	Sales Rep	A1	S1	011-1234567
Sr2	Fred	E 2	Sales Rep	A1	S1	011-1234567
Sr3	Ed	E 3	Manager	A1	S1	011-1234567

⚠ Store A2 data is missing

1 الجدول السابق

3. إذا أردنا تغيير رقم هاتف أحد المستودعات، عندها يجب تغيير رقم الهاتف في كل تسجيلات موظفي ذلك المستودع، فإذا تم تغيير جزء من هذه التسجيلات دون جزء آخر سيؤدي ذلك إلى بيانات متناقضة في قاعدة البيانات، وهذا ما يدعى بإشكالية التعديل (Update anomaly).

Id	Name	Address	Title	Store	Store_Address	Store_Phone
Sr1	Jane	E1	Sales Rep	A1	S1	011-1234567
Sr2	Fred	E2	Sales Rep	A1	S1	011-1234567
Sr3	Ed	E3	Manager	A1	S1	011-1234567
Sr4	Ann	E4	Sales Rep	A2	S2	011-3457890
Sr5	Jone	E5	Sales Rep	A2	S2	011-7891234
Sr6	Smith	E6	Manager	A2	S2	011-7891234

⚠ Conflicted Data

حل المشاكل السابقة يتم من خلال تجزيء الجدول السابق إلى جدولين، أحدهما يحمل بيانات الموظفين ويستورد مفتاح الجدول الثاني الذي يحمل بيانات المستودعات:

Id	Name	Address	Title	Store	Store_Address	Store_Phone
Sr1	Jane	E1	Sales Rep	A1	S1	011-1234567
Sr2	Fred	E2	Sales Rep	A1	S1	011-1234567
Sr3	Ed	E3	Manager	A1	S1	011-1234567
Sr4	Ann	E4	Sales Rep	A2	S2	011-7891234
Sr5	Jone	E5	Sales Rep	A2	S2	011-7891234
Sr6	Smith	E6	Manager	A2	S2	011-7891234

Id	Name	Address	Title	Store
Sr1	Jane	E1	Sales Rep	A1
Sr2	Fred	E2	Sales Rep	A1
Sr3	Ed	E3	Manager	A1
Sr4	Ann	E4	Sales Rep	A2
Sr5	Jone	E5	Sales Rep	A2
Sr6	Smith	E6	Manager	A2

Store	Store_Address	Store_phone
A1	S1	011-1234567
A2	S2	011-7891234

بذلك فإنه:

1. عند إدخال بيانات موظف جديد، يتم تسجيل مفتاح المستودع الذي يعمل فيه، أما بقية بيانات المستودع فهي مخزنة في جدول مستقل وبالتالي لن تظهر بيانات متناقضة في قاعدة البيانات بشأن المستودعات، وإذا أردنا إدخال بيانات مستودع جديد فيمكن إدخالها بصرف النظر عمن يعمل في المستودع، مما يجنبنا إشكالية الإدخال.
2. عند حذف الموظف الأخير في مستودع، تبقى بيانات المستودع في جدول مستقل.
3. عند تعديل رقم هاتف المستودع، يتم التعديل على تسجيلة واحدة في جدول المستودعات.

الإجراءات السابقة (تجزيء الجدول) هي عملية تنظيم قاعدة البيانات، والتي سنتعرف عليها بتفصيل أكثر في هذا الفصل والفصل القادم.

3. مستويات التنظيم

كما سبق وعرفنا، التنظيم هو عملية ترتيب وتوزيع جداول قاعدة المعطيات العلائقية، للتقليل من تكرار المعطيات وتقليص حجم التخزين المطلوب ولحل إشكاليات الإدخال والحذف والتعديل. وغالباً ما ينتج عن عملية التنظيم زيادة في عدد جداول القاعدة.

يتم تنظيم قاعدة البيانات من خلال إخضاعها لمجموعة اختبارات والتعديل على البنية لتحقيق مجموعة معايير.

مستويات التنظيم الأساسية هي ثلاث (الشكل النظامي الأول 1NF، الشكل النظامي الثاني 2NF، الشكل النظامي الثالث 3NF)، ويقصد عادةً بتنظيم القاعدة وضعها في الشكل النظامي الثالث.

يضاف إلى المستويات الثلاثة الشكل المقترح من قبل R.Boyce ويطلق عليه تسمية BCNF، والشكلين النظاميين الرابع والخامس.

سنتناول في هذا الفصل الأشكال النظامية الثلاثة الأساسية، وسنتناول المستويات BCNF, 4NF, 5NF في الفصل اللاحق.

1.3 الشكل النظامي الأول (1NF)

يقال عن جدول في قاعدة البيانات أنه من الشكل النظامي الأول إذا كان تقاطع كل سطر وعمود فيه، يتضمن قيمة وحيدة غير قابلة للتجزئة.

مثال:

يمثل الشكل التالي بيانات شركة تأجير عقارات:

C_id	P_num	C_name	P_address	R_start	R_end	Rent	O_num	Owner
01	Pr3	Jane	A	1-1-96	12-1-98	785	Po23	Jones
	Pr22		B	2-1-98	3-30-00	1200	Po44	Jan
02	Pr17	Fred	C	2-1-88	1-11-90	1000	Po32	Jill
03	Pr32	Ed	D	6-1-90	3-1-95	950	Po32	Jill
	Pr22		B	4-1-00	null	1200	Po44	Jan

C_id[2]	رقم الزبون
P_num	رقم العقار
C_name	اسم الزبون
P_address	عنوان العقار
R_start	تاريخ بدء الإيجار
R_end	تاريخ نهاية الإيجار
Rent	قيمة الإيجار
O_num	رقم مالك العقار
Owner	اسم مالك العقار

2 هو المفتاح الأساسي للجدول

مثال (تتمة):

يمكن ملاحظة أن قيم خلايا الجدول قابلة للتجزئة، وهذا التصميم سيء حيث لا يمكن فيه استخلاص معلومات مالك عقار معين بسهولة. يعاني التصميم أيضاً من مشكلة تكرار المعطيات فكلما تم تأجير عقار يجب أن تدخل بياناته التفصيلية (بيانات العقار) من جديد.

C_id	P_num	C_name	P_address	R_start	R_end	Rent	O_num	Owner
01	Pr3 Pr22	Jane	A B	1-1-96 2-1-98	12-1-98 3-30-00	785 1200	Po23 Po44	Jones Jan
02	Pr17	Fred	C	2-1-88	1-11-90	1000	Po32	Jill
03	Pr32 Pr22	Ed	D B	6-1-90 4-1-00	3-1-95 null	950 1200	Po32 Po44	Jill Jan

مثال (تتمة):

لوضع الجدول السابق في الشكل النظامي الأول يجب تجزئة الخلايا الحاوية على معطيات قابلة للتجزئة، يمكن عمل ذلك من خلال إدخال بيانات كل عملية إيجار في سطر جديد، أي يجب أن يصبح المفتاح الأساسي هو نتيجة تركيب رقم الزبون ورقم العقار (c_id, p_num) علماً أن التصميم مبني على أساس تخزين بيانات العقارات المؤجرة حالياً دون حفظ تاريخ حركات الإيجار، بمعنى أن نفس الزبون لا يمكن أن تتواجد له حركتي إيجار لنفس العقار.

C_id	P_num	C_name	P_address	R_start	R_end	Rent	O_num	Owner
01	Pr3 Pr22	Jane	A B	1-1-96 2-1-98	12-1-98 3-30-00	785 1200	Po23 Po44	Jones Jan
02	Pr17	Fred	C	2-1-88	1-11-90	1000	Po32	Jill
03	Pr32 Pr22	Ed	D B	6-1-90 4-1-00	3-1-95 null	950 1200	Po32 Po44	Jill Jan

C_id	P_num	C_name	P_address	R_start	R_end	Rent	O_num	Owner
01	Pr3	Jane	A	1-1-96	12-1-98	785	Po23	Jones
01	Pr22	Jane	B	2-1-98	3-30-00	1200	Po44	Jan
02	Pr17	Fred	C	2-1-88	1-11-90	1000	Po32	Jill
03	Pr32	Ed	D	6-1-90	3-1-95	950	Po32	Jill
03	Pr22	Ed	B	4-1-00	null	1200	Po44	Jan

تطبيق:

طبق معيار 1NF على الجدول التالي:

id	name	children	birth_date
1001	John Doe	Betty, Frank	2-2-88, 4-3-90
1002	Jane Doe	Betty, Frank	2-2-88, 4-3-90
1003	Freda Fish	Henry, Jane, Jill, Bill	4-4-79, 2-8-84, 7-9-88, 10-3-90
1004	Bill Bass	Hank, April, Ellen	5-4-89, 9-9-94, 7-10-98

الحل:

id	f_name	l_name
1001	John	Doe
1002	Jane	Doe
1003	Freda	Fish
1004	Bill	Bass

id	name	birth_date
1001	Betty	2-2-88
1001	Frank	4-3-90
1002	Betty	2-2-88
1002	Frank	4-3-90
1003	Henry	4-4-79
1003	Jane	2-8-84
1003	Jill	7-9-88
1003	Bill	10-3-90
1004	Hank	5-4-89
1004	April	9-9-94
1004	Ellen	7-10-98

وضع الجدول في الشكل النظامي الأول لا يحل مشاكل التصميم السيئ إنما هو خطوة في طريق حلها، فبالرغم من وضع جدول إيجار العقارات في الشكل النظامي الأول فإنه لا يزال يعاني من مشكلة تكرار البيانات، وإشكاليات الإضافة والتعديل والحذف.

2.3 الشكل النظامي الثاني (2NF)

يقال عن جدول أنه من الشكل النظامي الثاني إذا حقق ما يلي:

- هو من الشكل النظامي الأول.
- كل الواصفات التي لا تشكل جزءاً من المفتاح الأساسي، تعتمد وظيفياً وبشكل كلي على المفتاح الأساسي.

التبعية الوظيفية (functional dependency):

تعتمد الواصفة B على الواصفة A وظيفياً إذا كانت كل قيمة لـ A تقابلها قيمة وحيدة لـ B، فمثلاً رقم الزبون يحدد اسمه وكل قيمة لـ c_id تقابلها قيمة وحيدة لـ c_name لذلك يقال أن c_name تتبع وظيفياً لـ c_id، أو يقال أن c_id تحدد c_name (Determine).

B تعتمد وظيفياً بشكل كلي على مفتاح مركب (fully functional dependency)، إذا كان المفتاح المركب يحدد B، و B لا تعتمد وظيفياً على جزء منه.

تكتب علاقات التبعية بين الواصفات على الشكل التالي:

C_id → c_name

P_num → p_address,owner_num, owner

وضع الجداول في الشكل النظامي الثاني يبدأ بتحديد جميع علاقات التبعية بين الواصفات، ومن ثم تجزيء الجداول (decomposition) بشكل يضمن اعتماد جميع الواصفات التي لا تشكل جزءاً من المفتاح الأساسي كلياً وبشكل مباشر على المفتاح الأساسي.

يمكن الاستنتاج من التعريف السابق أن جميع الجداول ذات المفتاح البسيط (غير المركب) والتي تخضع لمعيار 1NF هي حتماً من الشكل النظامي الثاني.

بتطبيق ذلك على مثالنا نجد العلاقات التالية:

1. C_id + p_num → r_start, r_end.
2. C_id → c_name.
3. P_num → p_address, rent, owner_num, owner.
4. Owner_num → owner.
5. C_id + r_start → p_num, p_address, r_end, r_end, rent, owner_num, owner.
6. P_num + r_start → c_id, c_name, r_end.

بمناقشة هذه العلاقات نجد ما يلي:

1. C_id + p_num → r_start, r_end.

r_start, r_end تعتمد على(c_id, p_num) وبما أن (c_id, p_num) هو المفتاح المركب فذلك لا يناقض مع 2NF.

2. C_id → c_name.

C_name تعتمد جزئياً على المفتاح الأساسي (partial functional dependency)، هذا يتناقض مع 2NF.

3. P_num → p_address, rent, owner_num, owner.

هذه الحالة تشابه الحالة (2).

4. Owner_num → owner.

Owner تعتمد على owner_num و owner_num تعتمد على المفتاح الأساسي، أي أن owner تعتمد على المفتاح الأساسي بالتعدّي (transitive dependency)، وهذا لا يتناقض مع شرط 2NF.

5. C_id + r_start → p_num, p_address, r_end, r_end, rent, owner_num, owner.

c_id + r_start تحدد p_num, p_address, r_end, r_start, rent, owner, owner_num، ولكن (c_id, r_start) هي مفتاح مرشح ممكن أن يكون مفتاح أساسي، وبما أننا لم نقرر بعد (نحن في طور إعادة تصميم قاعدة البيانات) فإن هذه العلاقة لا تخرق شرط 2NF.

6. P_num + r_start → c_id, c_name, r_end.

هذه الحالة تشابه الحالة (5).

يبدأ تطبيق معيار 2NF بتجزيء الجدول إنطلاقاً من علاقة التبعية الكاملة (1):

C_id + p_num → r_start, r_end

C_id	P_num	C_name	P_address	R_start	R_end	Rent	O_num	Owner
01	Pr3	Jane	A	1-1-96	12-1-98	785	Po23	Jones
01	Pr22	Jane	B	2-1-98	3-30-00	1200	Po44	Jan
02	Pr17	Fred	C	2-1-88	1-11-90	1000	Po32	Jill
03	Pr32	Ed	D	6-1-90	3-1-95	950	Po32	Jill
03	Pr22	Ed	B	4-1-00	null	1200	Po44	Jan

Rental

C_id	P_num	R_start	R_end
01	Pr3	1-1-96	12-1-98
01	Pr22	2-1-98	3-30-00
02	Pr17	2-1-88	1-11-90
03	Pr32	6-1-90	3-1-95
03	Pr22	4-1-00	Null

Customer

C_id	C_name
01	Jane
02	Fred
03	Ed

Property

P_num	P_address	rent	O_num	Owner
Pr3	A	785	Po23	Jones
Pr22	B	1200	Po44	Jan
Pr17	C	1000	Po32	Jill
Pr32	D	950	Po32	Jill

تحل جداول الشكل النظامي الثاني إشكاليات الحذف والإضافة ولا تحل إشكالية التعديل، فمثلاً إذا أردنا تعديل بيانات أحد المالكين فيجب تعديل مجموعة من التسجيلات مما يتفق مع إشكالية التعديل سابقة الذكر.

3.3.الشكل النظامي الثالث (3NF)

يكون الجدول من الشكل النظامي الثالث إذا حقق ما يلي:

- من الشكل النظامي الثاني.
- لا توجد فيه واصفات لا تشكل جزءاً من المفتاح الأساسي وتعتمد بالتعدّي على المفتاح الأساسي.

لا تحوي الجداول Rental و Customer واصفات تعتمد بالتعدي على المفتاح الأساسي، بينما تعتمد الواصفة owner في الجدول Property على owner_num الذي يعتمد بدوره على P_num ولوضع هذا الجدول في الشكل النظامي الثالث يجب تجزيئه إلى جدولين كما يلي:

Property

P_num	P_address	rent	O_num	Owner
Pr3	A	785	Po23	Jones
Pr22	B	1200	Po44	Jan
Pr17	C	1000	Po32	Jill
Pr32	D	950	Po32	Jill

Property

P_num	P_address	rent	O_num
Pr3	A	785	Po23
Pr22	B	1200	Po44
Pr17	C	1000	Po32
Pr32	D	950	Po32

Owner

O_num	Owner
Po23	Jones
Po44	Jan
Po32	Jill

الشكل النظامي الثالث يحل إشكاليات التعديل والحذف والإضافة، بالإضافة إلى حل مشكلة تكرار البيانات، وعندما يطلب تنظيم قاعدة بيانات فيقصد بذلك عادةً وضعها في الشكل النظامي الثالث.

4. تدريبات

- في الشكل النظامي الأول، تقاطع السطر مع العمود هو قيمة غير قابلة للتجزيء.

1. صح
2. خطأ

الإجابة: صح

- يمكن الإنتقال إلى الشكل النظامي الثالث من الشكل الأول دون المرور بالشكل النظامي الثاني.

1. صح
2. خطأ

الإجابة: خطأ

- في حال كان المفتاح الأساسي للجدول بسيط (غير مركب)، فلا فرق بين التبعية الوظيفية والتبعية الوظيفية الكلية.

1. صح
2. خطأ

الإجابة: صح

- أي من العبارات التالية تصف الشكل النظامي الثالث (اختر 2 من الإجابات)

1. هو حتماً من الشكل النظامي الأول والثاني.
2. المفتاح الأساسي للجدول هو حتماً مفتاح بسيط (غير مركب).
3. لا توجد في الجدول علاقات تبعية بالتعدي.
4. الشكل النظامي الثالث يحل اشكالية الإدخال والحذف ولا يحل اشكالية التعديل.

الإجابة: (1-3)

- في حال الشكل النظامي الأول، كل خلية هي وحيدة القيمة (لا يمكن أن يكون محتوى الخلية هو مصفوفة من القيم).

1. صح
2. خطأ

الإجابة: صح

- في حال الشكل النظامي الأول، المدخلات في عمود هي من نفس النمط حتماً.

1. صح
2. خطأ

الإجابة: صح

الفصل السابع:

التنظيم (Normalization) (2)

العنوان	الصفحة
1. مقدمة	3
2. الشكل النظامي Boycee–Codd (BCNF)	3
3. الشكل النظامي الرابع (4NF)	7
4. الشكل النظامي الخامس (5NF)	9

ملخص

يستكمل هذا الفصل أشكال تنظيم قاعدة البيانات، التي بدءنا بدراستها في الدرس السابق.

الأهداف التعليمية

يتضمن هذا الفصل:

- الشكل النظامي BCNF (Boycee–Codd Normal form)
- الشكل النظامي الرابع (4NF)
- الشكل النظامي الخامس (5NF)

1. مقدمة

رأينا سابقاً أنه عندما يطلب تنظيم قاعدة البيانات، فالمقصود بذلك عادةً وضعها في الشكل النظامي الثالث (3NF)، ونادراً ما تحقق القاعدة شروط 3NF وتخرق الأشكال النظامية التالية (4Nf, 5NF, BCNF)، يضاف إلى ذلك أن الفائدة من تحويل القاعدة من 3NF إلى 4NF أو 5NF بسيطة ولا تقابل الجهد اللازم لهذا التحويل.

2. الشكل النظامي Boycee-Codd (BCNF)

يطبق الشكل النظامي BCNF على الجداول الحاوية على:

- عدة مفاتيح مرشحة
- مفاتيح مرشحة مركبة
- مفاتيح مرشحة متقاطعة

يكون الجدول من الشكل BCNF إذا كان كل محدد فيه هو مفتاح مرشح.
الواصفة (أو مجموعة الواصفات) المحددة هي واصفة تعتمد عليها واصفات أخرى بشكل كلي.
يحدث خرق شرط BCNF عادةً في الجداول الحاوية على أكثر من مفتاح مرشح مركب وفي حالات تقاطع المفاتيح المرشحة في الجدول.

مثال1:

ليكن لدينا الجداول التالية:

Rental

C_id	P_num	R_start	R_end
01	Pr3	1-1-96	12-1-98
01	Pr22	2-1-98	3-30-00
02	Pr17	2-1-88	1-11-90
03	Pr32	6-1-90	3-1-95
03	Pr22	4-1-00	Null

Customer

C_id	C_name
01	Jane
02	Fred
03	Ed

Property

P_num	P_address	rent	O_num
Pr3	A	785	Po23
Pr22	B	1200	Po44
Pr17	C	1000	Po32
Pr32	D	950	Po32

Owner

O_num	Owner
Po23	Jones
Po44	Jan
Po32	Jill

يتضمن كل من الجداول Customer, Owner, Property مفتاح بسيط تعتمد عليه بقية الحقول، فهي حتماً من الشكل BCNF.

نجد في الجدول Rental علاقات التبعية التالية:

1. C_id + p_num → r_start, r_end.
2. C_id + r_start → p_num, r_end
3. P_num + r_start → c_id, r_end.

(افترضنا في هذا الشكل أن المستأجر لا يستأجر شقة مرتين بنفس التاريخ، والشقة لا يمكن تأجيرها أكثر من مرة بنفس اليوم، وإلا فإن تصميم القاعدة سيختلف).

المحددات (C_id + p_num)، (C_id + r_start)، (P_num + r_start) كلها مفاتيح مرشحة، بالتالي فإن الجدول Rental هو من الشكل BCNF[1].

مثال2:

لننظر إلى الجدول التالي:

Client_Meeting

C_id	I_date	I_time	Room_num	Emp_num
08	04-20-00	09:30	P70	Pe-23
17	05-01-00	09:30	P60	Pe-32
20	05-01-00	10:30	P70	Pe-23

يتضمن هذا الجدول بيانات لقاءات موظفي التسويق مع الزبائن، (C_id, I_date) هو المفتاح الأساسي، ويفترض هذا الجدول أن الزبون لا يمكنه إجراء أكثر من مقابلة في نفس اليوم، بينما يستطيع الموظف إجراء أكثر من مقابلة في اليوم، وفي غرفة واحدة.

علاقات التبعية في الجدول السابق تتضمن:

1. C_id + I_date → I_time, room_num, Emp_num.
2. I_date + I_time + Emp_num → C_id.
3. I_date + I_time + Room_num → C_id, Emp_num.
4. I_date + Emp_num → Room_num.

1, 2, 3 لا تشكل خرقاً لـ BCNF لأن المحدد فيها هو مفتاح مرشح.

في **4** نجد أن الطرف اليساري لا يشكل مفتاح مرشح (يمكن للموظف إجراء أكثر من مقابلة في نفس اليوم)، هذه العلاقة لا تخرق شرط 3NF لأن الطرف اليميني هو جزء من مفتاح مرشح (**3**) ولكنها تخرق شرط BCNF.

لوضع الجدول في الشكل BCNF يجب تجزئته على الشكل:

C_id	I_date	I_time	Room_num	Emp_num
08	04-20-00	09:30	P70	Pe-23
17	05-01-00	09:30	P60	Pe-32
20	05-01-00	10:30	P70	Pe-23

1 يكون الجدول من الشكل BCNF إذا كان كل محدد فيه هو مفتاح مرشح.

C_id	I_date	I_time	Emp_num
08	04--20-00	09:30	Pe-23
17	05-01-00	09:30	Pe-32
20	05-01-00	10:30	Pe-23

Emp_num	I_date	Room_num
Pe-23	04--20-00	P70
Pe-32	05-01-00	P60
Pe-23	05-01-00	P70

الشكل النظامي الثالث هو شكل كاف لتنظيم قاعدة البيانات، وهو كاف لتلافي تكرار البيانات ومشاكل الحذف والتعديل والإضافة.

قد يكون من المفيد في بعض الأحيان إعادة تركيب الجداول الناتجة عن التنظيم، لاعتبارات تسريع أداء قاعدة البيانات وهذا ما يدعى بإعادة التنظيم (denormalization).

مثال3:

يضم الجدول التالي بيانات توريد مواد.

Supplier_id	Supplier_name	Part_num	Quantity

بفرض أن اسم المورد لا يتكرر، فهناك مفتاحان مرشحان لهذه العلاقة (supplier_id ,part_num) و(supplier_name ,part_num).

التبعيات الموجودة في الجدول هي:

1. Supplier_id → supplier_name.
2. Supplier_name → supplier_id.
3. Supplier_id + part_num → quantity.
4. Supplier_id + part_num → supplier_name.
5. Supplier_name + part_num → supplier_id.
6. Supplier_name + part_num →quantity.

هذا الجدول هو من الشكل 3NF لأن الواصفة الوحيدة التي لا تشكل جزء من مفتاح مرشح هي quantity وهي لا تتحدد بجزء من مفتاح مرشح.

Supplier_id, supplier_name كلاهما محدد وليس مفتاح مرشح، فالجدول ليس من الشكل BCNF.

لجعل الجدول من الشكل BCNF يجب تجزئته على الشكل:

Supplier_id	Supplier_name	Part_num	Quantity

Supplier_id	Part_num	Quantity

Supplier_id	Supplier_name

3. الشكل النظامي الرابع (4NF)

يعتمد تعريف 4NF على مفهوم التبعية متعددة القيم (<<—) (multivalued dependency)، تحدث التبعية متعددة القيم في جدول يضم ثلاثة حقول على الأقل، عندما تتطابق قيم عدة أسطر من عمود مع قيمة سطر وحيد في عمود آخر، أي أن قيمة حقل تحدد مجموعة قيم لحقل آخر.

يكون الجدول من الشكل 4NF إذا كان من الشكل BCNF وكل واصفة تحددها (تحديد متعدد القيم) مجموعة واصفات أخرى يجب أن تعتمد كلياً على مجموعة الواصفات هذه.

مثال:

موظفين تسند لهم مشاريع ويتمتعون بكفاءات معينة:

Emp	Prj	Skill
1211	1	Analysis
1211	5	Analysis
1211	1	Design
1211	1	Prog

مفتاح الجدول السابق هو الواصفات الثلاث مجتمعة.

علاقات التبعية للجدول السابق هي:

1. Emp + prj —> skill.
2. Emp + skill —> prj.

الجدول السابق هو من الشكل BCNF، ولكنه ليس من الشكل 4NF، لأن:

skill تعتمد على emp أي تعتمد جزئياً على (emp, prj)، والأمر نفسه نجده في العلاقة 2، حيث prj تعتمد جزئياً على (emp, skill).

لوضع الجدول في الشكل 4NF يجب تجزئته على الشكل:

Emp	Prj	Skill
1211	1	Analysis
1211	5	Analysis
1211	1	Design
1211	1	Prog

Emp	Prj

Emp	Skill

4. الشكل النظامي الخامس (5NF):

يكون الجدول من الشكل النظامي الخامس إذا لم يكن بالإمكان تجزئته إلى جداول أصغر.

- بافتراض أن لدينا الجدول T(A,B,C,D)، والتبعيات الوظيفية هي:

A→BCD

BC→D

D→B

هذا الجدول من الشكل النظامي الثالث، والمفاتيح هي A و BC

هل هذا الجدول من الشكل النظامي BCNF؟ لا

التبعية الوظيفية الأولى والثانية لا تخرق شرط BCNF لكون المحدد هو مفتاح مرشح.

في التبعية الثالثة D ليس مفتاح مرشح، بالتالي الجدول ليس من الشكل النظامي BCNF.

الفصل الثامن:

لغة الاستعلامات المهيكلة
(SQL) (1)

العنوان	الصفحة
1. مقدمة	3
2. تعليمات SQL	3
3. تعليمات DDL	4
1.3 توليد جدول Create Table	4
2.3 حذف جدول Drop Table	5
3.5 تعديل بنية جدول Alter Table	6
4. تعليمات DML	8
1.4 إدخال بيانات لجدول Insert	8
2.4 تعديل بيانات جدول Update	9
3.4 حذف بيانات من جدول Delete	10
5. تعليمات DCL	11
1.5 منح صلاحية Grant	11
2.5 حجب صلاحية Revoke	11
6. تعليمات TCL	12
7. تعليمات DQL	13
8. تحليل نموذج عن تعليمات SQL	14
9. المراجع	16

الكلمات المفتاحية:

جدول، سجل، عمود، تابع، قيمة، بيانات، حقل، جدول، رقمية، سلاسل محارف، تاريخ، صيغة، قاعدة بيانات.

ملخص:

هذه الوحدة هي ملخص وتذكرة بتعليمات لغة الإستعلام المُهيكلة والشكل الأبسط لكل من هذه التعليمات.

أهداف تعليمية:

يهدف هذا الفصل التعريف بالمفاهيم التالية:

- لمحة سريعة عن لغة الاستعلام المهيكلة
- الشكل الأبسط لتعليمات SQL:
  1. تعريف البنى في قاعدة المعطيات (DDL).
  2. معالجة البيانات (DML).
  3. تعليمات التحكم بالوصول للبيانات (DCL).
  4. تعليمات إدارة المناقلات (TCL).
  5. تعليمات اختيار البيانات (DQL).
- تحليل نموذج عن تعليمات SQL

1. مقدمة

يرمز اختصار SQL إلى "لغة الاستعلام المهيكلة" Structured Query Language.

تُستخدم لغة SQL للولوج إلى قواعد المعطيات والتعامل معها. ويقوم "المعهد الوطني الأميريكي للمقاييس" American National Standards Institute بإدارة مقاييس ومعايير هذه اللغة، ويشار إلى الهيئة المختصة بمواصفات هذه اللغة بهيئة ANSI SQL.

تُعتبر لغة SQL لغةً بسيطةً نسبياً، ولكنها فعالة للغاية، فالكثير من التعليمات البسيطة في هذه اللغة تخفي ورائها خصائص فعّالة يمكن استخدامها للقيام بالعديد من العمليات المعقدة المعروفة في قواعد المعطيات.

2. تعليمات SQL

يمكن تصنيف تعليمات SQL إلى خمس مجموعات حسب الغاية منها، وهي:

DDL (Data definition language) تعريف البنى في قاعدة المعطيات

Create	لإنشاء جدول أو أي غرض آخر في قاعدة المعطيات
Alter	لتعديل بنية غرض في قاعدة المعطيات
Drop	لحذف غرض من قاعدة المعطيات

DML (Data manipulation language) لمعالجة البيانات في قاعدة المعطيات

Insert	لإدخال بيانات في جدول
Update	تعديل البيانات في جدول
Delete	حذف بيانات من جدول

DCL (Data Control language) للتحكم بالوصول إلى بيانات جدول

Grant	منح صلاحية إدخال أو تعديل أو حذف لمستخدم
Revoke	حجب صلاحية عن مستخدم

DQL (Data Query language) لاختيار بيانات من جدول

Select	انتقاء وعرض بيانات من جدول أو أكثر

TCL (Transaction Control language) لإدارة المناقلات

Commit	لتثبيت التعديلات بشكل نهائي
Rollback	للتراجع عن التعديلات

3. تعليمات DDL

توليد جدول Create Table

تُستخدم تعليمة Create Table لتوليد جدول جديد ضمن قاعدة المعطيات. والشكل الأبسط لهذه التعليمة هو:

```
Create table ( tablename (column1 data type,
column2 data type, column3 data type);
```

ويكون لها الشكل المُعقد التالي في حال تم وضع شروط تكامل مرجعي على حقول الجدول (Primary Key, Foreign key, unique, not null):

```
Create table table_name
(column1  data type          [constraint],
 column2  data type          [constraint],
 column3  data type          [constraint]);
```

```
Create table table_name
(column1  data type          [constraint],
 column2  data type          [constraint],
 column3  data type          [constraint]);
```

- يبدأ اسم الجدول واسم أي حقل فيه، بحرف حصراً، ويمكن أن تتبعه أرقام أو حروف أو إشارات "_" (underscores)، شرط ألا يزيد طول سلسلة المحارف الممثلة للإسم عن 30 محرف
- يجب تجنب استخدام أي كلمة مفتاحية مثل SELECT أو CREATE كاسم لجدول أو لحقل
- يحدد البند data type أنماط المعطيات والتي تكون: char(size)، varchar(size)، number(size)، date، number(size,d)

char(size)	سلسلة محارف بطول ثابت مُحدد في size لا يزيد على 255 بايت
varchar(size)	سلسلة محارف بطول متغير مُحدد في size
number(size)	نمط عددي صحيح بعدد أرقام مُحدد في size
date	تاريخ
number(size,d)	نمط عددي حقيقي بعدد أرقام قبل الفاصلة مُحدد في size ومُحدد في d بعد الفاصلة

```
Create table Employee
( Emp_Id  Number          Primary key,
 Name           varchar(30),
 BOD            Date);
```

Id	Name	DOB
1	John Smith	1/1/107
2	Samer	2/4/1994

حذف جدول Drop Table

تُستخدم تعليمة Drop Table لحذف جدول من قاعدة المعطيات. والشكل الأبسط لهذه التعليمة هو:

```
Drop table table_name;
```

مثال:

```
Drop table Employee;
```

تعديل بنية جدول Alter Table

تُستخدم تعليمة Alter Table لتعديل بنية جدول من قاعدة المعطيات. علماً أن التعديل يتضمن إضافة حقل، تعديل نمط حقل، حذف حقل وإعادة تسمية أحد الحقول، والشكل الأبسط لهذه التعليمة هو:

```
Alter table table-name add (column1-name datatype,
column2-name datatype, column3-name datatype);

Alter table table-name modify (column-name datatype);

Alter table table-name drop (column-name);

Alter table table-name rename old-column-name to column-
name;
```

مثال:

Employee

Id	Name	DOB
1	John Smith	1/1/1071
2	Samer	2/4/1994

Alter table Employee drop (DOB);

Id	Name
1	John Smith
2	Samer

Alter table Employee add (Hire_date date);

Id	Name	Hire_date
1	John Smith	
2	Samer	

Alter table Employee Modify (Hire_date varchar(50));
Insert into Employee values(5,'Jamal','January 2015');

Id	Name	Hire_date
1	John Smith	
2	Samer	
5	Jamal	January 2015

Alter table Empoyee rename Hire_date to Hire_month;

Id	Name	Hire_month
1	John Smith	
2	Samer	
5	Jamal	January 2015

لتعديل نمط حقل في جدول يجب أن تكون القيم المتضمنة في هذا الحقل قابلة للتحويل إلى النمط الجديد، وإلا فإن رسالة خطأ ستنتج عن تطبيق التعليمة.

في حالة تعديل حقل، يمكن أن يكون التعديل هو تغيير نمط الحقل، أو إضافة قيمة افتراضية أو شرط تكامل مرجعي.

4. تعليمات DML:

إدخال بيانات لجدول Insert

تُستخدم تعليمة Insert لإدخال بيانات إلى جدول جديد ضمن قاعدة المعطيات. والشكل الأبسط لهذه التعليمة هو:

```
Insert into table-name values (data1,data2,...);
Insert into table-name (column1, column2,…) values
(data1,data2,...);
```

مثال:

Employee

Id	Name	Hire_month
1	John Smith	
2	Samer	
5	Jamal	January 2015

```
Insert into Employee values (8,'John Do','2013');
```

Id	Name	Hire_month
1	John Smith	
2	Samer	
5	Jamal	January 2015
8	John Do	2013

```
Insert into Employee (Id,Name) values (9,'xxx');
```

Id	Name	Hire_month
1	John Smith	
2	Samer	
5	Jamal	January 2015
8	John Do	2013
9	xxx	

- الحقول التي لا يتم إدخال قيم لها، تأخذ قيمة Null إذا لم يتعارض ذلك مع شرط تكامل مرجعي.
- يمكن إدخال Null بشكل صريح لأحد الحقول، على الشكل التالي:

```
Insert into Employee values (15, 'yyy', Null);
```

تعديل بيانات جدول Update

تُستخدم تعليمة Update لتعديل بيانات تسجيلة أو أكثر في جدول ضمن قاعدة المعطيات. والشكل الأبسط لهذه التعليمة هو:

```
Update table-name set column-name = value where condition;
```

مثال:

Employee

Id	Name	Hire_month
1	John Smith	
2	Samer	
5	Jamal	January 2015
8	John Do	2013

```
------
Update Employee set Hire_month='June 2013' where id=8;
--------
```

Id	Name	Hire_month
1	John Smith	
2	Samer	
5	Jamal	January 2015
8	John Do	June 2013

```
-----
```

```
Update Employee set Name='Smith';
```

Id	Name	Hire_month
1	Smith	
2	Smith	
5	Smith	January 2015
8	Smith	2013

سنتعرف على الشروط الممكنة وكيفية استخدامها، في تعليمة Update عند الحديث لاحقاً عن تعليمة الاختيار Select

حذف بيانات من جدول Delete

تُستخدم تعليمة Delete لحذف تسجيلة أو أكثر من جدول ضمن قاعدة المعطيات. والشكل الأبسط لهذه التعليمة هو:

```
Delete from  table-name where condition;
```

مثال:

Employee

Id	Name	Hire_month
1	John Smith	
2	Samer	
5	Jamal	January 2015
8	John Do	2013

```
Delete from  Employee where id>=5;
```

Id	Name	Hire_month
1	John Smith	
2	Samer	

عدم استخدام الشرط يؤدي إلى حذف جميع سجلات الجدول، فمثلاً التعليمة "Delete from Emplyee" ستؤدي إلى حذف جميع سجلات الجدول Employee.

5. تعليمات DCL

لتنفيذ أي عملية على قاعدة بيانات أو أي غرض منها، (إنشاء جدول، إدخال بينات للجدول، إنشاء اتصال مع قاعدة البيانات ...) يحتاج المستخدم إلى صلاحية مناسبة، يمكن لمدير النظام أو مالك الصلاحية منحها لمستخدم باستخدام Grant، ويمكن حجب الصلاحية باستخدام Revoke.

منح صلاحية Grant

الشكل الأبسط لهذه التعليمة هو:

```
Grant privilege on object-name to user-name;
Grant role to user-name;
```

مثال:

```
Grant select on Employee to Scott;
Grant sysdba to Scott;
```

حجب صلاحية Revoke

الشكل الأبسط لهذه التعليمة هو:

```
Revoke privilege on object-name from user-name;
Revoke role from user-name;
```

مثال:

```
Revoke select on Employee from Scott;
Revoke sysdba from Scott;
```

6. تعليمات TCL

تعليمات إدارة المناقلات هي تعليمتين، Commit لتثبيت التعديلات التي تمت منذ آخر استدعاء لها، و Rollback للتراجع إلى آخر تعليمة Commit. عند أنهاء جلسة (بشكل نظامي) لمستخدم متصل مع قاعدة البيانات يتم استدعاء Commit بشكل ضمني، وعند انهاء الجلسة بشكل غير نظامي يتم استدعاء Rollback.

مثال:

Employee

Id	Name	Hire_month
1	John Smith	
2	Samer	
5	Jamal	January 2015
8	John Do	2013

```
-----
Delete from  Employee where id=5;
Commit;
Delete from  Employee where id=8;
Commit;
Insert into Employee values (11,'xxx','2015');
Delete from employee;
Rollback;
-----
```

Id	Name	Hire_month
1	John Smith	
2	Samer	

7. تعليمات DQL

تعليمات اختيار البيانات هي تعليمة واحدة Select لها أشكال وخيارات متعددة، تستخدم تعليمة لاسترجاع بيانات من جدول أو أكثر، وتستخدم أيضاً لمعالجة البيانات المسترجعة.

سنعرض هنا أبسط أشكالها مع نماذج عن استخدامها، وسنفصل في جميع الخيارات الممكنة لها في الفصل التالي.

الشكل الأبسط لهذه التعليمة هو:

```
Select column1_name, Column2_name,…
From table-name;
```

مثال:

Employee

Id	Name	Hire_month
1	John Smith	
2	Samer	
5	Jamal	January 2015
8	John Do	2013

```
Select Name, Id from Employee;
```

Name	Id
John Smith	1
Samer	2
Jamal	5
John Do	8

```
Select * from Employee;
```

Id	Name	Hire_month
1	John Smith	
2	Samer	
5	Jamal	January 2015
8	John Do	2013

- ترتيب استرجاع البيانات من الجدول ليس بالضرورة هو نفس ترتيب إدخالها.

8. تحليل نموذج عن تعليمات SQL

في المثال التالي سنقوم بتحليل نموذج عن تعليمة Select لتكون مقدمة لدرسنا القادم:

```
SELECT select_list [ INTO new_table ]
FROM table_source
[ WHERE search_condition ]
[ GROUP BY group_by_expression ]
[ HAVING search_condition ]
[ ORDER BY order_expression [ ASC | DESC ] ]
-----
Employee
```

Id	Name	Hire_month
1	John Smith	
2	Samer	
5	Jamal	January 2015
8	John Do	2013

```
Select          Name, Id
From            Employee
Where           id<5
Order by  Name desc;
```

تعليمة Select السابقة توضح أن:

- الحقول المطلوب عرضها هي الاسم Name والرقم Id وبالترتيب
- مصدر البيانات هو الجدول Employee
- البيانات المطلوبة هي التي تحقق الشرط (الرقم أصغر تماماً من 5)
- ترتيب عرض النتيجة هي حسب الترتيب الأبجدي للاسم تنازلياً
- لا يمكن الاستغناء عن الجزء الأول والثاني من تعليمة Select ، بينما الجزء الثالث والرابع يمكن حذفهما إذا لم يكن هناك شرط على التسجيلات المسترجعة من الجدول، ولا شرط على ترتيب النتيجة المسترجعة
- Desc في الجزء الرابع تعني ترتيب تنازلي، والقيمة الافتراضية هي Asc أي تصاعدي

يمكن إضافة أكثر من شرط في الجزء الثالث من التعليمة، وربط الشروط يتم بالعمليات المنطقية And, Or, Not

---

Name	Id
2	Samer
1	John Smith

تعليمة Select هي موضوع الفصل التالي

9. المراجع:

- http://www.studytonight.com/dbms/select-query

الفصل التاسع:

لغة الاستعلامات المهيكلة (SQL) (2)

العنوان	الصفحة
1. مقدمة	3
2. استرجاع البيانات من الجداول المصدر	3
1.2 الاختيار Selection	3
2.2 الاسقاط Projection	4
3.2 الضم Join	5
4.2 الجداء Product	8
5.2 الاجتماع Union	9
6.2 التقاطع Intersection	11
7.2 الفرق Difference	12
8.2 متممات	13
3. ترتيب البيانات المسترجعة Order by clause	14
4. تجميع البيانات المسترجعة واختيار المجموعات Group by & Having clause	16
5. إلغاء التكرار في البيانات المسترجعة Distinct Keyword	17
6. تطبيق	18
7. المراجع	20
8. تدريبات	21

الكلمات المفتاحية:

جدول، سجل، عمود، تابع، قيمة، بيانات، حقل، جدول، رقمية، سلاسل محارف، تاريخ، صيغة، قاعدة بيانات.

ملخص:

هذه الوحدة هي عرض لتعليمات لغة الإستعلام المُهيكلة والأشكال المحتملة لكل تعليمة، مع شرح المفهوم النظري لحالات استخدام تعليمات SQL.

أهداف تعليمية:

يهدف هذا الفصل التعريف بالمفاهيم التالية:

- استرجاع البيانات من الجداول المصدر:
  1. الاختيار Selection
  2. الاسقاط Projection
  3. الضم Join
  4. الجداء Product
  5. الاجتماع Union
  6. التقاطع Intersection
  7. الفرق Difference
  8. متممات
- ترتيب البيانات المسترجعة Order by clause
- تجميع البيانات المسترجعة واختيار المجموعات Group by & Having clause
- الغاء التكرار في البيانات المسترجعة Distinct Keyword
- تطبيق

1. مقدمة

تستخدم تعليمة Select لاسترجاع بيانات من جدول أو أكثر، وتستخدم أيضاً لمعالجة البيانات المسترجعة:

```
SELECT select_list [ INTO new_table ]
FROM table_source
[ WHERE search_condition ]
[ GROUP BY group_by_expression ]
[ HAVING search_condition ]
[ ORDER BY order_expression [ ASC | DESC ] ]
```

في هذا الفصل سنحلل جميع عناصر تعليمة Select المعروضة في الشكل السابق، مع شرح المفهوم النظري لكل من أجزاء التعليمة.

2. استرجاع البيانات من الجداول المصدر

الاختيار Selection

يعني الاختيار استرجاع مجموعة جزئية من تسجيلات المصدر، يتم تحديد السجلات المسترجعة بشرط أو مجموعة شروط، ويمكن جمع عدة شروط بالمعاملات المنطقية and, or, not.

Employee

Id	Name	Section
1	A	HR
2	B	Finance
3	C	HR
4	D	Sales

```
Select *
From Employee
Where id <=3 and name <> 'B' ;
```

```
----
```

Id	Name	Section
1	A	HR
3	C	HR

الإسقاط Projection

يعني الاسقاط استرجاع مجموعة جزئية من أعمدة المصدر، ويمكن ترتيب الأعمدة المسترجعة بغير ترتيبها في المصدر ويمكن أيضاً اعطاؤها تسميات جديدة.

Employee

Id	Name	Section
1	A	HR
2	B	Finance
3	C	HR
4	D	Sales

```
Select section as S, Name as E
From Employee
Order by section;
```

```
----
```

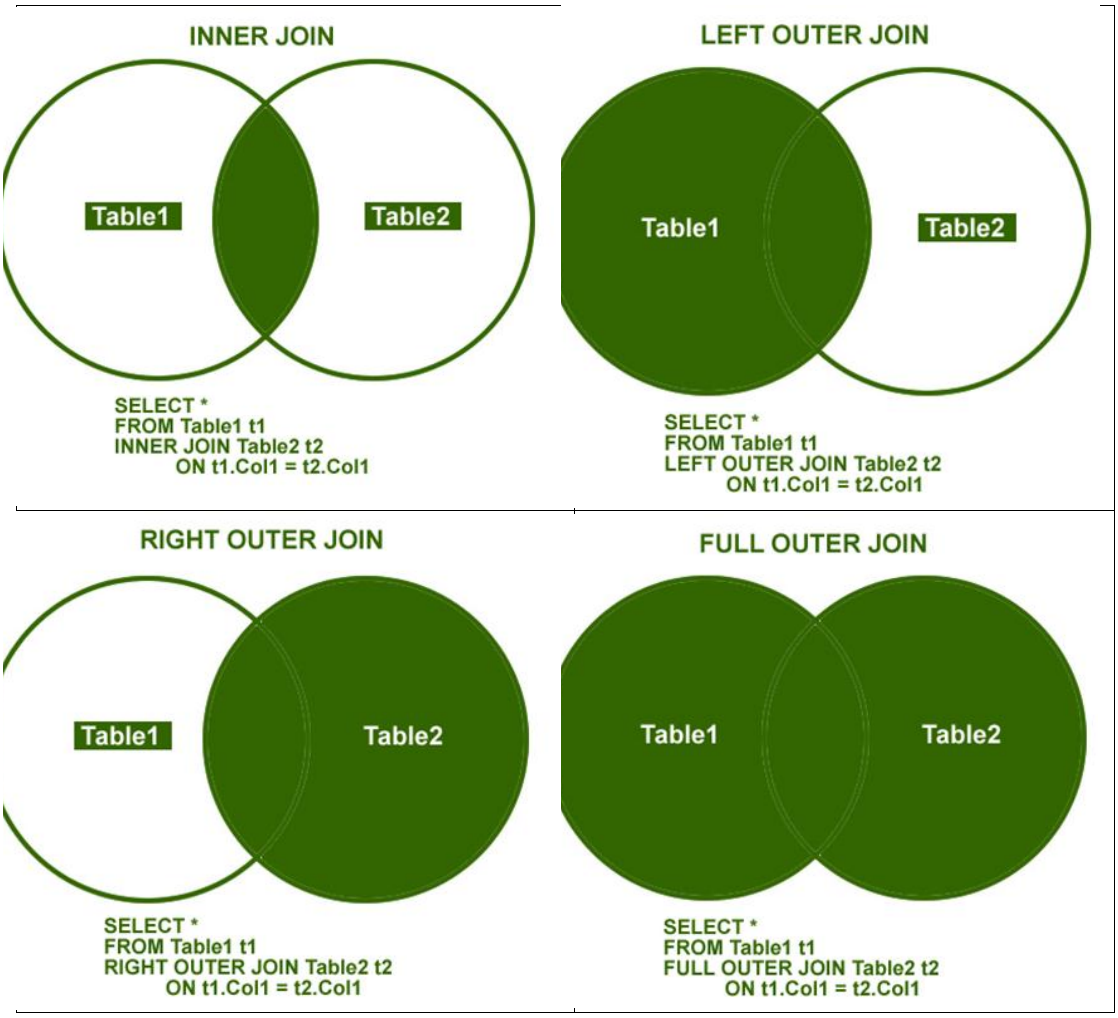
S	E
Finance	B
HR	A
HR	C
Sales	D

الضم Join

عند اختيار البيانات من أكثر من مصدر (جدولين مثلاً)، وفي حال عدم ضم الجدولين (ربطهما) بإحدى طرق الضم الأربع المشروحة في هذه الفقرة، فإننا سنحصل على الجداء الديكارتي لمحتوى الجدولين.

طرق الضم هي:

نمط الضم	الطرف المشارك اليميني	الطرف المشارك اليساري
Inner Join	فقط السجلات التي لها مقابل في الطرف الآخر	فقط السجلات التي لها مقابل في الطرف الآخر
Left Join	فقط السجلات التي لها مقابل في الطرف الآخر	جميع السجلات
Right Join	جميع السجلات	فقط السجلات التي لها مقابل في الطرف الآخر
Outer Join	جميع السجلات	جميع السجلات



مثال:

Employee

E_Id	E_Name	E_Section
1	A	3
2	B	4
3	C	3
4	D	

Section

S_Id	S_Name
1	HR
2	Finance
3	Production
4	Sales

```
Select S_Name,E_Name from
Employee inner join Section
On Employee.E_Section=Section.S_Id;
```

S_Name	E_Name
Production	A
Sales	B
Production	C

```
Select S_Name,E_Name from
Employee Left outer join Section
On Employee.E_Section=Section.S_Id;
```

S_Name	E_Name
Production	A
Sales	B
Production	C
	D

```
Select S_Name,E_Name from
Employee Right outer join Section
On Employee.E_Section=Section.S_Id;
```

S_Name	E_Name
Production	A
Sales	B
Production	C
HR	
Finance	

```
Select S_Name,E_Name from
Employee Full Outer join Section
On Employee.E_Section=Section.S_Id;
```

S_Name	E_Name
Production	A
Sales	B
Production	C
	D
HR	
Finance	

الجداء Product

في حال اختيار السجلات من جدولين دون تطبيق احدى طرق الضم عليهما، فإننا سنحصل على الجداء الديكارتي (كل سجل من الطرف الاول مع جميع سجلات الطرف الثاني).
بتطبيق ذلك على الجدولين Employee و Section سنحصل على ما يلي:

Employee

E_Id	E_Name	E_Section
1	A	3
2	B	4
3	C	3
4	D	

Section

S_Id	S_Name
1	HR
2	Finance
3	Production
4	Sales

```
Select S_Name,E_Name from
Employee, Section;
----
```

S_Name	E_Name
HR	A
HR	B
HR	C
HR	D
Finance	A
Finance	B
Finance	C
Finance	D
Production	A
Production	B
Production	C
Production	D
Sales	A
Sales	B
Sales	C
Sales	D

الاجتماع Union

تجمع هذه العملية كافة الأسطر المسترجعة من تعليمتي Select، دون تكرار الأسطر الموجودة في نيتجة كلا التعليمتين، ولتطبيق هذه العملية يجب أن تتطابق النتيجتين اللتين نجمعهما في ترتيب وأنماط الأعمدة.

مثال:

Staff

Id	Name	Location
1	A	Sales section
2	B	HR section
3	C	Sales section

Manager

Id	Department	Name
1	HR section	B
3	QA	D

```
Select location as  dept from Staff
Union
Select Department as dept  from Manager;
```

---

dept
Sales section
HR section
Sales section
QA

```
Select * from Staff
Union
Select * from Manager;
```

?	?	?
1	A	Sales section
2	B	HR section
3	C	Sales section
1	HR section	B
3	QA	D

```
Select id, name, location from Staff
Union
Select id, name, department from Manager;
```

?	?	?
1	A	Sales section
2	B	HR section
3	C	Sales section
3	D	QA

التقاطع Intersection

تعطي هذه العملية تقاطع الأسطر المسترجعة من تعليمتي Select (الأسطر المكررة في نتيجة التعليمتين)، دون تكرار، ولتطبيق هذه العملية يجب أن تتطابق النتيجتين اللتين نقاطعهما في ترتيب وأنماط الأعمدة.

مثال:

Staff

Id	Name	Location
1	A	Sales section
2	B	HR section
3	C	Sales section

Manager

Id	Department	Name
1	HR section	B
3	QA	D

```
Select location as  dept from Staff
Intersect
Select Department as dept  from Manager;
```

---

dept
HR section

```
Select * from Staff
Intersect
Select *  from Manager;
```

Nothing

```
=======
```

```
Select name, location from Staff
Intersect
Select name, department  from Manager;
```

```
_____
```

?	?
B	HR section

الفرق Difference

تعطي هذه العملية فرق الأسطر المسترجعة من تعليمتي Select (الأسطر المسترجعة من التعليمة الأولى باستثناء الموجودة في نتيجة التعليمة الثانية)، ولتطبيق هذه العملية يجب أن تتطابق النتيجتين اللتين نقاطعهما في ترتيب وأنماط الأعمدة.

مثال:

Staff

Id	Nam	Location
1	A	Sales section
2	B	HR section
3	C	Sales section

Manager

Id	Department	Name
1	HR section	B
3	QA	D

```
Select location from Staff
Except
Select Department from Manager;
```

```
____
```

Location
Sales section

متممات

تعبير Like:

عند اختيار مجموعة جزئية من السجلات لاسترجاعها من المصدر، يمكن وضع شرط أو عدة شروط مرتبطة بالعمليات المنطقية (and, or, not)، وتختلف بنية الشرط حسب نمط المعطيات للحقول التي يتم تطبيق الشروط عليها.

في حال كان الحقل المراد تطبيق شرط عليه من نمط نصي، فإن أهم تعبير للمقارنة هو Like، وفيما يلي أمثلة على استخدامه.

Employee

Id	Name	Location
1	A	Sales section
2	B	HR section
3	C	Sales section

```
Select * from Employee where Name like 'A%';
```
يبدأ الاسم بحرف A

```
Select * from Employee where Name like '%A';
```
ينتهي الاسم بحرف A

```
Select * from Employee where Name like '%A%';
```
يحوي الاسم حرف A

```
Select * from Employee where Name like 'A_';
```
الاسم مؤلف من حرفين أولهما A

```
Select * from Employee where Name like '_A%';
```
الحرف الثاني من الاسم هو A

% يمكن استبدالها بأي سلسلة محارف

_ تستبدل بمحرف وحيد

تعبير Into:

يمكن تخزين نتيجة تطبيق تعليمة Select من التسجيلات في جدول جديد في قاعدة المعطيات، لتحفظ بشكل دائم.

Employee

Id	Name	Location
1	A	Sales section
2	B	HR section
3	C	Sales section

```
Select id, name into HREmp from Employee
Where location='HR section';
```

=======

انشاء جدول جديد باسم HREmp يتضمن بيانات موظفي قسم شؤون الموظفين.

HREmp

Id	Location
1	Sales section
2	HR section
3	Sales section

3. ترتيب البيانات المسترجعة Order by clause:

عند اختيار مجموعة جزئية من السجلات لاسترجاعها من المصدر فلا ضمانة على ترتيب السجلات المسترجعة (الترتيب ليس بالضرورة حسب ترتيب الادخال)، ولترتيب النتيجة المسترجعة لا بد من استخدام Order by، ويمكن الترتيب تصاعدياً أو تنازلياً (asc | desc)، علماً أن الترتيب للحقول الرقمية حسب الأصغر والأكبر بينما الترتيب للحقول النصية يتم حسب الترتيب الأبجدي، ويتم ترتيب حقول التاريخ حسب الأقدم والأحدث.

Employee

Id	Name	Location
1	Ahmad	HR Section
2	Bassem	Sales Section
3	Ahmad	Sales Section
4	Mazen	HR Section

```
Select * from Employee id desc;
```

=====

النتيجة مرتبة حسب تسلسل الرقم المعرف تنازلياً.

Employee

Id	Name	Location
4	Mazen	HR Section
3	Ahmad	Sales Section
2	Bassem	Sales Section
1	Ahmad	HR Section

=====

```
Select * from Employee order by name asc, id desc;
```

====

Employee

Id	Name	Location
3	Ahmad	Sales Section
1	Ahmad	HR Section
2	Bassem	Sales Section
4	Mazen	HR Section

النتيجة مرتبة بحسب أبجدية الاسم، وفي حال تشابه اسمين يتم ترتيبهما حسب الرقم المعرف تنازلياً.

4. تجميع البيانات المسترجعة واختيار المجموعات Group by & Having clause

يمكن تجميع البيانات المسترجعة من عملية Select حسب واحد أو أكثر من الأعمدة المسترجعة باستخدام Group by، ويمكن اختيار مجموعات من المجموعات الناتجة بوضع شرط على المجموعة باستخدام Having.
يجب التمييز هنا بين Where التي تضع شرطاً على التسجيلات المسترجعة، و Having التي تضع شرطاً على المجموعات المسترجعة.

مثال:

Employee

Id	Name	Age	Salary
401	Anu	22	9000
402	Shane	29	8000
403	Rohan	34	6000
404	Scott	44	9000
405	Tiger	35	8000

```
Select name, age from Employee group by salary;
```

======

خطأ: الحقول المنتقاة يجب أن تكون إما جزء من الحقول التي يجري التجميع حسبها، أو نتيجة تطبيق تابع يعطي قيمة وحيدة لكل مجموعة.

=====

```
Select salary,count(name) from Employee group by salary;
```

======

Salary	Count(salary)
9000	2
8000	2
6000	1

=======

```
Select salary,count(name)
from Employee
group by salary
having count(name)>1 ;
```

=====

Salary	Count(salary)
9000	2
8000	2

5. إلغاء التكرار في البيانات المسترجعة Distinct Keyword

مثال:

Employee

Id	Name	Location
1	Ahmad	HR Section
2	Bassem	Sales Section
3	Ahmad	Sales Section
4	Mazen	HR Section

```
Select distinct name from Employee;
```

في حال وجود اسم مكرر يظهر مرة واحدة في النتيجة المسترجعة.

======

Name
Ahmad
Bassem
Mazen

==========

Employee

Id	Name	Location
1	Ahmad	HR Section
2	Bassem	Sales Section
3	Ahmad	HR Section
4	Mazen	HR Section

```
Select distinct (name, location) from Employee;
```

في حال وجود اسم مكرر بنفس مكان العمل يظهر مرة واحدة في النتيجة المسترجعة.

======

Name	Location
Ahmad	HR Section
Bassem	Sales Section
Mazen	HR Section

6. تطبيق

تتألف قاعدة المعطيات في تطبيقنا من ثلاثة جداول كما يلي:

- Person

Id	FName	LName	Gender	Nationality	Age
1	John	Smith	1	2	32
2	Adam	Sandler	1	1	45
3	Mary	Clair	0	1	40
…	…	…	…	…	…

==========

- Nationality

Id	Name
1	American
2	Frensh
…	…

==========

- ContactMedia

Id	Media
1	Mobile
2	E-Mail
3	Phone
…	…

==========

- PersonContact

Id	Person	ContactMedia	ContactValue
1	1	1	00963944111222
2	1	2	xxx@gmail.com
…	…	…	…

========

في مثالنا تصدر الجنسية Nationality مفتاحها إلى الجدول Person (ارتباط واحد لعدة)، وترتبط طريقة الاتصال ContactMedia مع جدول Person من خلال جدول وسيط هو PersonContact (يمكن أن يكون لنفس الشخص أكثر من طريقة للاتصال به).

المطلوب:

- قائمة بأسماء الذكور مرتبة أبجدياً؟
- قائمة بأسماء الأشخاص الذين لهم بيانات اتصال مدخلة؟
- قائمة بكل زوج من الأشخاص فارق العمر بينهما أقل من 5 سنوات، وهما من جنسين مختلفين؟
- قائمة بأرقام موبايل الأشخاص من جنسية فرنسية؟
- قائمة بأسماء الأشخاص الذين ليست لهم بيانات اتصال مدخلة؟

==========

7. المراجع:

http://www.studytonight.com/dbms/select-query

8. تدريبات

- Invoice

Id	Serial	Date	Client
1	1	1-1-2015	20
2	2	2-1-2015	30
3	3	1-1-2015	10
4	4	4-2-2015	20
5	5	4-2-2015	10

- Client

Id	Name
10	C 1
20	C 2
30	C 3

- Material

Id	Name	Price
1	Printer	15000
2	Monitor	30000
3	Case	2500
4	MB	22000

- Invoice_Item

Id	Invoice	Material	QTY
1	1	1	1
2	1	2	1
3	1	4	1
4	2	2	2
5	2	3	2

- نتيجة تطبيق جملة التعليمات التالية:

```
Select      Sum (price)
From        Invoice, Material, Invoice_Item
Where       Invoice.Id=Invoice_item.Invoice
            And Invoice_Item.material=Material.Id
            And Invoice.id=2;
```

هي:

1. اجمالي سعر جميع فواتير.
2. اجمالي الفاتورة رقم 2.
3. اجمالي فواتير الزبون 1.
4. مجموع أسعار المواد المتضمنة في الفاتورة رقم 2.

الإجابة: (4)

للحصول على اجمالي الفاتورة رقم 2 يجب استبدال Sum(price) بالعبارة Sum(Price * QTY).

- نتيجة التعليمة التالية:

```
Select      Sum (QTY * Price)
From        Invoice, Material, Invoice_Item
Where       Invoice.Id=Invoice_item.Invoice
            And Invoice_Item.material=Material.Id
            And Client.Name='C 1';
```

هي:

1. اجمالي فواتير الزبون C 1.
2. اجمالي الفواتير بصرف النظر عن الزبون.
3. خطأ.
4. مجموع أسعار المواد المتضمنة في فواتير الزبون C 1.

الإجابة: (4)

الجدول المصدر لاسم الزبون غير وارد في قائمة From

- نتيجة التعليمة التالية:

```
Select      Sum (QTY * Price)
From        Invoice, Material, Invoice_Item
Where       Invoice.Id=Invoice_item.Invoice
            And Invoice_Item.material=Material.Id
Group by    Invoice_Item.Invoice
Having      count(Invoice_item.id) >1;
```

هي:

1. قائمة باجمالي الفواتير.
2. قائمة باجمالي الفواتير التي تحوي أكثر من قلم.
3. مجموع قيم الفواتير التي تحوي أكثر من قلم .
4. اجمالي الفواتير مرتبة حسب رقم الفاتورة.

الإجابة: (2)

- نتيجة التعليمة التالية:

```
Select      distinct Invoice.Date
From        Invoice, Client
Where       Invoice.Client=Client.Id
            And Client.Name = 'C 2';
```

هي:

1. تاريخ آخر فاتورة للزبون C 2.
2. قائمة بتاريخ الأيام التي اشترى فيها الزبون C 2 بدون تكرار.
3. قائمة بتاريخ فواتير الزبون C 2 .
4. اجمالي فواتير الزبون C 2.

الإجابة: (2)

الفصل العاشر:

لغة الاستعلامات المهيكلة
(SQL) (3)

العنوان	الصفحة
1. مقدمة	4
2. شروط التكامل المرجعي (Integrity constraints)	4
1.2 المفتاح الأساسي Primary key	4
2.2 المفتاح المستورد Foreign key	4
3.2 الوحدانية Unique	5
4.2 اجباري Not Null	5
3. القيم الافتراضية Default	6
4. شروط التحقق Check	6
5. التوابع في SQL	7
1.5 التوابع التجميعية	7
2.5 التوابع الدرجية	9
6. الأسماء البديلة Aliases	11
7. تعليمة Select المتداخلة (nested select statement)	13
8. تعليمات Select المترابطة (Correlated select statement)	15
9. تطبيق	15
10. المراجع	18
11. تدريبات	19

الكلمات المفتاحية:

جدول، سجل، عمود، تابع، قيمة، بيانات، حقل، جدول، رقمية، سلاسل محارف، تاريخ، صيغة، قاعدة بيانات.

ملخص:

هذه الوحدة هي توسع في لغة الإستعلام المُهيكلة وتعرض مفاهيم متقدمة في SQL وتدريبات تطور مهارة الطالب في استخدام SQL.

أهداف تعليمية:

يهدف هذا الفصل التعريف بالمفاهيم التالية:

- شروط التكامل المرجعي في SQL:
  1. المفاتيح الأساسية Primary Keys
  2. المفاتيح المستوردة Foreign Keys
  3. الوحدانية Unique
  4. الالزامية Not null
- القيم الافتراضية Default
- شروط التحقق Check
- التوابع في SQL
  5. التوابع التجميعية
  6. التوابع الدرجية
- الأسماء البديلة Aliases
- تعليمة Select **المتداخلة (Nested Select statement)**
- تعليمات Select المترابطة Correlated Select statement
- تطبيق

المخطط:

- مقدمة
- شروط التكامل المرجعي (Integrity constraints)
- القيم الافتراضية Default
- شروط التحقق Check
- التوابع في SQL
- الأسماء البديلة Aliases
- تعليمة Select المتداخلة (nested select statement)
- تعليمات Select المترابطة (Correlated select statement)
- تطبيق

1. مقدمة

في هذا الفص سنعرض حالات متقدمة لاستخدام تعليمة Select موضوع الفصل السابق، وتتمات أساسية في لغة الاستعلام المهيكلة.

2. شروط التكامل المرجعي (Integrity constraints):

شروط التكامل المرجعي هي مجموعة من القواعد، تطبق على نمط وقيم البيانات التي يمكن إدخالها إلى أعمدة الجداول، والغاية منها ضمان صحة وتكامل البيانات في مجموعة جداول قاعدة البيانات.
شروط التكامل المرجعي هي: مفتاح أساسي (Primary Key)، مفتاح مستورد (Foreign Key)، وحيد (Unique)، اجباري (Not Null).

المفتاح الأساسي Primary key

المفتاح الأساسي هو حقل يعرف بشكل وحيد كل تسجيلة من الجدول، وقد شرحنا سابقاً كيفية اختياره من المفاتيح الأعظمية والمفاتيح المرشحة (راجع الفصل: النموذج العلائقي لقواعد البيانات)، أما كيفية تعريفه فهي على الشكل التالي:

```
Create table Student (s_id int PRIMARY KEY, Name varchar(60), Age int);
```

الحقل s_id هو المفتاح الأساسي للجدول Student، لا يمكن ادخال قيم مكررة أو Null في هذا الحقل. ومعرفة s_id تحدد حتماً قيم بقية حقول الجدول.

المفتاح المستورد Foreign key

المفتاح المستورد هو وسيلة ربط جدولين، بحيث تكون قيم الحقل المستورد في الجدول الابن هي حتماً موجودة في حقل المفتاح الأساسي في الجدول الأب، إن لم تكن Null.

يمكن فهم المفتاح المستورد من خلال المثال التالي:

Department

Id	Name
1	HR
2	Sales

```
Create table Emp (id number primary key, name varchar(30), dept foreign key references Department(id));
```

Emp

Id	Name	dept
1	A	1
2	B	2
3	C	2
4	D	1

القيم المتاحة للحقل dept في الجدول Emp هي 1 (HR) و 2 (Sales) و Null، وأي محاولة لإدخال قيمة أخرى ستفشل مع رسالة خطأ (تم خرق قيد تكامل مرجعي).

```
Insert into Emp values (5,'E',3);
```

خطأ – فشل في تطبيق التعليمة بسبب خرق قيد تكامل مرجعي.

الوحدانية Unique

تعني الوحدانية، عدم امكانية إدخال نفس القيمة للحقل أكثر من مرة:

```
Create table Student (s_id int Primary Key, Name varchar(60) Unique, Age int);
```

لا يمكن ادخال قيم مكررة للحقل Name، و يمكن ادخال Null في هذا الحقل.

اجباري Not Null

في حال كان الحقل اجباري، لا يمكن ادخال قيمة Null فيه بشكل صريح أو بشكل ضمني:

```
Create table Student (s_id int Primary Key, Name varchar(60) not null, Age int);

Insert into Student values (6,Null,2);
```

خطأ: لا يمكن ادخال Null للحقل بشكل صريح.

```
Insert into Student (id, age) values (6,2);
```

خطأ: لا يمكن ادخال Null للحقل بشكل ضمني.

3. القيم الافتراضية Default

يمكن تعريف قيمة افتراضية لأحد الحقول عند انشاء الجدول، وفي هذه الحالة يمكن ادخال قيمة صريحة في هذا الحقل وإلا فإن القيمة الافتراضية بدلاً من Null ستدخل في الحقل عند إضافة تسجيلة .

```
Create table Student (s_id int Primary Key, Name varchar(60), Age int default 30 );
```

S_id	Name	age

======

```
Insert into student(id, name) values(1,'x');
Insert into student values(2,'y',45);
```

S_id	Name	age
1	x	30
2	y	45

في التسجيلة الأولى تم ادخال القيمة الافتراضية (30) في حقل العمر، بدلاً من Null.

4. شروط التحقق Check:

يفيد شرط التحقق في وضع قيد على القيم المدخلة لأحد الحقول، كأن يكون ضمن مجال معين، أو أكبر من قيمة محددة.

```
Create table Student (s_id int Primary Key, Name varchar(60), Age int check (age>18) );
```

S_id	Name	age

```
Insert into student  values(1,'x',12);
```

خطأ – فشل في تطبيق التعليمة بسبب خرق قيد تكامل مرجعي.

5. التوابع في SQL

التابع هو عبارة عن تعبير رياضي يأخذ مجموعة من قيم الدخل التي ندعوها مُعاملات، ويعيد قيمة خرج وحيدة ندعوها قيمة التابع. تتعلق قيمة التابع (أي الخرج) بمُعاملاته (أي بالدخل)، كحال التابع الذي يقوم بحساب مجموع قيم عددية.

تصنّف توابع SQL إلى:

- **التوابع التجميعية** وهي التوابع التي تأخذ كمُعاملات مجموعة من القيم وتعيد قيمة وحيدة، مثل التابع الذي يحسب مجموع أعداد حقيقية.
- **التوابع الدرجية** وهي التوابع التي تأخذ مُعاملاً وحيداً وتُعيد قيمة وحيدة، مثل تابع القيمة المُطلقة لعدد حقيقي.

التوابع التجميعية

أهم التوابع التجميعية والأكثر استخداماً هي:

التابع	استخدامه
AVG	يقوم بحساب معدل القيم لحقل معين
COUNT	يقوم بحساب عدد البيانات الخاصة بحقل معين
MIN	يقوم بإعادة القيمة الصغرى من قيم حقل معين
MAX	يقوم بإعادة القيمة العظمى من قيم حقل معين
SUM	يقوم بحساب مجموع قيم حقل معين

فيما يلي أمثلة على استخدام التوابع التجميعية:

Employee

Id	Name	Location	Age	Salary
1	X	1	32	15000
2	Y	1	44	12000

Dept

Id	Name
1	HR
2	Sales

```
Select    avg(age)
from      Employee,Dept
where     Employee.location=Dept.id
          and Dept.name='Sales';
```

النتيجة معدل أعمار موظفي المبيعات.

```
Select Dept.name, Sum(Employee.salary)
from Employee,Dept
where     Employee.location=Dept.id
group by Dept.name
having     Count(Employee.id)>10;
```

النتيجة مجاميع رواتب الأقسام التي فيها أكثر من عشرة موظفين.

ملاحظات:

- التابع Avg يمكن أن يأخذ أحد دخلين بالإضافة إلى الحقل المراد حساب الوسطي له، الدخلين هما All وDistinct، ويفيد الثاني منهما باستبعاد القيم المكررة عند حساب الوسطي، أما الأول فهو الخيار الافتراضي ولا يتم فيه استبعاد القيم المكررة.
- التابع Count يأخذ إحدى ثلاثة قيم للدخل هي (*, All, Distict) بالإضافة لاسم الحقل، وتعني:
  - *: أي عدد التسجيلات بما فيها التسجيلات التي تحوي Null كقيمة للحقل اللذي نحسب مجموع ادخالاته.
  - All: تستبعد هنا القيم Null من العدد.
  - Distinct: نستبعد هنا القيم Null والقيم المكررة للحقل.

التوابع الدرجية

التوابع الدرجية هي أربعة أنواع:

التوابع الرقمية	وهي التوابع الخاصة بالعمليات على الأرقام، مثل تابع التقريب إلى أقرب فاصلة عشرية
توابع سلاسل المحارف	وهي التوابع الخاصة بالعمليات على سلاسل المحارف، مثل تابع تحديد طول سلسلة محرفية
توابع التاريخ والوقت	وهي التوابع الخاصة بالعمليات على التاريخ والوقت، مثل تابع حساب الزمن الفاصل بين تاريخين
توابع التحويل	هي التوابع الخاصة بعملية تحويل مُعامل الدخل، من نمط بيانات إلى آخر.

- التوابع الرقمية: وهي التوابع الدرجية الخاصة بالعمليات على القيم الرقمية ومن أهمها التوابع التالية:

Floor	وهو التابع الذي يُقرِّب مُعامل الدخل إلى أقرب عدد صحيح أصغر من مُعامل الدخل
Ceiling	وهو التابع الذي يُقرِّب مُعامل الدخل إلى أقرب عدد صحيح أكبر من مُعامل الدخل
Round	وهو التابع الذي يُقرِّب مُعامل الدخل ذو الفاصلة العشرية إلى أقرب عدد صحيح أو عدد حقيقي بدقة محددة
Abs	وهو التابع الذي يعيد القيمة المطلقة لمُعامل الدخل
Sin, Cos, Tan, Atan	وهي التوابع التي تحسب قيم ظل، تظل، جب، تجب الزاوية التي نأخذها كمُعامل دخل.
SQRT	هو التابع الذي يُعيد قيمة الجذر التربيعي لمُعامل الدخل
RAND	وهو التابع الذي يُعيد رقم عشوائي بين 0 و 1 و يستخدم مُعامل الدخل كأساس لتوليد الرقم العشوائي

- توابع سلاسل المحارف: هي التوابع الدرجية الخاصة بالعمليات على السلاسل المحرفية، ومن أهمها التوابع التالية:

Left()	وهو التابع الذي يُعيد جزء من السلسلة، يبتدئ من بدايتها حتى عدد محدد من المحارف
Right()	وهو التابع الذي يُعيد جزء من السلسلة، يبتدئ من نهايتها حتى عدد محدد من المحارف
Substr()	وهو التابع الذي يُعيد جزء من السلسلة، يبتدئ من موقع محدد فيها وبطول عدد محدد من المحارف
Length()	وهو التابع الذي يُعيد طول السلسلة المحرفية
Concat()	وهو التابع الذي يُستخدم لدمج أكثر من سلسلة محرفية
Lower() / Upper()	وهو التابع الذي يحول جميع محارف السلسلة إلى أحرف كبيرة أو صغيرة
Trim()	وهو التابع الذي يلغي الفراغات من بداية ونهاية السلسلة المحرفية
Instr()	وهو التابع الذي يُستخدم لتحديد موقع سلسلة جزئية ضمن سلسلة رئيسية

- توابع التاريخ والوقت: هي التوابع الخاصة بالعمليات على التاريخ والوقت ومن أهمها التوابع التالية:

التابع	**استخدامه**
DateDiff()	يُعيد الفرق بين تاريخين
GetDate()	يُعيد السنة، والتاريخ، واليوم، والساعة، والدقيقة، والثانية، وأجزاء الثانية
CURRENT_DATE	يعيد التاريخ الحالي الخاص بنظام إدارة قاعدة البيانات
CURRENT_TIME	يُعيد التوقيت الخاص بنظام إدارة قاعدة البيانات
CURRENT_TIMESTAMP	يُعيد التاريخ والتوقيت الخاصين بنظام إدارة قاعدة البيانات

- توابع التحويل: هي التوابع الخاصة بالتحويل من نمط بيانات إلى نمط آخر ومن أهمها التوابع التالية:

التابع	استخدامه
Str()	يُحول قيمة الدخل العددية إلى سلسلة محرفية.
To_Number()	يُحول سلسلة المحارف المارة كمُعامل دخل إلى عدد.
Cast()	يُحول قيمة الدخل إلى قيمة من أي نمط آخر من البيانات
Convert()	يُحول قيمة الدخل إلى قيمة من أي نمط آخر من البيانات

أمثلة:

Employee

Id	Name	Location	Age	Salary
1	X	HR	32	15000
2	Y	Sales	44	12000

```
Select Left (Name,1)
from      Employee;
```

النتيجة الحرف الأول من اسم كل موظف.

==========

```
Select Concat (name, location)
from      Employee;
```

النتيجة قائمة بأسماء الموظفين مدموجة مع أسماء الأقسام التي يعملون فيها.

6. الأسماء البديلة Aliases

قد تتضمن الجداول حقول متشابهة في الاسم، فمثلاً يمكن أن نجد حقل باسم Id في جدول Employee وفي جدول Dept (مثالنا السابق)، ولتمييز حقلين بنفس الاسم من جدولين مختلفين في نفس عبارة Select نستخدم الأسماء البديلة على الشكل التالي.

Employee

Id	Name	Location
1	X	1
2	Y	1

Dept

Id	Name
1	HR
2	Sales

```
Select Employee.id as e_id,Employee.name as e_name,
Dept.name as department
from Employee, Dept
where Employee.location=Dept.id;
=========
```

E_id	E_Name	Department
1	X	HR
2	Y	HR

يمكن أيضاً استخدام نفس الجدول في تعليمة Select أكثر من مرة (nested, correlated, self–join)، وفي هذه الحالة يجب تمييز ورودات الجدول عن بعضها باستخدام الأسماء البديلة، كما في المثال التالي:

Person

Id	Name	Gender	Age
1	X	1	32
2	Y	1	24
3	A	0	30
4	B	0	28

```
Select A.name as m1, B.name as m2
from Person A, Person B
where A.gender + B.Gender=1 and abs(A.age-B.age)<5;
======
```

النتيجة: هي الأزواج المحتملة من ذكور واناث بفارق عمر لا يزيد عن 5 سنوات

M1	M2
X	A
X	B
Y	B
A	X
B	X
B	y

نلاحظ أن الأزواج مكررة (ما الشرط اللذي يجب أن نضيفه لإلغاء التكرار؟)

المثال السابق[1] يدعى بالارتباط الذاتي (self-join)، الحالات الأخرى التي نضطر فيها لاستخدام أسماء بديلة للجداول هي حالات تداخل select (nested select statement)، وحالة select المترابطة (Correlated select statement)، وسنناقش الحالتين في الفقرات التالية.

7. تعليمة Select المتداخلة (nested select statement)

يمكن استخدام نتيجة تعليمة Select كمصدر للبيانات تبنى عليه تعليمة Select أخرى، تساعد هذه التقنية في تسهيل فهم وقراءة تعليمات Select إلا أنها ليست الخيار الأفضل في حال كان تنفيذ المطلوب بعبارة Select واحدة مع الضم والاختيار، وذلك لدواعي سرعة التنفيذ.

فيما يلي سنحلل كيفية حل مسألة باستخدام تعليمات Select متداخل، وسنرى أننا يمكن أن نحقق نفس النتيجة بتعليمة Select واحدة:

[1] `Select A.name as m1, B.name as m2`
`from Person A, Person B`
`where A.gender + B.Gender=1 and abs(A.age-B.age)<5;`

Company

Id	Name	City
401	x	Damascus

Product

Id	Name	Producer
1	x	401

Order

Id	Product	Client
10	1	John
20	1	Joe

المطلوب هو المدن التي فيها شركات تنتج مواد تم طلبها من قبل John.

المواد التي طلبها John هي:

```
Select Product.id
from Product, Order
where Product.id=Order.product and Order.Client='John';
```

باستبدال أرقام المواد بأرقام مصنعيها، يبقى أن نختار اسم كل شركات رقمها ضمن قائمة أرقام المصنعين التي نتجت معنا.

```
Select City from Company where Company.Id in
 (
    Select Product.Producer
    from Product, Order
    where          Product.id=Order.product          and
Order.Client='John'
);
==========
```

نفس النتيجة يمكن تحقيقها بتعليمة Select واحدة هي:

```
Select    Company.City
from      Product, Order, Company
where     Product.id=Order.product
      and Product.Producer=Company.id
     and Order.Client='John';
```

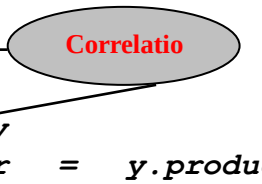
8. تعليمات Select المترابطة (Correlated select statement):

Product

Name	Price	Category	Producer	Year

المطلوب قائمة بالمنتجات الأغلى من أية منتجات أنتجها نفس المصنّع قبل عام 2014

```
SELECT DISTINCT  name, producer
FROM     Product AS x
WHERE  price > ALL  (SELECT  price
                     FROM    Product AS y
                     WHERE     x.producer   =   y.producer   AND
y.year < 2014);
```



9. تطبيق:

في المثال التالي سنعرض بعض الاستخدامات الخاصة لتعليمة Select:

Person

Id	FName	LName	Gender	Nationality	Age
1	John	Smith	1	2	32
2	Adam	Sandler	1	1	45
3	Mary	Clair	0	1	40
…	…	…	…	…	…

=======

Nationality

Id	Name
1	American
2	Frensh
…	…

========

ContactMedia

Id	Media
1	Mobile
2	E-Mail
3	Phone
…	…

=======

PersonContact

Id	Person	ContactMedia	ContactValue
1	1	1	00963944111222
2	1	2	xxx@gmail.com
…	…	…	…

=========

في مثالنا الجنسية Nationality تصدر مفتاحها إلى الجدول Person (ارتباط واحد لعدة)، وطريقة الاتصال ContactMedia ترتبط مع جدول Person من خلال جدول وسيط هو PersonContact (يمكن أن يكون لنفس الشخص أكثر من طريقة للاتصال به).

المطلوب:

- قائمة بفرق العمل المحتملة، علماً أن كل فريق يتألف من شخصين من جنسية واحدة و فارق العمر بينهما لا يتجاوز 5 سنوات ؟
- قائمة بالجنسيات التي يحملها على الأقل شخصين مسجلين في قاعدة البيانات ؟
- وسيلة الاتصال الأكثر استخداماً (المدخل بياناتها للأشخاص المسجلين) ؟
- وسطي أعمار الأشخاص المسجلين في كل جنسية ؟

10. المراجع:

- http://www.studytonight.com/dbms/select-query

11. تدريبات:

Invoice

Id	Serial	Date	Client
1	1	1-1-2015	20
2	2	2-1-2015	30
3	3	1-1-2015	10
4	4	4-2-2015	20
5	5	4-2-2015	10

Client

Id	Name
10	C 1
20	C 2
30	C 3

Material

Id	Name	Price
1	Printer	15000
2	Monitor	30000
3	Case	2500
4	MB	22000

Invoice_Item

Id	Invoice	Material	QTY
1	1	1	1
2	1	2	1
3	1	4	1
4	2	2	2
5	2	3	2

- نتيجة تطبيق جملة التعليمات التالية:

```
Select      Id,Name
From        Material
Where       Id not in   ( Select material from invoice, invoice_item
                        where invoice.id=invoice_item.invoice
                        and invoice.date <>'1-1-2015');
```

هي:

1. المواد التي تم شراؤها في 1-1-2015.
2. المواد التي لم يتم شراء كميات منها.
3. المواد التي لم يتم شراء أي كمية منها بتاريخ 1-1-2015.
4. المواد المتضمنة في الفاتورة رقم 1 و الفاتورة رقم 3.

الإجابة: (3)

- نتيجة التعليمة التالية:

```
Select      Id,Name
From        Material
Where       Price >  (  Select Max(price) from material, invoice_item
                        where invoice_item.material=material.id
                        );
```

هي:

1. المادة الأغلى سعراً.
2. المواد التي سعرها أغلى من سعر أية مادة ظهرت في إحدى الفواتير.
3. المادة المباعة و الأعلى سعراً.
4. المواد التي لم يتم البيع منها.

الإجابة: (2)

- نتيجة التعليمة التالية:

```
SELECT      Name, Sum(price*qty) AS TotalSales
FROM        Material, Invoice,Invoice_item
WHERE
            Material.id=Invoice_item.material
            And invoice_item.invoice=invoice.id
            And date > '1/1/2015'
GROUP BY    name
```

هي:

1. اجمالي الفواتير بعد تاريخ 1-1-2015
2. إجمالي المبيعات حسب المادة بعد تاريخ 1-1-2015.
3. مجموع أسعار المواد المباعة بتاريخ 1-1-2015
4. مبيعات المواد مجمعة حسب التاريخ.

الإجابة: (2)

الفصل الحادي عشر:
إدارة المناقلات والتحكم المتزامن

العنوان	الصفحة
1. مقدمة	3
2. المناقلات	3
3. خصائص المناقلات (ACID)	5
1.3 الكتليّة Atomicity	5
2.3 التسلسلية Consistency	5
3.3 العزل Isolation	5
4.3 الاستمرارية Durability	5
4. سجل المناقلات	7
5. التحكم المتزامن	8
1.5 التعديلات الضائعة	8
2.5 المعطيات غير المؤكدة	9
3.5 الاسترجاعات غير المتسقة	10
6. المجدول	12
7. التحكم المتزامن بالمناقلات (خوارزميات الإقفال)	13
1.7 مستويات الإقفال	13
2.7 أنماط الإقفال	16
3.7 أنظمة القفل على مرحلتين	17
4.7 مشكلة الإقفال المتبادل	18
5.7 الأختام الزمنية	19
8. إدارة التعافي في قواعد المعطيات	20
9. المراجع	22

الكلمات المفتاحية:

مناقلة، اتّساق المعطيات، الكتليّة، الاستمرارية، التسلسلية، العزل، تأكيد التغيرات COMMIT، التراجع عن التغيرات ROLLBACK، سجل المناقلات، التحكم المتزامن، التنفيذ المتزامن، مشكلة التعديلات الضائعة، مشكلة المعطيات غير المؤكّدة، مشكلة الاسترجاعات غير المتّسقة، خوارزميات القفل، طرائق الأختام الزمنية، قفل ثنائي، قفل مشترك، قفل حصري، القفل على مستوى قاعدة المعطيات، القفل على مستوى الجدول، القفل على مستوى الصفحة، القفل على مستوى السطر، القفل على مستوى الحقل، بروتوكول الإقفال على مرحلتين، تضارب الأقفال، الإقفال المتبادل، العناق المميت، الاستبعاد المتبادل، منع حصول الإقفال المتبادل، اكتشاف الإقفال المتبادل، تجنب الإقفال المتبادل، إدارة التعافي، بروتوكول كتابة سجل المناقلات أولاً، سجلات مناقلات مكررة، خازنات قواعد المعطيات، الكتابة المؤجلة، التعديل المؤجل، الكتابة المباشرة، التعديل المباشر.

ملخص:

يتم في هذا الفصل التعرف على المناقلات وكيف تتم إدارة عملية المزامنة فيما بينها في أنظمة إدارة قواعد المعطيات.

الأهداف تعليمية:

يهدف هذا الفصل التعريف بالمفاهيم التالية:

- المناقلات وخصائصها
- كيف تتم إدارة مناقلات قواعد المعطيات
- ما هو التحكم المتزامن وما هو الدور الذي يلعبه في تحقيق تكامل قواعد المعطيات
- ما هي طرائق القفل وكيف يعمل
- كيف تتم إدارة التعافي في صيانة قواعد المعطيات

1. مقدمة

في هذا الفص سنعرض حالات متقدمة لاستخدام تعليمة Select موضوع الفصل السابق، وتتمات أساسية في لغة الاستعلام المهيك تعكس مناقلات قواعد المعطيات، الإجراءات الحقيقية التي تجري من حولنا كشراء منتجات أو تسجيل معلومات شخصية أو إيداع أو سحب من حساب في بنك معين.

تتكون المناقلات من أجزاء مختلفة، فإجراء مناقلة بيع على سبيل المثال يمكن أن يؤدي إلى إجراء مناقلة تحديث لحساب الزبون وأخرى لتعديل موجودات المستودع وثالثة لتعديل حساب البائع بزيادة قيمة المنتج المباع.

ينبغي أن تنجح كافة أجزاء المناقلة لكي يتم ضمان صحة وسلامة وتكامل المعطيات المخزنة، وبالتالي تعد نظم إدارة وتنفيذ المناقلات من أهم نظم إدارة قواعد المعطيات.

2. المناقلات

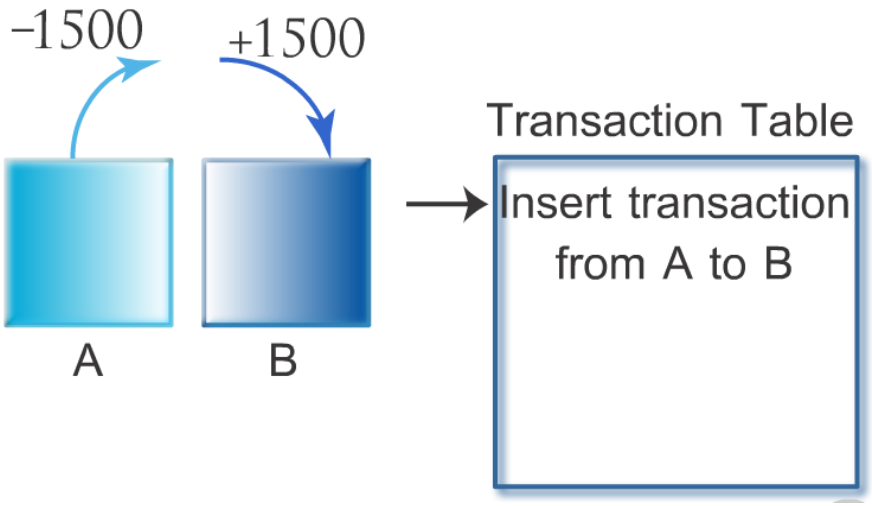
هي كل فعل يؤدي إلى قراءة و/أو كتابة في قاعدة المعطيات، بحيث يمكن أن تتكون المناقلة من استعلام بسيط أو من مجموعة عبارات لتحديث المعطيات في جدول معين أو من مجموعة معطيات أخرى لإضافة المعطيات إلى جدول ما، أو من مزيج من العبارات السابقة.
والمناقلة عبارة عن وحدة عمل منطقية، إما أن تتم بالكامل أو أن تُلغى بالكامل.

والمناقلة الناجحة هي التي تنقل قاعدة المعطيات من حالة متّسقة إلى حالة متّسقة أخرى، –بحيث يقال عن قاعدة المعطيات بأنها متّسقة إذا ما كانت تحقق كافة شروط تكامل المعطيات– بالتالي، ولضمان الحفاظ على اتّساق قاعدة المعطيات ينبغي على المناقلة أن تنطلق اعتماداً على حالة أوليّة متّسقة.

يطلق على كل استعلام بسيط بلغة الاستعلام القياسية SQL اسم طلب من قاعدة المعطيات، فالمناقلة التي تتكون من عمليتي تحديث وعملية إدخال تتكون بالتالي من ثلاثة طلبات لقاعدة المعطيات، ويولد كل طلب قاعدة معطيات العديد من عمليات الدخل/خرج والتي تقرأ أو تكتب على القرص الصلب.

المثال التالي هو مناقلة تتألف من ثلاث طلبات على قاعدة المعطيات:

- تسجيل حركة نقل مبلغ 15000 من حساب A إلى حساب B
- حسم مبلغ 15000 من حساب A
- اضافة مبلغ 15000 إلى حساب B



Account

Id	Name	balance
1	A	50000
2	B	10000
3	C	75000

Trans

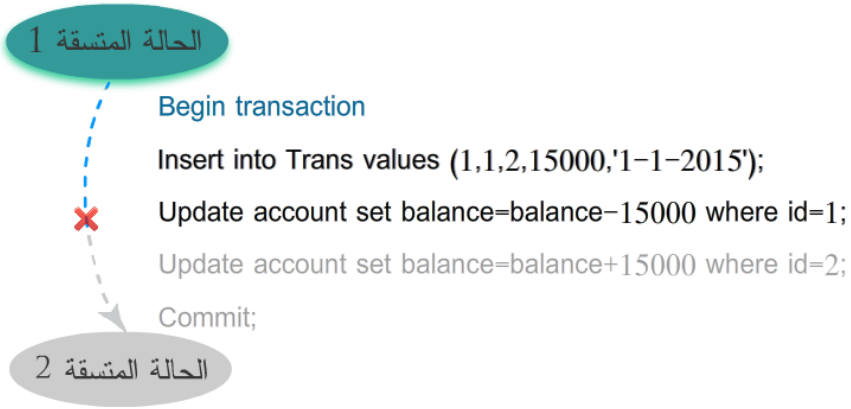
Id	Acc_from	Acc_to	Sm	Date
1	1	2	15000	1-1-2015

```
Begin transaction
Insert into Trans values (1,1,2,15000,'1-1-2015');
Update account set balance=balance-15000 where id=1;
Update account set balance=balance+15000 where id=2;
Commit;
```

تعريف الطلبات الثلاث السابقة كمناقلة واحدة، يضمن نقل قاعدة البيانات من حالة متسقة إلى حالة متسقة أخرى، ولفهم ذلك سنفرض أن خطأً ما حدث بعد تنفيذ الطلب الثاني (حسم من حساب A)، ونناقش الحالتين (وضع الطلبات ضمن مناقلة واحدة، تنفيذ الطلبات الثلاث على التوالي دون تعريف مناقلة).

الحالة الأولى (الطلبات الثلاث موضوعة ضمن مناقلة واحدة):

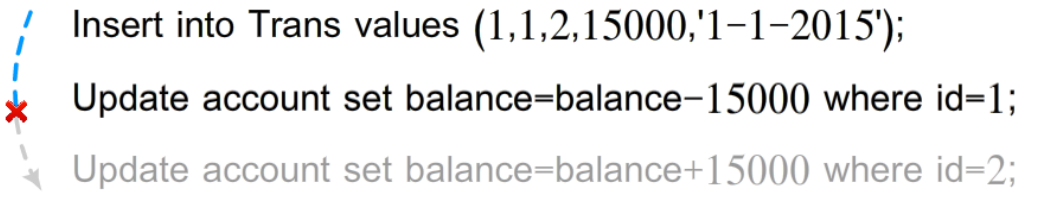
في هذه الحالة سيتم تسجيل الحركة في جدول Trans، ومن ثم سيتم حسم 15000 من حساب A، وبما أن الخطأ حصل قبل الوصول إلى نهاية المناقلة، يجري التراجع (Rollback) عن العمليتين السابقتين لتعود قاعدة المعطيات إلى حالتها المتسقة الأولى.



الحالة الثانية (الطلبات الثلاث تنفذ على التوالي دون مناقلة):

في هذه الحالة سيتم تسجيل الحركة في جدول Trans، ومن ثم سيتم حسم 15000 من حساب A، ولن يضاف مبلغ 15000 إلى حساب B، بسبب حدوث الخطأ (انقطاع التيار الكهربائي مثلاً)، بمعنى أن المبلغ حسم من حساب دون أن يضاف إلى حساب آخر، وهو ما يعكس خطأً في البيانات المسجلة في قاعدة المعطيات.

في الحالة الأولى مجموع أرصدة الحسابات هو 135000 قبل تنفيذ المناقلة وبعدها، بينما في الحالة الثانية سيصبح مجموع أرصدة الحسابات 120000 علماً أن الحركة التي كان من المفترض أن تتم هي نقل مبلغ من حساب لآخر، والطبيعي أن لا يتغير مجموع أرصدة الحسابات.



<table>
<tr>
<td rowspan="2">باستخدام المناقلة</td>
<td>
Account
<table>
<tr><th>Id</th><th>Name</th><th>balance</th></tr>
<tr><td>1</td><td>A</td><td>50000</td></tr>
<tr><td>2</td><td>B</td><td>10000</td></tr>
<tr><td>3</td><td>C</td><td>75000</td></tr>
</table>
الحالة المتسقة الأولى
</td>
<td>
Account
<table>
<tr><th>Id</th><th>Name</th><th>balance</th></tr>
<tr><td>1</td><td>A</td><td>35000</td></tr>
<tr><td>2</td><td>B</td><td>25000</td></tr>
<tr><td>3</td><td>C</td><td>75000</td></tr>
</table>
الحالة المتسقة الثانية
</td>
</tr>
<tr>
<td>من دون استخدام مناقلة</td>
<td>
Account
<table>
<tr><th>Id</th><th>Name</th><th>balance</th></tr>
<tr><td>1</td><td>A</td><td>50000</td></tr>
<tr><td>2</td><td>B</td><td>10000</td></tr>
<tr><td>3</td><td>C</td><td>75000</td></tr>
</table>
الحالة الأولى
</td>
<td>
Account
<table>
<tr><th>Id</th><th>Name</th><th>balance</th></tr>
<tr><td>1</td><td>A</td><td>35000</td></tr>
<tr><td>2</td><td>B</td><td>10000</td></tr>
<tr><td>3</td><td>C</td><td>75000</td></tr>
</table>
الحالة الثانية
</td>
</tr>
</table>

3. خصائص المناقلات (ACID)

الكتليّة Atomicity

وتعني وجوب تنفيذ كافة أجزاء المناقلة –أي طلبات قاعدة المعطيات– ككتلة واحدة، فإذا لم يتم تنفيذها بالكامل ينبغي التراجع عنها بالكامل، –أو بطريقة أخرى– ينبغي التعامل مع المناقلة على أنها كتلة عمل منطقية وحيدة وغير قابلة للتجزئة.

التسلسلية Consistency

وتعني استمرار اتّساق قاعدة المعطيات بعد تنفيذ عدّة مناقلات متزامنة، خاصةً في قواعد المعطيات الموزّعة أو متعددة المستخدمين.

العزل Isolation

أي لا يمكن استخدام معطيات من قبل مناقلة إذا ما كان هناك مناقلة أخرى تستخدم تلك المعطيات.

الاستمرارية Durability

وتعني استمرارية اتّساق قاعدة المعطيات، أي أن قاعدة المعطيات ينبغي أن تنتقل بعد تنفيذ المناقلة من حالة متّسقة إلى حالة متّسقة أخرى لا يمكن فقدانها حتى ولو فشل النظام ككل.

نلاحظ مما سبق أن خاصتي التسلسلية والعزل محققتان في نظام وحيد المستخدم، لأن المناقلات يتم تنفيذها كل على حده، في حين ينبغي ضمان الكتليّة والاستمرارية.

4. سجل المناقلات

تنقل المناقلات قاعدة المعطيات من حالة متسقة لأخرى، باستخدام مفهومي تأكيد التعليمات (Commit)، والتراجع للحالة البدئية (Rollback).

وتستخدم أنظمة إدارة قواعد المعطيات سجل المناقلات لكي تتبع مسير كافة العمليات التي تقوم بها المناقلات والتي تتغيّر فيها حالة قاعدة المعطيات، مما يساعد في استرجاع حالات متّسقة.

يحتوي السجل على المعلومات التالية:

- إشارة إلى بداية المناقلة.
- تفصيل حول كل طلب قاعدة معطيات محتوى في المناقلة.
- نوع العملية المنفذة (إضافة، حذف، تعديل)
- اسم الغرض المتأثر بتلك العملية (الجدول)
- القيم السابقة واللاحقة للحقول المتغيرة
- مؤشرات إلى القيم السابقة واللاحقة، أي مداخل سجلات المناقلة
- إشارة إلى نهاية المناقلة

ويتميز سجل المناقلات بما يلي:

- بناء واستخدام السجل يمكن أن يزيد من العبء المفروض على نظام إدارة قواعد المعطيات، ولكن يمكن تخطي ذلك مقارنةً بالمنفعة التي يمكن الحصول عليها منه.
- يعتبر السجل بحد ذاته، قاعدة معطيات يتم إدارتها من قبل نظام إدارة قواعد المعطيات كأية قاعدة معطيات أخرى، وهو معرّض بالتالي إلى كافة الأخطار التي تتعرض لها قواعد المعطيات كفشل القرص الصلب أو مشاكل تتعلق بالمساحة وامتلاء سعة القرص.
- يقوم نظام إدارة قواعد المعطيات باستدعاء تعليمة ROLLBACK على بعض أجزاء السجل أي على المناقلات غير المنتهية.

فيما يلي عرض للسجل الذي يعبر عن المناقلة التي تم تحويل 15000 من الحساب A إلى الحساب B:

رقم السطر	رقم المناقلة	السابق	التالي	العملية	الجدول	معرف السطر	الواصفة	القيمة السابقة	القيمة الجديدة
341	50	NULL	352	START	*بداية مناقلة	–	–	–	–
352	50	341	363	اضافة	حركة	1	Id, acc_from, Acc_to, sm, Date		1, 1, 2, 15000, 1-1-2015
363	50	352	365	تعديل	حساب	1	balance	50000	35000
365	50	363	366	تعديل	حساب	2	balance	10000	25000
366	50	365	NULL	COMMIT	*نهاية مناقلة	–	–	–	–

5. التحكم المتزامن

يُعبر التحكم المتزامن عن الإدارة والإشراف على التنفيذ المتزامن للمناقلات في نظام إدارة قواعد معطيات متعدد المستخدمين. وهو ضمان لتحقق خاصية تسلسلية المناقلات في بيئة قواعد معطيات متعددة المستخدمين.

هناك ثلاثة أنواع رئيسية من المشاكل التي يمكن أن تحدث نتيجةً للتنفيذ المتزامن للمناقلات وهي:

مشكلة التعديلات الضائعة ومشكلة المعطيات غير المؤكّدة ومشكلة الاسترجاعات غير المتّسقة.

سنقوم فيما يلي بشرح كل مشكلة من المشاكل السابقة بالتفصيل.

التعديلات الضائعة

تحصل مشكلة التعديلات الضائعة عندما تقوم مناقلتان بإجراء مجموعة من التعديلات على نفس الحقل في قاعدة المعطيات بشكل متزامن.

مثال: لنفترض وجود المناقلتين التاليتين:

T1: تقوم بشراء 100 وحدة من منتج معين– T2: تقوم ببيع 30 وحدة من نفس المنتج.

وأن جدول المنتج يحتوي على 35 وحدة من ذلك المنتج حالياً،

فإن التنفيذ المنطقي سيكون وفق المخطط التالي: ولكن لنفترض أن التنفيذ السابق للمناقلتين قد تم كما يلي:

الزمن	المناقلة	الخطوة	قيمة الحقل "كمية"
1	T1	قراءة الحقل "كمية"	35
2	T1	إضافة 100 وحدة	–
3	T1	كتابة النتيجة	135
4	T2	قراءة الحقل "كمية"	135
55	T2	حذف 30 وحدة	–
6	T2	كتابة النتيجة	105

الزمن	المناقلة	الخطوة	قيمة الحقل "كمية"
1	T1	قراءة الحقل "كمية"	35
2	T2	قراءة الحقل "كمية"	35
3	T1	إضافة 100 وحدة	–
4	T2	حذف 30 وحدة	–
5	T1	كتابة النتيجة	135
6	T2	كتابة النتيجة	5

نلاحظ أنه في الخطوة رقم /5/ من الجدول السابق قد حصلت مشكلة تتعلق بالتزامن بين المناقلتين T1 و T2 وكانت النتيجة ضياع في التعديلات.

المعطيات غير المؤكدة

تحصل ظاهرة المعطيات غير المؤكّدة عندما تُنفّذ مناقلتان بشكل متزامن بحيث تقوم الأولى بالتراجع عن التنفيذ بعد أن تقوم الثانية بقراءة المعطيات المكتوبة من قبل الأولى، مما يهدد خاصة العزل التي يجب أن تتحقق في المناقلات.

مثال:

لنحاول تطبيق نفس المثال السابق لنشرح من خلاله كيف يمكن أن تقع هذه المشكلة:

T1: تقوم بشراء 100 وحدة من منتج معين– T2: تقوم ببيع 30 وحدة من نفس المنتج.

وجدول المنتج يحتوي على 35 وحدة من ذلك المنتج حالياً.

إن التنفيذ المنطقي سيكون وفق المخطط التالي:

الزمن	المناقلة	الخطوة	قيمة الحقل "كمية"
1	T1	قراءة الحقل "كمية"	35
2	T1	إضافة 100 وحدة	–
3	T1	كتابة النتيجة	135
4	T1	ROLLBACK	35
5	T2	قراءة الحقل "كمية"	35
6	T2	حذف 30 وحدة	–
7	T2	كتابة النتيجة	5

ولكن لنفترض أن التنفيذ السابق للمناقلتين قد تم كما يلي:

الزمن	المناقلة	الخطوة	قيمة الحقل "كمية"
1	T1	قراءة الحقل "كمية"	35
2	T1	إضافة 100 وحدة	–
3	T1	كتابة النتيجة	135
4	T2	قراءة الحقل "كمية"	135
5	T2	حذف 30 وحدة	–
6	T1	ROLLBACK	35
7	T2	كتابة النتيجة	105

نلاحظ أنه في الخطوة رقم /4/ من الجدول السابق قد حصلت مشكلة تتعلق بالتزامن بين المناقلتين T1 و T2 وكانت النتيجة قراءة معطيات غير مؤكّدة.

الاسترجاعات غير المتسقة

تحصل ظاهرة الاسترجاعات غير المتّسقة عندما تقوم مناقلة ما بتطبيق تابع تجميعي معين على مجموعة معطيات بينما تقوم مناقلات أخرى بتحديث تلك المعطيات، أي يمكن أن تقوم مناقلة بقراءة معطيات قبل أو بعد أن تتغير وبالتالي تنتج حالات غير متّسقة.

مثال:

لنفترض المناقلة T1 تقوم بحساب مجموع كميات كافة المنتجات الموجودة في الجدول "منتج"، بينما تقوم المناقلة T2 بنفس الوقت بتحديث كمية منتجين محددين من ذلك الجدول.

لنفترض الآن وجود القيم التالية في جدول "منتج":

رقم المنتج	"الكمية" قبل التعديل	"الكمية" بعد التعديل
11	8	8
13	32	32
46	15	15+10=25
58	23	23−10=13
17	8	8
18	6	6
المجموع	92	92

نلاحظ من الجدول أن المجموع قبل التعديل يساوي 92، ولكن تحدث مناقلتان في المعطيات بحيث تضيف إحداهما 10 وحدات إلى المنتج 46 وتنقص الأخرى 10 وحدات أيضاً من المنتج 58، فبالتالي ينبغي أن يكون المجموع مساوياً للقيمة 92 أيضاً.

مثال:

لنفترض الآن السيناريو التالي:

الزمن	المناقلة	الخطوة	القيمة	المجموع
1	T1	قراءة كمية المنتج 11	8	8
2	T1	قراءة كمية المنتج 13	32	40
3	T2	قراءة كمية المنتج 46	15	
4	T2	إضافة 10 وحدات	–	
5	T2	كتابة القيمة الجديدة	25	
6	T1	قراءة كمية المنتج 46	25	65
7	T1	قراءة كمية المنتج 58	23	88
8	T2	قراءة كمية المنتج 58	23	
9	T2	حذف 10 وحدات	–	
10	T2	كتابة القيمة الجديدة	13	
11	T2	COMMIT		
12	T1	قراءة كمية المنتج 17	8	96
13	T1	قراءة كمية المنتج 18	6	102

نلاحظ في الخطوة رقم /7/ من الجدول السابق أن المناقلة T1 التي تقوم بتطبيق التابع التجميعي قد قرأت قيمة كمية المنتج 58 قبل التحديث، ثم من الخطوة رقم /8/ إلى الخطوة رقم /11/ تتم عملية تحديث قيمة المنتج 58 من قبل المناقلة T2 وتخزين القيمة الجديدة، مما يؤدي إلى حالة عدم اتّساق في قاعدة المعطيات، لأن نتيجة تنفيذ المناقلة T1 –أي نتيجة تنفيذ التابع التجميعي– مخالفة لمحتويات قاعدة المعطيات الفعلية.

6. المجدول

- يمكن أن تنشأ العديد من المشاكل عندما يتم تنفيذ عدّة مناقلات بشكل متزامن
- تنقل عدّة مناقلات قاعدة المعطيات من حالة متّسقة إلى أخرى
- يمكن أن نضمن اتّساق قاعدة المعطيات فقط قبل وبعد تنفيذ المناقلات
- بما أنه لا يمكن على الحاسب أن يقوم بتنفيذ عدّة تعليمات في آنٍ واحد، فإنه لابد من تنفيذ المناقلات بشكل متسلسل، مما قد يؤدي إلى حرمان بعض المناقلات من الحصول على معطيات تمتلكها مناقلات أخرى
- تظهر مشكلة التضارب بين المناقلات بوضوح عندما تقوم عدّة مناقلات باستخدام نفس المعطيات، عندئذٍ ينبغي حل تلك المشكلة من خلال تحديد ترتيب معين يسمح لكل المناقلات باستخدام المعطيات بشكل سليم.

يوضح الشكل التالي كيف يمكن أن يحدث التضارب في المعطيات فيما بين المناقلات:

العملية	المناقلات		النتيجة
	T2	T1	
	قراءة	قراءة	لا يوجد تضارب
	قراءة	كتابة	تضارب
	كتابة	قراءة	تضارب
	كتابة	كتابة	تضارب

يعالج نظام إدارة قواعد المعطيات تلك المشكلة من خلال برنامج خاص يقوم بإدارة ترتيب تنفيذ المناقلات المتزامنة بحيث يضمن تحقق تسلسل وعزل المناقلات وذلك بالاعتماد على خوارزميات وطرائق خاصة كخوارزميات القفل أو طرائق الأختام الزمنية.

7. التحكم المتزامن بالمناقلات (خوارزميات الإقفال)

- يضمن الإقفال الاستخدام الحصري للمعطيات من قبل مناقلة معينة
- يمكن أن تقوم مناقلة ما بقفل معطيات معينة بحيث تمنع مناقلات أخرى من الحصول على تلك المعطيات إلى أن يتم تحرير القفل من قبل المناقلة التي وضعته
- تتم إدارة الأقفال من خلال برنامج خاص يتولّى منح القفل أو استرداده من المناقلات
- يمكن أن يتم الإقفال على عدّة مستويات وهي: قاعدة المعطيات، الجدول، الصفحة، السطر، الحقل
- هناك نوعان أساسيان من الأقفال هما: الأقفال الثنائية والأقفال المشتركة/الحصرية

مستويات الإقفال:

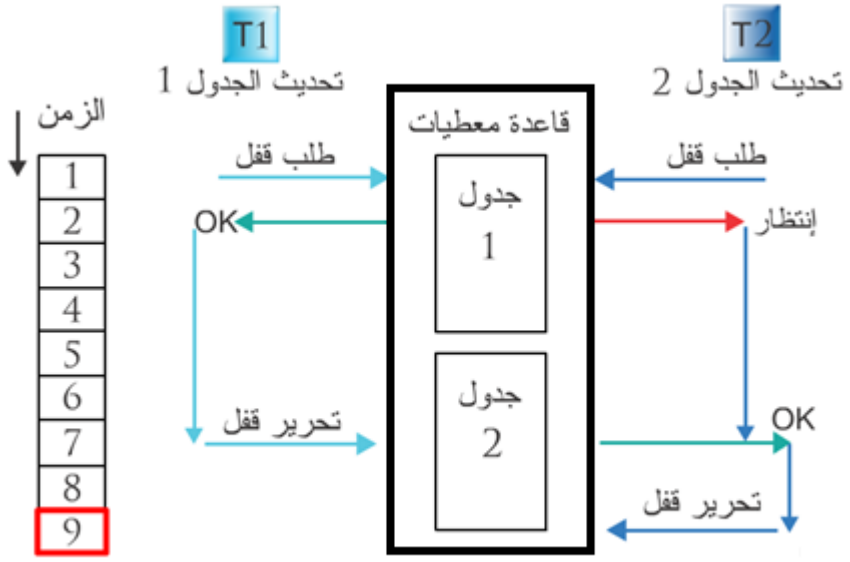
يمكن أن يتم الإقفال على عدّة مستويات وهي: قاعدة المعطيات، الجدول، الصفحة، السطر، الحقل.

القفل على مستوى قاعدة المعطيات

يتم في هذا المستوى قفل قاعدة المعطيات ككل مع كافة جداولها ومنحها لمناقلة معينة في حين تمنع كافة المناقلات الأخرى من الولوج إلى قاعدة المعطيات ريثما يتم تحرير القفل.

يمكن هنا أن نستنتج عدم فعالية هذا النوع من مستويات القفل في نظم إدارة قواعد المعطيات متعددة المستخدمين، وخاصة إذا ما أخذنا بعين الاعتبار البطء الشديد الذي يمكن أن ينتج كلما ازدادت المناقلات.

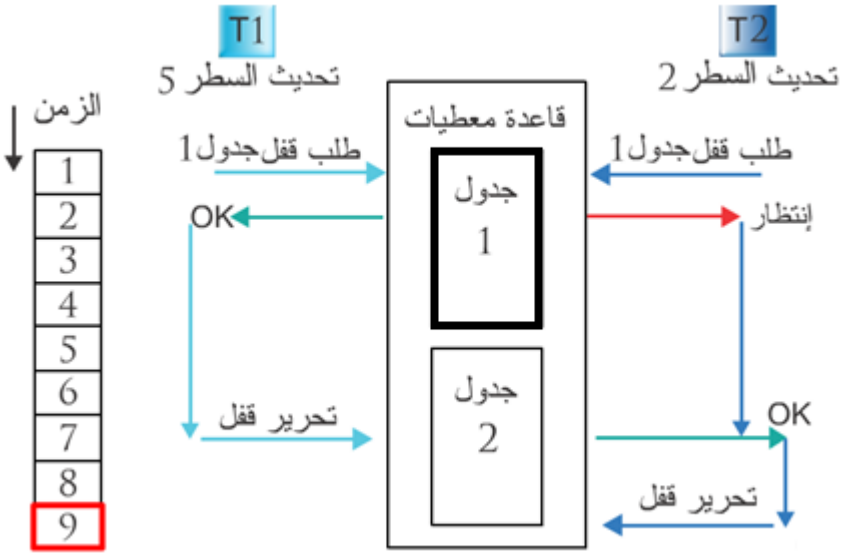
يوضح الشكل التالي كيف يتم إجراء القفل على مستوى قاعدة المعطيات:



القفل على مستوى الجدول

يتم في هذا المستوى قفل جدول بالكامل بحيث تُمنع مناقلة ما من دخول ذلك الجدول إذا ما كانت هناك مناقلة أخرى تستخدم معطيات منه.

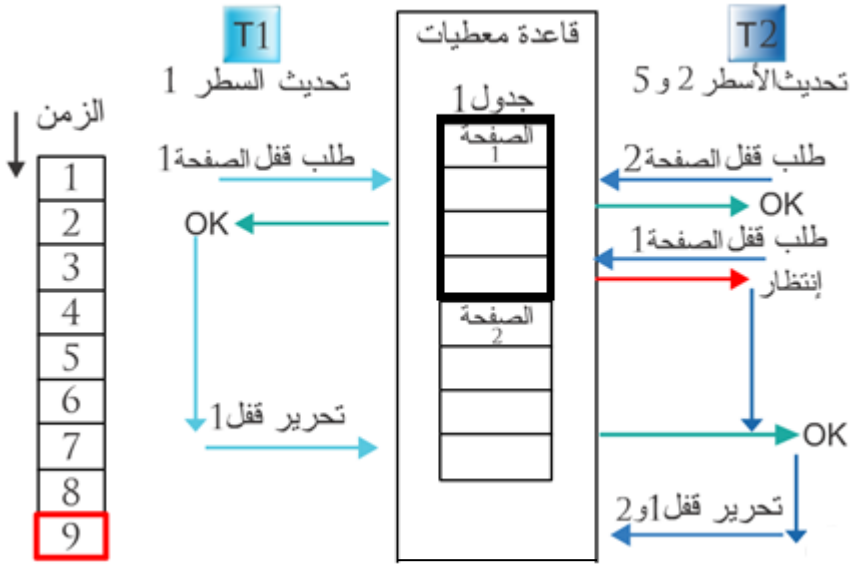
نلاحظ أن هذا النوع من الأقفال يعد أكثر أريحية من المستوى السابق ولكن تبقى نفس المشكلة، أي فيما إذا كان هناك العديد من المناقلات التي تحاول ولوج نفس الجدول في آن واحد.
يوضح الشكل التالي كيف يتم إجراء القفل على مستوى الجدول:



القفل على مستوى الصفحة

يقصد بالصفحة الكتلة من القرص، يتم في هذا المستوى قفل الصفحة بالكامل عندما يتم طلبها من مناقلة قبل معينة.
يعتبر هذا النوع من الأقفال مناسباً لبيئة نظم إدارة قواعد المعطيات متعددة المستخدمين، ولكن تتمثل سيئاته بأن الصفحة يمكن أن تحتوي على العديد من الأسطر، وبالتالي فإن قفل كل الصفحة يمكن أن يؤدي إلى حرمان مناقلات أخرى من أسطر معينة لا يتم استخدامها من قبل المناقلة التي طلبت القفل.

يوضح الشكل التالي كيف يتم إجراء القفل على مستوى الصفحة:

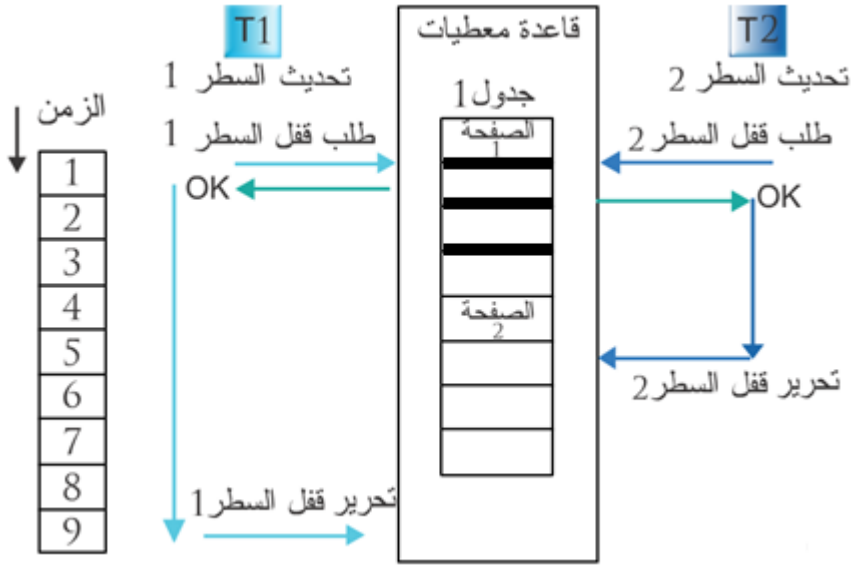


القفل على مستوى السطر

وهو أسلوب أقل تقييداً من المستويات السابقة في القفل، بحيث يسمح لعدّة مناقلات بالولوج إلى عدّة أسطر مختلفة في جدول معين حتى ولو كانت تلك الأسطر في نفس الصفحة.

تبرز سيئات هذه الطريقة من خلال العبء المتولد نتيجةً لكثرة المعالجة المفروضة على كل سطر من أسطر الجداول.

يوضح الشكل التالي كيف يتم إجراء القفل على مستوى السطر:



القفل على مستوى الحقل

وهو أكثر أنواع مستويات القفل ملاءمة للبيئة متعددة المستخدمين ولكن –على الرغم من ذلك– نادراً ما يتم استخدام هذه الطريقة وذلك للعبء الكبير جداً المتولد على الحاسب نتيجةً لتطبيقها.

أنماط الإقفال

هناك نوعان أساسيان من الأقفال التي تستخدمها نظم إدارة قواعد المعطيات، وهي:
الأقفال الثنائية والأقفال المشتركة/الحصرية

الأقفال الثنائية

1. يتميز القفل الثنائي بحالتين وحيدتين هما: مقفول (1) و غير مقفول (0).
2. ما أن يتم قفل غرض معين سواء كان قاعدة معطيات أو جدول أو صفحة أو سطر أو حتى حقل من قبل مناقلة معينة، فإنه لا يمكن أبداً لأية مناقلة أخرى أن تستخدم ذلك الغرض ما لم تقم الأولى بتحرير قفله
3. ينبغي بالتالي أن تقوم كل مناقلة بقفل أو حجز كافة الموارد أو الأغراض التي تريد استخدامها، وهذه العملية تتم على عاتق نظام إدارة قواعد المعطيات
4. يمكن أن نلاحظ سيئة في هذا النمط من الأقفال تكمن في كونها شديدة جداً ولا تدعم التنفيذ المتزامن، فعلى سبيل المثال لنفترض وجود مناقلتين تقومان بقراءة غرض معين، بالتالي –وفي ظل استخدام نمط الأقفال الثنائية– سوف لن يسمح نظام إدارة قواعد المعطيات للمناقلتين بقراءة الغرض بنفس الوقت حتى ولو لم تقم أياً منهما بإجراء أية تعديلات على ذلك الغرض.

الأقفال المشتركة/الحصرية

- يُعبر القفل الحصري عن الحالة التي ينبغي فيها منح حقوق حصرية لمناقلة معينة بالولوج إلى غرض ما دون غيرها من المناقلات، أو في الحالات التي يحتمل حدوث تضارب بالمعطيات.
- يُعبر القفل المشترك عن الحالة التي يمكن فيها لمناقلتين أن تلجأ إلى معطيات مشتركة للقراءة فقط، بحيث لا يمكن أن تولد عمليات قراءة المعطيات أي تضارب فيما بين المناقلات.
- يمنح القفل المشترك على العنصر X إلى أية مناقلة تريد قراءة ذلك العنصر حتى ولو وجد على العنصر X قفل مشترك من قبل مناقلة أخرى.
- يمنح القفل الحصري على العنصر X إلى مناقلة ما، فقط إذا لم يكن على ذلك العنصر أي قفل من أي نمط.

- تطلق تسمية الاستبعاد المتبادل على السيناريو التالي:
إذا وجد قفل مشترك أو حصري على عنصر المعطيات X من قبل المناقلة T1، فإنه لا يمكن لمناقلة أخرى T2 أن تحصل على قفل حصري على X، إنما ينبغي على T2 أن تنتظر حتى تقوم T1 بتحرير العنصر X وتنفيذ التعليمة COMMIT.

وعلى الرغم من أن استخدام هذا النمط من الأقفال يحسن من أسلوب ولوج المعطيات ويضمن السلامة والتكامل، إلا أنه يزيد من العبء المفروض على مدير الأقفال وذلك بسبب ما يلي:

1. ينبغي أن تتم معرفة نوع القفل قبل أن يتم منحه للمناقلة
2. هناك ثلاثة أنواع مختلفة من العمليات المطبقة على الأقفال، وهي اختبار نوع القفل، ومنح القفل، وتحرير القفل
3. لقد تم تحسين النموذج ليشتمل على تطوير نوع القفل أي تحويله من مشترك إلى حصري، أو تخفيضه أي تحويله من حصري إلى مشترك.

المشاكل التي يمكن أن تنشأ نتيجة لاستخدام الأقفال المشتركة والحصرية:

- يمكن أن تتهدد خاصية تسلسلية المناقلات
- يمكن أن تتولد حالات إقفال متبادل، بحيث تقوم مناقلتين بانتظار بعضهما بعضاً لتحرير عنصر معطيات معين

أنظمة القفل على مرحلتين

- تستخدم أنظمة القفل على مرحلتين لمواجهة تهديد خاصية تسلسلية المناقلات الذي يظهر بسبب استخدام نظام الأقفال المشتركة والحصرية.
- يعرف نظام القفل على مرحلتين كيف يتم منح الأقفال للمناقلات وكيف يتم تحرير تلك الأقفال.
- يضمن هذا النظام تحقق التسلسلية بين المناقلات ولكنّه لا يمنع حدوث حالات الإقفال المتبادل.

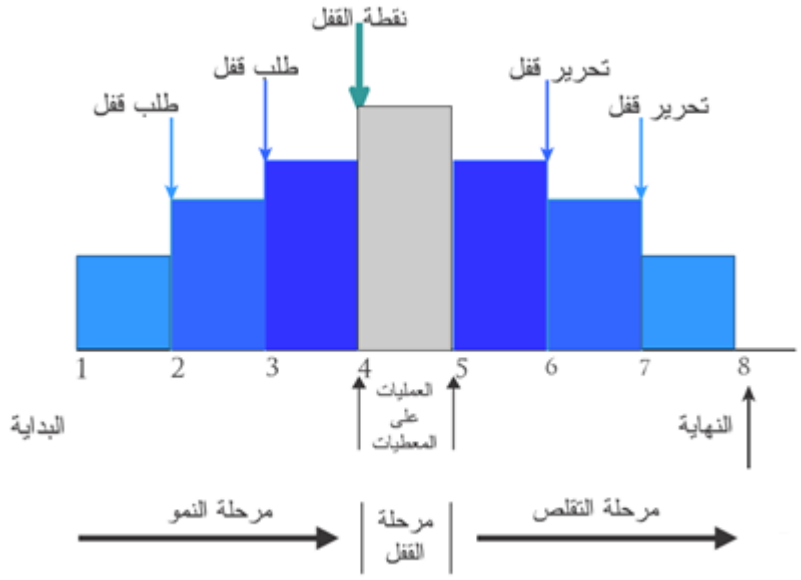
يتكون نظام القفل على مرحلتين من مرحلة صاعدة ومرحلة هابطة:

- المرحلة الصاعدة (مرحلة النمو): وفيها تقوم المناقلة بطلب كافة الأقفال على المعطيات التي تحتاجها وبدون إجراء أي عملية تحرير لأي عنصر معطيات، إلى أن تصل المناقلة إلى نقطة القفل، وهو مصطلح يدل على أن المناقلة قد انتهت من مرحلة طلب الأقفال
- المرحلة الهابطة (مرحلة التقلص): والتي تقوم فيها المناقلة بتحرير كافة الأقفال التي كانت قد حجزتها، بحيث يتم ذلك من دون أن تقوم تلك المناقلة بحجز أي قفل جديد.

يضمن بروتوكول الإقفال على مرحلتين تحقق ما يلي:

1. لا يمكن أن تتضارب أقفال مناقلتين
2. لا يمكن أن يتم إجراء أي عملية تحرير قفل قبل أي عملية حجز قفل في نفس المناقلة
3. لا تتأثر المعطيات حتى يتم حجز كافة أقفال المناقلة.

يوضح الشكل التالي مفهوم بروتوكول القفل على مرحلتين:



مشكلة الإقفال المتبادل

يحدث الإقفال المتبادل عندما تنتظر مناقلتان كل منهما الأخرى لكي تحرر قفلها عن عنصر معطيات معين.

مثال:

ليكن لدينا المناقلتان التاليتان:

T1 تقوم بالولوج إلى عنصر المعطيات X وعنصر المعطيات Y

T2 تقوم بالولوج إلى عنصر المعطيات Y وعنصر المعطيات X.

فإذا لم تحرر T1 قفلها على العنصر Y لا يمكن للمناقلة T2 أن تبدأ، وبالمثل، إذا لم تحرر T2 قفلها على العنصر X لا يمكن للمناقلة T1 أن تبدأ.

يطلق على الإقفال المتبادل أيضاً تسمية "العناق المميت".

هناك عدة تقنيات أساسية تستخدم للتحكم بالإقفال المتبادل:

- **منع حصول الإقفال المتبادل** يتم منع حصول إقفال متبادل من خلال رفض منح قفل لمناقلة ما إذا وجد احتمال حصول إقفال متبادل، كما يتم التراجع عن كافة التغيرات التي أجرتها تلك المناقلة، وبعد ذلك يتم إعادة جدولة تلك المناقلة للتنفيذ.
  يتم منع حصول الإقفال المتبادل إذاً لأننا نتجنب الحالات التي يمكن أن تؤدي إلى حدوثه.

- **اكتشاف الإقفال المتبادل** يقوم نظام إدارة قواعد المعطيات –وبشكل دوري– باختبار وجود حالات إقفال متبادل، فإذا تم رصد إحداها فإنه يقوم باختيار مناقلة ما لتكون "الضحية" ويتم التراجع عن التغيرات التي أجرتها في سبيل حل مشكلة الإقفال المتبادل المتولدة.

- **تجنب الإقفال المتبادل** يتم تجنب حصول حالات إقفال متبادل إذا ما قامت المناقلة باستحضار وتجهيز كافة الأقفال التي تحتاجها قبل البدء بالتنفيذ.

تعتمد استراتيجية اختيار طريقة التحكم بالإقفال المتبادل على بيئة نظام إدارة قواعد المعطيات، فإذا كان احتمال حدوث إقفال متبادل قليل نسبياً، بالتالي يعد استخدام طريقة اكتشاف الإقفال المتبادل هو الحل الأنسب، أما إذا كان الاحتمال كبير نسبياً فعندئذٍ ستكون طريقة منع حصول الإقفال المتبادل هي الأنسب، وفي حالة نظام بطيء الاستجابة فإن طريقة تجنب الإقفال المتبادل ستكون الأكثر تفضيلاً.

الأختام الزمنية

يعتمد مفهوم الختم بالزمن على ربط كل المناقلة بمعرف وحيد خاص بها، –أو بأسلوب آخر– بختمها زمنياً.
تتميز أختام الزمن بخاصتين أساسيتين هما : الوحدانية: أي عدم التكرار، والتزايدية: أي أن قيمة ختم الزمن تزداد باستمرار.
تملك العمليات التي تنتمي إلى نفس المناقلة نفس الختم الزمني، فإذا ما تضاربت مناقلتان مع بعضهما، يقوم نظام إدارة قواعد المعطيات باختيار الضحية ليتراجع عن كافة التغيرات التي قامت بها ثم يسند إليها ختماً زمنياً جديداً ويعدّها من جديد للتنفيذ.

تتجلى مساوئ طرائق الأختام الزمنية في التحكم بالتنفيذ المتزامن للمناقلات بأمرين أساسيين هما العبء المتولد عنها والذاكرة الإضافية التي تحتاجها، ذلك لأن كل حقل في قاعدة المعطيات سيتطلب حقلي أختام زمنية إضافيين، أحدهما لتخزين آخر زمن قراءة والآخر لتخزين آخر زمن تعديل.

مخططات انتظر / مت واجرح / انتظر:

وهي المخططات التي يتم من خلالها –أو بالاعتماد عليها– اختيار المناقلة الضحية التي يجب التراجع عنها وتحديد المناقلة التي ينبغي أن تتابع التنفيذ.

لنفترض وجود مناقلتين، لكل منهما ختم زمني وإحداهما أكبر من الأخرى... باستخدام مخطط انتظر/مت نعرف ما يلي:

- إذا كانت المناقلة التي تطلب القفل هي المناقلة الأكبر، فعليها أن تنتظر حتى تنتهي المناقلة الأصغر وتحرر عنصر المعطيات.
- إذا كانت المناقلة التي تطلب القفل هي المناقلة الأصغر، فعليها أن تموت (أي يتم التراجع عنها) وتعيد جدولة نفسها بنفس الختم الزمني الذي تمتلكه، أي أن الكبير ينتظر والصغير يعيد جدولة نفسه.

أما في حالة مخطط اجرح/انتظر، فلدينا الاعتبارات التالية:

- إذا كانت المناقلة التي تطلب القفل هي المناقلة الأكبر، فستقوم بجرح المناقلة الأصغر منها مجبرة إياها على التراجع عن التغيرات التي أجرتها وإعادة جدولة نفسها بنفس قيمة ختمها الزمني
- إذا كانت المناقلة التي تطلب القفل هي المناقلة الأصغر، فعليها أن تنتظر حتى تنتهي بقية المناقلات من عملها وتحرر القفل المفروض على عنصر المعطيات المطلوب، أي أن الكبير يجرح الصغير ويجبره على إعادة جدولة نفسه.

8. إدارة التعافي في قواعد المعطيات

إدارة التعافي عبارة عن مصطلح يدل على إعادة تخزين قاعدة المعطيات اعتباراً من حالة معينة غالباً ما تكون غير متّسقة إلى حالة متّسقة سابقة، وذلك من أجل الحصول على نسخ احتياطية يمكن استخدامها لاسترداد المعطيات في الحالات التي يمكن أن تحصل فيها مشاكل أو أخطاء.

تزود بعض أنظمة إدارة قواعد المعطيات بأدوات خاصة لإجراء نسخ احتياطي للمعطيات وبشكل آلي كل فترة زمنية محددة.

مستويات النسخ الاحتياطي:

يمكن أن يتم نسخ المعطيات على عدّة مستويات وهي:

1. نسخ كامل لقاعدة المعطيات ككل
2. نسخ التغيرات فقط، أي يتم تحديث آخر نسخة احتياطية بالمعطيات الجديدة المتغيرة
3. حفظ سجل المناقلات.

ينبغي الاحتفاظ بالنسخ الاحتياطية في مكان آمن، غالباً في مبنى آخر، كما ينبغي تأمينها من الأخطار المختلفة كالحرائق أو السرقات أو الفيضانات... الخ.

أنواع الأخطاء التي ينبغي تجنبها باستخدام النسخ الاحتياطي:

1. أخطاء برمجية: هي تلك الأخطاء المتولدة عن البرامج، كأنظمة التشغيل أو أنظمة إدارة قواعد المعطيات أو التطبيقات الأخرى أو الفيروسات
2. أخطاء عتادية: كالأخطاء المتولدة عن الذاكرة أو فشل القرص الصلب أو بعض قطاعاته ...الخ
3. الأخطاء والاستثناءات الناتجة عن التطبيقات: كالأخطاء الناتجة عن سوء الاستخدام من قبل المستخدمين، كمشكلة القسمة على صفر مثلاً
4. أخطاء ناجمة عن المناقلات: مثل حالات الإقفال المتبادل
5. أخطاء خارجية: كالفيضانات أو الحرائق أو الزلازل أو السرقات وغيرها.

9. المراجع:

http://www.studytonight.com/dbms/select-query

الفصل الثاني عشر:
إدارة المناقلات والتحكم المتزامن

العنوان	الصفحة
1. مقدمة	3
2. ميزات مخازن المعطيات	3
3. الفرق بين مخازن المعطيات وقواعد المعطيات العملياتية	4
4. بنية مخازن المعطيات Data warehouse architecture	4
1.4 جدول الحقائق (Fact table)	5
2.4 جدول الأبعاد (Dimension table)	5
3.4 البنية النجمية (Star schema)	5
4.4 البنية البلورية (Snowflake schema)	6
5. بناء مخازن المعطيات	8
1.5 استخلاص المعطيات Extraction	8
2.5 معالجة المعطيات Transformation	8
3.5 تحميل المعطيات Load	10
6. استخدامات مخازن البيانات	11
7. المراجع	12
8. تدريبات	13

الكلمات المفتاحية:

نظم دعم القرار، مستودع المعطيات، فلترة المعطيات، استخلاص المعطيات، الأبعاد، مستويات التجميع، الحفر للأسفل، التجميع للأعلى، مخزن المعطيات الصغير، تكامل المعطيات، إجرائية التحليل المباشر (OLAP)، المعطيات متعددة الأبعاد، مكعب المعطيات، شرائح المعطيات، إجرائية التحليل المباشر العلاقاتية (ROLAP)، إجرائية التحليل المباشر متعددة الأبعاد (MOLAP)، بنية نجمية، الحقائق، الواصفات، مكعب فائق، جدول الحقيقة، جدول البعد، هرمية الواصفات، التنقيب عن المعطيات.

ملخص:

يركز هذا الفصل على مستودعات المعطيات، ميزاتها الأساسية، اختلافها عن قواعد المعطيات العملياتية وطريقة بنائها والبنى الأساسية فيها (البنية النجمية)، بالإضافة إلى الفوائد منها.

الأهداف تعليمية:

يهدف هذا الفصل التعريف بالمفاهيم التالية:

- تعريف مخزن المعطيات
- ميزات مخازن المعطيات:
  1. موجهة حسب الغرض منها Subject oriented
  2. متكاملة Integrated
  3. متغيرة زمنياً Time-variant
  4. غير متحركة Non-volatile
- الفرق بين مخازن المعطيات وقواعد المعطيات العملياتية
- بنية مخازن المعطيات Data warehouse architecture:
  1. جدول الحقائق (Fact table):
  2. جدول الأبعاد (Dimension table):
  3. البنية النجمية Star schema
  4. البنية البلورية Snowflake schema
- بناء مخزن المعطيات:
  5. الاستخلاص Extract
  6. المعالجة Transform
  7. التحميل Load
- استخدامات مخازن البيانات

1. مقدمة

يعتبر مخزن المعطيات حسب تعريف Bill Inmon عام 1990، هو بيانات مجمعة من مصادر متنوعة. مجمّعة حسب الغاية منها، متكاملة فيما بينها، متغيرة زمنياً وغير متحركة (non-volatile)، بمعنى أنه لا تتم عليها عمليات إدخال وتعديل وحذف (transactions)، هذه البيانات مجمعة بغرض مساعدة محلل البيانات ومتخذ القرار في استخلاص ما يطلبه من معلومات بسرعة.

2. ميزات مخازن المعطيات

يتم تخزين البيانات في قواعد المعطيات العملياتية، ضمن جداول تحكم بنيتها قواعد التنظيم (فصل 6-7)، وقواعد تصميم بنى المعطيات العلائقية (فصل 2)، وغاية هذه القواعد الوصول إلى طريقة التخزين الأفضل للمناقلات (ادخال، تعديل وحذف).

أما مخازن المعطيات، فبنى المعطيات فيها لا تتبع قواعد التنظيم والقواعد العلائقية (فمثلاً يمكن أن تحوي جداول مخزن المعطيات حقول محسوبة وبيانات مكررة)، لأن الغاية الأساسية منها هي سرعة تنفيذ الاستعلامات، و تغير بياناتها يتم من خلال عمليات الاستخلاص – التحويل – التحميل (ETL)، ولا يتم عبر المناقلات التقليدية.

أهم ما يميز مخازن البيانات، هي أربع خصائص:

- **موجهة حسب الغرض منها Subject oriented**

تُنَظَم معطيات مستودع المعطيات وتُختَزَل في عدة مجالات، لإعطاء أجوبة عن الأسئلة المختلفة للشركة (حجم المبيعات حسب الفرع، حجم المبيعات موزعة على أشهر السنة، ...)، أي أن الأسئلة التي تتعلق بإجمالي المبيعات يمكن ايجاد أجوبتها في إحدى بنى مخزن المعطيات، ويمكن أن نجد بنى أخرى للاجابة على أسئلة أخرى (مثلاً إجمالي التكاليف موزعة حسب أقسام الشركة أو زمنياً). وسنشرح بنى التخزين في مخزن المعطيات في فقرة لاحقة.

- **متكاملة Integrated**

مصدر البيانات في مخزن المعطيات هو قواعد بيانات قد تكون متنافرة ومن أنواع مختلفة (MS access db, Oracle db, Excell sheets)، وقبل إدخالها إلى مخزن المعطيات تمر بمراحل من التنظيف والتصحيح والإكمال، وفي النهاية يجب مكاملتها قبل أن تصبح جزءاً من مخزن المعطيات.

- **متغيرة زمنياً Time-variant**

تتم تغذية مخزن المعطيات بالبيانات عادةً بشكل دوري، وغالباً ما تحمل بعداً زمنياً، للإجابة على الأسئلة التي تعكس التطور الزمني (تزايد المبيعات، تقليص التكاليف ..).

- **غير متحركة Non-volatile**

تدخل البيانات إلى مخزن المعطيات بتحميلها من المصدر (Bulk load)، ولا تدخل التسجيلات إفرادياً عبر المناقلات، وغالباً لا يتم حذف البيانات أو تعديلها بعد أن تصبح جزءاً من المخزن، والعمليات التي تتم عليها هي الاستعلام فقط.

3. الفرق بين مخازن المعطيات وقواعد المعطيات العملياتية

ثلاثة اختلافات أساسية (الفترة الزمنية، مستويات التجميع، الأبعاد):

- **الفترة الزمنية**

تغطي المعطيات العملياتية فترة صغيرة من الزمن، فالمناقلات تحتاج لسجلات المبيعات والفواتير اليومية والكميات المباعة والمخزنة، أما بالنسبة لنظم دعم القرار فلا تهتم بفاتورة معينة أو بمشتريات زبون معين، إنما بعمليات الشراء التي حدثت في شهر أو سنة، أو مثلاً بمشتريات نمط معين من الأشخاص.

- **مستويات التجميع**

يتم تجميع المعطيات في نظم دعم القرار في مستويات متعددة من معطيات فردية تقريباً إلى معطيات كلية شاملة. حيث يمكن مثلاً للمدير أن يرى المبيعات بحسب القطاع أو بحسب المدينة داخل القطاع أو بحسب المتجر داخل المدينة، وتدعى عمليات طلب معلومات تفصيلية أكثر بالحفر للأسفل (drill down)، أم طلب معلومات مُجمَّعة فتدعى التجميع للأعلى (roll up).

- **الأبعاد**

يتعامل نظام دعم القرار مع أبعاد متعددة للمعطيات، مثلاً إذا أردنا أن نعرف المبيعات التي حدثت في قطاع معين وخلال شهر معين فلدينا بعدين للمعطيات (بعد للمنطقة) و(بعد للزمن).

4. بنية مخازن المعطيات Data warehouse architecture

تعتبر بنى التخزين في قواعد المعطيات جداول تخضع لقواعد التنظيم، أما في مخازن المعطيات فتختلف بنى التخزين قليلاً، فهي جداول إلا أنها ليست بالضرورة من الشكل النظامي الثالث، ويتم تصميمها وفق إحدى بنيتين: البنية النجمية (Star schema) والبنية البلورية (Snowflake schema)، وفيما يلي شرح لكلا البنيتين وتطبيق على مثال يعكس تحليل مبيعات شركة.

قبل شرح البنية النجمية والبنية البلورية ، يجب أن نعرف أن كلا البنيتين يعتمد على نوعين من الجداول، هما جدول الحقائق، وجدول الأبعاد.

جدول الحقائق (Fact table)

يتألف جدول الحقائق من نوعين من الحقول: مفاتيح مستوردة من جداول الأبعاد، وقياسات تتضمن قيم (غالباً ما تكون رقمية)، وهذه القياسات قد تكون مفصّلة أو مجمعة.

مثال عن المفاتيح المستوردة: رقم الفرع، الشهر، العام، صنف المنتج، وغيرها.

مثال عن القياسات: القيمة الإجمالية للمبيعات، كمية المبيعات، وغيرها.

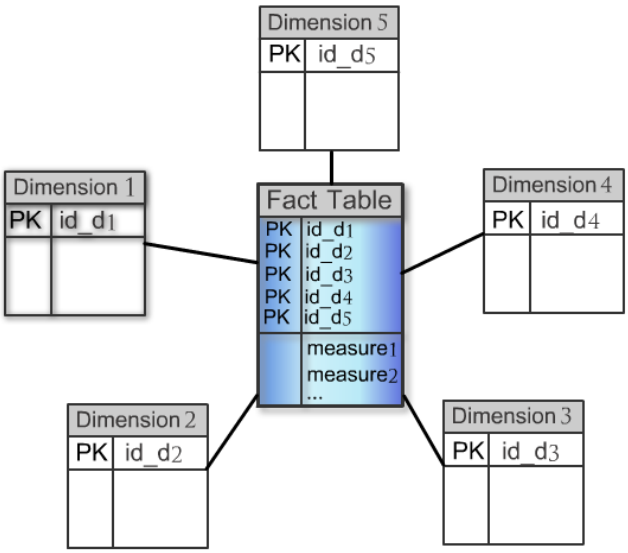
جدول الأبعاد (Dimension table)

قد يكون جدول الأبعاد بسيطاً (مفتاح وقيمة واحدة نصية أو تاريخ)، وقد يمثل بنية هرمية (مفتاح، سنة، شهر، يوم مثلاً)، وفي الحالتين يصدّر جدول الأبعاد مفتاحه لجدول الحقائق ويكون جزءاً من مفتاحه الأساسي المركب.

وتمثل جداول الأبعاد عادةً توزع جغرافي أو تصنيف لمنتج، وهي غالباً الأبعاد التي تطلب الاحصاءات مجمّعة حسبها.

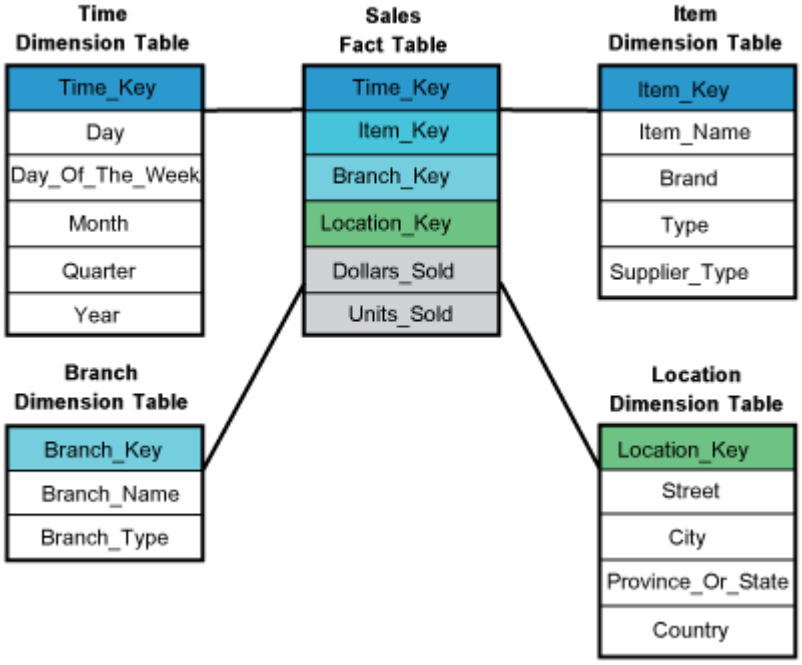
البنية النجمية (Star schema)

البنية النجمية هي بنية بسيطة، تشبه نجمة مركزها هو جدول الحقائق، ورؤوسها هي جداول الأبعاد:



أهم ميزات البنية النجمية:

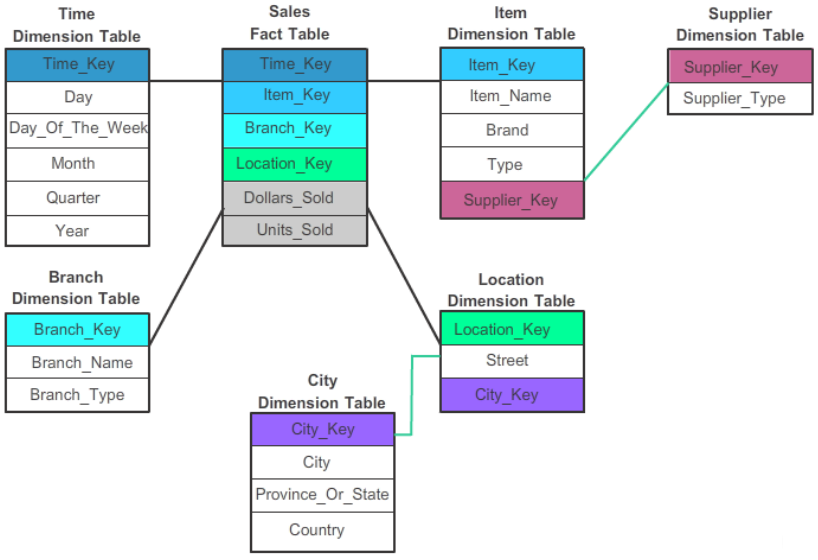
- سهلة الفهم وبسيطة نسبياً.
- الاستعلامات المبنية على جدول الحقائق بسيطة التركيب (عدد قليل من الجداول للربط) وسريعة التنفيذ.
- قد يزداد حجم جداول الأبعاد كثيراً (جداول غير منظمة)، وقد يستهلك تحميلها بالمعطيات وقتاً كبيراً.
- تعتبر البنية النجمية الأوسع انتشاراً والأكثر استخداماً في مخازن المعطيات.



البنية البلورية (Snowflake schema)

تكون جداول الأبعاد في البنية النجمية غير منظمة واحتمال التكرار فيها وارد، فمثلاً في الشكل السابق يمكن أن نجد في البعد المكاني (Location)، تكرار للقيمة دمشق بعدد الشوارع المدخلة أسماؤها.

بينما تكون جداول الأبعاد في البنية البلورية منظمة، ومثل الجدول Location تتم تجزءته إلى أكثر من جدول كما في الشكل التالي:



ويتم بناء الجداول بنوعيها (الحقائق والأبعاد) في مخزن المعطيات والتخاطب معها من خلال توسعة للغة SQL تدعى DMQL، وتدعمها معظم نظم ادارة قواعد المعطيات العلائقية (Oracle, SQL Server).

سنعرض هنا بعض الأمثلة على لغة DMQL:

البنية النجمية

Sales fact table

```
define cube sales star [time, item, branch, location]:

dollars sold = sum(sales in dollars), units sold = count(*)
```

Dimension tables

```
define dimension time as (time key, day, day of week, month, quarter, year)

define dimension item as (item key, item name, brand, type, supplier type)

define dimension branch as (branch key, branch name, branch type)

define dimension location as (location key, street, city, province or state, country)
```

==========

البنية البللورية

Sales fact table

```
define cube sales snowflake [time, item, branch, location]:

dollars sold = sum(sales in dollars), units sold = count(*)
```

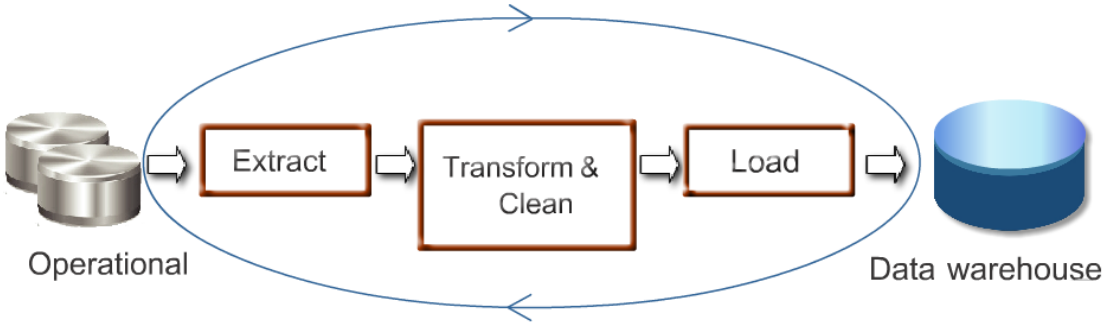
Dimension tables

```
define dimension time as (time key, day, day of week, month, quarter, year)

define dimension item as (item key, item name, brand, type, supplier (supplier key, supplier type))
define dimension branch as (branch key, branch name, branch type)

define dimension location as (location key, street, city (city key, city, province or state, country))
```

5. بناء مخازن المعطيات

تبدأ عملية بناء مخزن المعطيات باستخلاص البيانات من مصادرها، تعديلها وتصحيحها ومكاملتها، وأخيراً إدخالها إلى مخزن المعطيات (ETL)، ويمكن تكرار العملية من بدايتها بشكل دوري (شهرياً مثلاً)، ويستخدم مخزن البيانات بعد بنائه وتغذيته بالمعطيات، كقاعدة معطيات تبنى عليه أدوات الاسترجاع والعرض وتحليل البيانات.

فيما يلي سنفصّل مراحل بناء مخزن المعطيات، مع الأمثلة:



استخلاص المعطيات Extraction

تكون البيانات في مخزن المعطيات مجمعة لتخدم غرضاً معيناً، كتحليل مبيعات شركة مثلاً، وهنا نجد أن البيانات المطلوبة قد تكون موزعة في أكثر من مصدر (قواعد بيانات عملياتية، جداول الكترونية Excel sheets وغيرها)، وقد تكون قواعد البيانات العملياتية مختلفة من حيث النوع والبنى داخلها.

وهنا يتم عادةً توحيد صيغة البيانات المستخلصة لنقلها إلى المخزن، إما عن طريق مراحل وسيطة (ملفات نصية أو قواعد معطيات)، وإما عن طريق التحكم بتعليمات Select التي يتم من خلالها استخلاص البيانات من المصدر، ويمكن دائماً استخدام أدوات ETL التي توفرها نظم إدارة قواعد المعطيات العلائقية.

معالجة المعطيات Transformation

غالباً ما تكون البيانات المستخلصة من المصادر بحاجة إلى معالجة: **تصحيح** و**استكمال** و**تنظيف**، لتتلاءم وتكون قابلة للنقل إلى الحقول المقابلة في مخزن المعطيات.

التصحيح: قد تكون البيانات مخزنة بأنماط لا تتوافق مع نمط الجهة التي ستصب فيها، يمكن مثلاً أن نجد التاريخ في أحد مصادر فواتير الزبائن مخزن على شكل نص، أو قد نجد فروع الشركة في أحد المصادر مرمزة ومدخلة في فهارس بينما نجدها في مصادر أخرى مدخلة بشكل نصي، وهنا لا بد من تصحيح بيانات المصادر المختلفة وتوحيد صيغتها لتسهيل نقلها إلى مخزن المعطيات.

الاستكمال: في كثير من الأحيان يمكن أن تكون بيانات المصدر ناقصة، فمثلاً إذا كان أحد المصادر هو قاعدة بيانات لمبيعات أحد الفروع في الشركة، فمن الممكن أن لا نجد اشارة للفرع في الفواتير كونها تخص فرعاً واحداً، وهنا لا بد من إضافة رمز أو الرقم المعرف للفرع لكل فاتورة قبل نقل بياناتها لمخزن المعطيات.

التنظيف: يمكن أن يتضمن المصدر بيانات خاطئة (فاتورة غير مستكملة – اسم الزبون غير مدخل مثلاً)، والتنظيف هنا يعني حذف هذه الادخالات أو اسناد قيم افتراضية للمدخلات الناقصة أو المدخلة خطأً.

وتعتبر المعالجة الأهم للبيانات قبل نقلها إلى مخزن المعطيات، هي توحيد صيغتها.

وفيما يلي مثال عن مصدرين لمبيعات فرعين لشركة، ونموذج عن عمليات التصحيح اللازم تطبيقها عليها:

مبيعات المنطقة الجنوبية (المصدر الأول)

المبيعات مدخلة بالليرة السورية والاجمالي متضمن في الفاتورة

Invoice

Id	Date	City	Total
1	1/1/2015	Damascus	50000
2	15/1/2015	Sweida	40000
3		Damascus	15000
4	4/2/2015	Daraa	2500
5	1/3/2015	Damascus	10000

==========

مبيعات حمص (المصدر الثاني)

المبيعات مدخلة بالدولار

Invoice

Id	Date	Client
1	1/1/2015	X client
2	15/2/2015	Y client

Item

Id	Invoice	Mat	Qty	Price
1	1	M1	2	15
2	1	M2	5	22
3	2	M2	10	22
4	1	M3	6	30

==========

مخزن البيانات

Sales

Branch	Month	Total
Damascus	Jan	50000
Damascus	Mar	10000
Sweida	Jan	40000
Daraa	Feb	2500
Homs	Jan	99000
Homs	Feb	66000

==========

ملاحظات:

- الفاتورة رقم 3 (المنطقة الجنوبية) لم يتم ادخال التاريخ لها، وهي حالة يتم تنظيفها باستثناء الادخالات التي لا تتضمن تاريخ.
- الأسعار مدخلة في المصدر الأول بالليرة السورية وفي المصدر الثاني بالدولار، وهي حالة بحاجة معالجة (يمكن تحويل الأسعار في المصدر الثاني إلى الليرة السورية بضربها بوسطي سعر 300)
- المدينة في المصدر الثاني غير مدخلة (استكمال)، يمكن تصحيحها بإضافة "حمص" في التسجيلات التي يتم استخلاصها من المصدر الثاني.
- يمكن نقل بيانات كل مصدر إلى محطة وسيطة (بعد معالجتها) قبل نقلها إلى مخزن البيانات.
- البيانات في مخزن المعطيات مجمعة حسب الشهر والمدينة.

==========

```
Select date, city, sum(total) from Invoice where not (isnull(date))
and not (isnull(city))
Group by city,date;

Select date, 'Homs', sum(y) from
(
Select invoice.id as x, date, 'Homs', sum(qty*price*300) as y
From invoice, item
Where invoice.id=item.invoice
Group by date
)
Group by date;
```

- البيانات الناتجة عن تعليمتي SQL السابقتين هي من الشكل:

date	city	Total

وقبل نقلها لمخزن المعطيات يجب تحويل التاريخ إلى شهر، والتجميع حسب الأشهر.

تحميل المعطيات Load

يعني تحميل البيانات نقلها بعد استخلاصها ومعالجتها إلى مخزن المعطيات، ويتم تحميل البيانات بشكل دوري (كل شهر مثلاً)، ويمكن أن يتم التحميل إما من المصدر مباشرة أو من المحطات الوسيطة.

تتوفر في معظم نظم إدارة قواعد المعطيات العلائقية (Oracle, MS Sql server, ...)، أدوات الاستخلاص والمعالجة وتحميل البيانات، وغالباً ما تكون هذه الأدوات بيانية وسهلة الاستخدام.

6. استخدامات مخازن البيانات

تعتبر أهم التطبيقات التي تبنى على مخازن المعطيات هي نظم دعم القرار، وهي تعتمد على استخلاص المعلومات (حجوم كبيرة من المعطيات)، وعرضها بشكل مقروء (نصوص، جداول، مخططات بيانية)، بحيث تسهل عمل متخذ القرار وتعطيه المعلومات اللازمة وبشكل سريع ليبني قراره على معلومات دقيقة.

تحتاج نظم دعم القرار إلى حجوم كبيرة من المعطيات (عينة احصائية كبيرة)، لتكون المؤشرات والحقائق المبنية عليها، أقرب ما يكون إلى الدقة.

من أهم المجالات التي تخدم فيها مخازن المعطيات في توجيه القرار والمساعدة على اتخاذه:

- تخطيط الانتاج، بالاعتماد على البيانات والاحصاءات التي تتعلق بحجم المبيعات وتوزعها الزمني على مدار العام.
- تحليل الزبائن، من حيث أفضليات الشراء ومواعيدها.
- التحليل العملياتي كإدارة العلاقة مع الموردين (Supply chain management)، ومع الزبائن والموزعين.

7. المراجع

- http://www.tutorialspoint.com/dwh/dwh_data_warehousing.htm

8. تدريبات

- مخازن المعطيات هي قواعد معطيات علائقية تخضع لقواعد التنظيم ؟
  1. صح.
  2. خطأ.

الإجابة: (2) (ليس بالضرورة أن تكون الجداول في مخزن المعطيات من أي شكل نظامي)

- تصميم مخزن المعطيات مقاد بالمعطيات ؟
  1. صح.
  2. خطأ.

الإجابة: (2) (تصميم مخزن المعطيات مقاد بالعمليات)

- لغة التخاطب مع محتوى مخازن المعطيات هي DMQL؟
  1. صح.
  2. خطأ.

الإجابة: (1)

- تكرار المعطيات غير وارد في مخزن المعطيات ؟
  1. صح.
  2. خطأ.

الإجابة: (2) (تكرار المعطيات وارد في مخازن المعطيات، وخاصة في البنية النجمية)