الفصل الرابع
تنظيم قاعدة البيانات

1. مفهوم التنظيم (Normalization)

يُقصد بالتنظيم (أوالتسوية Normalization) تصميم جداول قاعدة البيانات بحيث نتحكم بتكرار البيانات ونتجنب حالات الشذوذ التي يمكن أن تنتج عن عمليات الإضافة والحذف والتعديل على تلك البيانات.

سنقوم بدايةً بدراسة بعض الإشكاليات التي يُعاني منها التصميم السيئ لقاعدة البيانات، ومن ثم سندرس مستويات التنظيم وخصائص كل منها.

1.1. إشكاليات التصميم السيئ لقاعدة البيانات

يوجد هناك مشاكل أساسية ينبغي علينا تجنبها خلال عملية تصميم قاعدة البيانات.

من هذه المشاكل:

1. التكرار غير المفيد في تخزين البيانات (Redundancy Anomaly)
2. االإضافة الخاطئة(Insertion Anomaly)
3. الحذف الخاطئ (Deletion Anomaly)
4. التعديل الخاطئ (Update Anomaly)

في الواقع إن استخدام طريقة منظمة لتصميم جداول قاعدة البيانات يقلل من هذه المشاكل إلى حد كبير. سنتناول بعض الأمثلة التي توضح الشذوذ التي يمكن أن تؤثر على تصميم قاعدة البيانات وبعض المشاكل المحتمل حدوثها.

مثال1: لنفرض لدينا الجدول الآتي:

Name	Project	Profession	Salary	Budget
Mhd	DB	Assistant	7000	10000
Samer	UML	Designer	1000	20000

Samer	XML	Designer	1000	60000
Ali	XML	Manager	3000	60000
All	UML	Consulter	3000	20000
All	DB	Consulter	3000	10000
Saml	WEB	Employee	4000	3300
Majd	DB	Manager	500	10000
Majd	XML	Designer	500	60000
Yasser	XML	Designer	15000	60000
Yasser	DB	Designer	15000	10000

المشاكل التي يمكن أن نلاحظها في الجدول يمكن أن نُبيّنها فيما يلي.

1. التكرار غير المفيد:

- إن المعلومات المتعلقة بالراتب لكل موظف مكررة عدة مرات بلا فائدة.
- إن المعلومات المتعلقة بالميزانية لكل مشروع مكررة عدة مرات بلا فائدة.
- قيم الحقل الاسم (Name) مكررة وبالتالي لا يمكن اعتباره مفتاح للعلاقة.
- أيضاً قيم الحقل المشروع (Project) مكرر وبالتالي لا يمكن اعتباره مفتاح.
- إن المفتاح لهذا الجدول هو التركيب (Name, Project).

2. الإضافة الخاطئة:

ولما كان المفتاح هو (Name, Project) فإنه ليس بالإمكان إدراج موظف بدون تحديد المشروع الذي يعمل فيه لأنه لا يجوز أن يكون المفتاح عديم القيمة (Null). كذلك لا يمكن إدراج مشروع بدون تحديد على الأقل اسم موظف يعمل في هذا المشروع.

3. الحذف الخاطئ:

بفرض أن موظفاً ما قد أنهى عمله في الشركة أو أن موظفاً ما قد تمّ إعفاؤه من الخدمة في مشروع. بفرض أن هذا الموظف هو الموظف الوحيد في المشروع (Sami). إن عملية حذف سجل ذلك الموظف ستؤدي إلى فقدان معلومات عن المشروع الذي يعمل فيه.

4. التحديث الخاطئ:

إن التكرار في تخزين البيانات سيؤدي إلى تكرار في تحديث تلك البيانات. فمثلاً في الجدول السابق هناك تكرار في تخزين الراتب لكل موظف. إذا ما أردنا تحديث راتب موظف ما فإن ذلك يتطلب تحديثه في كل سجلات الجدول. إن هذا يقلل من فعّالية قاعدة البيانات. بالطريقة نفسها فإن تحديث ميزانية مشروع ما سيؤدي إلى المشكلة نفسها.

مثال2: بفرض أن شركة ما تضم مجموعة من مسؤولي المبيعات الذين يعملون في عدة مواقع. ترغب الإدارة بتخزين البيانات المتعلقة بموظفيها وبيانات مستودعاتها في قاعدة بيانات. المحاولة الأولى لبناء هذه القاعدة يمكن أن تكون وفق الجدول الآتي:

ID	Name	Address	Role	Store N°	Store Address	Phon Store
ID 1	Jane	Add1	Sales	S1	Sa1	011-1234567
ID 2	Fred	Add 2	Sales	S1	Sa1	011-1234567
ID 3	Edwar	Add 3	Manager	S1	Sa1	011-1234567
ID 4	Ann	Add 4	Sales	S2	Sa2	021-7891234
ID 5	Jhone	Add 5	Sales	S2	Sa2	021-7891234
ID 6	Smith	Add 6	Manager	S2	Sa2	021-7891234

يتضمن الجدول السابق بيانات الموظفين والمستودعات، ومن الواضح وجود تكرار غير مفيد في تلك البيانات نبينها فيما يأتي:

- إذا أردنا إدخال بيانات موظف جديد فيجب أيضاً إدخال بيانات المستودع الذي يعمل فيه، وللحفاظ على عدم تناقض البيانات في القاعدة يجب إدخال بيانات المستودع بدقة وبشكل مطابق للقيم المدخلة سابقاً (لأحد الموظفين السابقين في المستودع نفسه)، هنا تظهر مشكلة تكرار البيانات بالإضافة طبعاً إلى حجم التخزين المهدور.

- بفرض أننا نريد إدخال بيانات مستودع جديد قبل توظيف أحد فيه، هذا يتطلب إدخال قيم Null في بيانات الموظف ومن ضمنها الحقل المفتاح (ID) وهذا يخرق شرط تكامل البيانات (Integrity). هذا ما يدعى بإشكالية الإدخال (Anomaly Insertion).
- بفرض أننا حذفنا سجل الموظف الأخير في أحد المستودعات، سيؤدي ذلك إلى فقدان بيانات مستودع موجود في الشركة. هذا يرتبط بإشكالية الحذف (Deletion Anomaly).
- إذا أردنا تغيير رقم هاتف أحد المستودعات، عندها يجب تغيير رقم الهاتف في كل سجلات موظفي ذلك المستودع، فإذا تم تغيير جزء من هذه السجلات دون جزء آخر سيؤدي ذلك إلى بيانات متناقضة في قاعدة البيانات، وهذا ما يرتبط بإشكالية التعديل (Update Anomaly).

هذه الأخطاء والمشاكل عادةً لا تظهر إذا قام مصمم قاعدة البيانات باتباع الطرق الصحيحة في عملية تصميم جداول تلك القاعدة مع العلم أنه يمكن أن تظهر هذه المشاكل عند عدم تحديد المتطلبات المختلفة للنظام بشكل صحيح وكامل. يمكن مبدئياً حل المشاكل السابقة من خلال تقسيم الجدول السابق إلى جدولين، أحدهما يحمل بيانات الموظفين ويستورد مفتاح الجدول الثاني الذي يحمل بيانات المستودعات كما يأتي:

ID	Name	Address	Role	Store N°
ID1	Jane	Add1	Sales	S1
ID 2	Fred	Add 2	Sales	S1
ID 3	Ed	Add 3	Manager	S1
ID 4	Ann	Add 4	Sales	S2
ID 5	Jhone	Add 5	Sales	S2
ID 6	Smith	Add 6	Manager	S2

Store N°	Store Address	Store Phone
S1	Sa1	011-1234567
S2	Sa2	021-7891234

الأن عند إدخال بيانات موظف جديد، يتم تسجيل رقم المستودع الذي يعمل فيه، أما بقية بيانات المستودع فهي مخزنة في جدول مستقل وبالتالي لن تظهر بيانات متناقضة في قاعدة البيانات، وإذا أردنا إدخال بيانات مستودع جديد فيمكن إدخالها بغض النظر عمن يعمل في المستودع، وهذا يُجنبنا إشكالية الإدخال. وعند حذف الموظف الأخير في مستودع، تبقى بيانات تلك المستودع في جدول مستقل. وعند تعديل رقم هاتف مستودع ما، يتم التعديل على سجل واحد في جدول المستودعات.

تُعد عملية تقسيم الجدول السابقة من عمليات تنظيم قاعدة البيانات، التي سنتعرف عليها بتفصيل أكثر، ولكن بعد عرض مجموعة من المفاهيم الأساسية في هذا الخصوص من خلال الفقرات اللاحقة.

1.2. التبعية الوظيفية

(Functional Dependency, FD)

يُقال عن الواصفة X إنها تُحدد (Determine) الواصفة Y أو إن Y تعتمد اعتماداً وظيفياً (أو نقول إنه يوجد ارتباطاً تابعياً) على X إذا كانت كل قيمة من X تُحدد قيمة واحدة فقط من Y، ويُرمز لذلك بالتبعية الوظيفية الآتية $X \rightarrow Y$. يمكن التعبير عن ذلك رياضياً كما يأتي:

إذا كان هناك سطران T1 وT2 في مجموعة الأسطر R وكان T1.X=T2.X وأن $T1,T2 \in R$ فإن T1.Y=T2.Y.

لنتأمل العلاقة الموضحة في الجدول الآتي:

EmpN°	EmpName	EmpAdress
112	Samer	EA1
652	Salem	EA2
214	Hadi	EA3
432	Salem	EA4

نلا حظ من الجدول أن رقم الموظف (EmpN°) يُحدد اسمه (EmpName)، وكل قيمة لـ EmpN° تقابلها قيمة وحيدة لـ EmpName. لذلك يُقال إن EmpName تتبع وظيفياً لـ EmpN°، أو يُقال إن EmpN° تُحدد EmpName. نقول إن هناك ارتباطاً وظيفياً بين EmpN° و EmpName لأنه لا يمكننا أبداً إدراج الزوج <214,Sami> بدون حذف الزوج <214,Hadi>. بالمقابل لا يوجد ارتباط وظيفي بين اسم الموظف (EmpName) وعنوان الموظف (EmpAdress). لذلك نكتب

EmpName−/→ EmpAdress.

مثال آخر: إذا كان لدينا جدول يمثل طلاب الجامعة ويحتوي على خاصية رقم الطالب (Number) وكذلك اسمه (Name) ومعدله (Average) كما في الجدول الآتي:

Number	Name	Average
10611205	Samer	67
10711311	Majd	75
10854325	Sami	67
10612312	Hamd	83
10723141	Salem	75

نلاحظ من الجدول السابق أن الواصفة رقم الطالب تحدد الواصفة اسم الطالب لأن كل قيمة من الواصفة رقم الطالب تحدد قيمة واحدة من الواصفة اسم الطالب ونكتب Number → Name. بينما لو نظرنا إلى المعدل فإنه لا يحدد رقم الطالب كما أنه لا

request (clientN°, proudctN°, clientName, productName)

clientN, proudectN -> clientName

clientN, prodectN → productName

يحدد اسم الطالب وذلك لأنه من الممكن أن نجد قيمة للمعدل مرتبطة بأكثر من قيمة من رقم الطالب وكذلك من اسم الطالب. أي إنه من الممكن أن يكون نفس المعدل لأكثر من طالب.

clientN → clientName

1.2.1. التبعية الوظيفية والمفاتيح

انطلاقاً من مفهوم التبعية الوظيفية ومن تعريف مفتاح العلاقة الذي هو واصفة أو أكثر تحدد صف واحد داخل الجدول (بدون تكرار)، يمكن أن نستنتج أن المفتاح هو الواصفة (أو مجموعة الواصفات) التي نستطيع من خلالها تحديد وظيفياً الواصفات الأخرى الموجودة كافة في الجدول. أي إن المفتاح هو واصفة (أو أكثر) من واصفات العلاقة بحيث تكون مصدراً لجميع التبعيات الوظيفية اللواتي هدفها باقي واصفات تلك العلاقة.

A	B	C	D

1.2.2. الإعتمادات الوظيفية التامة والاعتمادات الوظيفية المباشرة

نقول إن الواصفة Y تعتمد وظيفياً بشكل تام (أو كلي) على واصفة مركبة (Fully Functional Dependency)، إذا كانت تلك الواصفة المركبة تُحدد Y، و Y لا تعتمد وظيفياً على جزء منها. أي إن الاعتماد الوظيفي $A,B,C \to D$ هو اعتماد وظيفي تام إذا كان $A,B -/\to D$ و $A,C -/\to D$ و $B,C -/\to D$ و $A -/\to D$ و $B -/\to D$ و $C -/\to D$.

نقول عن الاعتماد الوظيفي $A \to B$ إنه اعتماد وظيفي مباشر إذا تعذر وجود واصفة أخرى C تحقق $A \to C$ و $C \to B$.

1.2.3. بديهيات الارتباطات الوظيفية

لتكن لدينا الواصفات X, Y, Z, U, V في الجدول العلائقي R. يمكن بسهولة برهان صفات الارتباطات الوظيفية الآتية:

1. الإنعكاسية (Reflexivity): إذا كانت الواصفة X مجموعة جزئية من الواصفة Y في جدول معين ($X \subseteq Y$) فإن X تعتمد اعتماداً وظيفياً على Y ويمكن كتابة ذلك بالشكل $Y \to X$.

2. الازدياد (Augmentation): إذا كانت $X \to Y$ وكان U, V واصفتين ما في R بحيث $U \subseteq V$ فإن $X, V \to Y, U$.

3. المتعدية (Transitivity): إذا كان الارتباط الوظيفي $X \to Y$ وكذلك الارتباط الوظيفي $Y \to Z$ فإن $X \to Z$.

4. قاعدة شبه التعدي (Pseudo-Transitivity): إذا كانت الاعتماد الوظيفي $X \to Y$ وكذلك الاعتماد الوظيفي $Y, U \to Z$ فإن $X, U \to Z$.

5. قاعدة الاجتماع (Union): إذا كانت الاعتمادات الوظيفية $X \to Y$ و $X \to Z$ فإن $X \to Y, Z$.

6. قاعدة التحليل (Decomposition): إذا كان الارتباط الوظيفي $X \to Y, Z$ فإن $X \to Y$ و $X \to Z$.

1.3. مستويات التنظيم

كما سبق وعرفنا، التنظيم بأنه عملية تسوية وتنظيم لجداول قاعدة البيانات العلائقية بهدف التقليل من تكرار البيانات وتقليص حجم التخزين المطلوب والحد من إشكاليات الإدخال والحذف والتعديل على القاعدة.

بشكل عام يتم تنظيم قاعدة البيانات من خلال إخضاعها لمجموعة اختبارات وتعديل هذه القاعدة لتحقق مجموعة معايير. من أجل ذلك نعرف مجموعة من مستويات التنظيم.

مستويات التنظيم الأساسية هي ثلاث:

- الشكل النظامي الأول

(First Normal Form, 1NF)

- الشكل النظامي الثاني

(Second Normal Form, 2NF)

- الشكل النظامي الثالث

(Third Normal Form, 3NF)

يضاف لهذه المستويات الثلاثة الشكل المقترح من قبل R.Boyce والذي يُطلق عليه BCNF (Boys Cood Normal Form)، والشكلين النظامي الرابع (Forth Normal Form, 4NF) والشكل النظامي الخامس (Fifth Normal Form, 5NF). فيما يأتي سنقوم بدراسة هذه الأشكال.

1.3.1. الشكل النظامي الأول

(First Normal Form, 1NF)

يُقال عن جدول في قاعدة البيانات إنه من الشكل النظامي الأول إذا كانت كل واصفة لا يمكن أن تظهر أكثر من مرة في العلاقة، وكذلك لا يمكن تقسيم أي واصفة إلى مجموعة من الواصفات الأخرى (بمعنى آخر لا تشكل بحد ذاتها علاقة). أي إذا كان تقاطع كل سطر وعمود في الجدول يتضمن قيمة وحيدة غير قابلة للتجزئة.

لنتأمل الجدول الآتي:

Employee

Employee Name	Children		Obtained Diplomas		
	First Name	Birth Day	Type	Exam	
				Place	Date
Sami	Jim	1975	Bac	Damascus	1999
	Loulou	1977	License	Aleppo	1994
	Sophi	1972	Engineer	Damascus	1993
Paul	Jhon	1973	License	Damascus	1999
	Rimi	1977	Doctorate	Homs	1991

نلاحظ من الجدول السابق أن الواصفة Children مقسمة إلى واصفتين First Name و Birth Day في نفس العلاقة. كذلك الواصفة Type من أجل Sami تظهر 3 مرات: Bac, License, Engineer فهذه العلاقة ليست في الشكل النظامي الأول (1NF).

كمثال على علاقة في الشكل النظامي الأول يمكن أن نورد العلاقة Request الآتية:

Request(Client N°,Product N°, Client Name, Product Name)

وضع العلاقة (أو الجدول) في الشكل النظامي الأول لا يحل مشاكل التصميم السيئ لقاعدة البيانات إنما هو خطوة في طريق حلها، فعلى الرغم من أن العلاقة السابقة في الشكل النظامي إلا أنها لا تزال تعاني من مشكلة تكرار البيانات، وإشكاليات الإضافة والتعديل والحذف. السبب في ذلك هو فقط الارتباط الوظيفي client N°,product N° → client Name وكذلك الارتباط الوظيفي client N°,product N° → product Name يمكن أخذهما بعين الاعتبار، وبالتالي يمكننا إدراج الأسطر الخاطئة (زبونين يحملان الرقم نفسه أو مادتين لهما الرقم نفسه) الآتية بدون أي مشكلة:

< 52, p3, Alice , chair >
< 52, p5, Sami, Table >
< 27, p3, Sophi, Table >

بهدف أخذ جميع الارتباطات الوظيفية بعين الاعتبار من المفيد إذاً تقسيم العلاقة السابقة إلى 3 علاقات وهي:

Client(client N°, client Name)
Product(product N°, product Name)
Request(client N°, product N°)

1.3.2. الشكل النظامي الثاني

(Second Normal Form, 2NF)

يُقال عن علاقة إنها من الشكل النظامي الثاني إذا حققت ما يأتي:

1. هي من الشكل النظامي الأول.

2. كل الارتباطات الوظيفية التي مصدرها مفتاح العلاقة وهدفها باقي واصفات العلاقة هي ارتباطات تامة. بمعنى آخر كل الواصفات التي لا تشكل جزءاً من المفتاح، تعتمد وظيفياً بشكل تام على تلك المفتاح.

يمكن أن نستنتج بسهولة أن كل علاقة من الشكل النظامي الأول ومفتاحها يقتصر على واصفة واحدة فقط هي من الشكل النظامي الثاني، لكن العكس غير صحيح بالضرورة.

لناخذ مثلاً العلاقة Client الآتية:

Client(client N°, client Name, merchantman N°, merchantman Name)

هذه العلاقة أيضاً تُبرز مشكلة لأن الارتباط الوظيفي

merchantman N° → merchantman Name

لم يتم أخذه بعين الاعتبار وبالتالي يمكننا إدراج الأسطر الخاطئة الآتية (الرقم نفسه لأكثر من تاجر):

< 73, Jhon, m17, Loulou >

< 125, Sami, m17, Alice >

إذاً من المناسب تقسيم العلاقة السابقة إلى علاقتين على الشكل الآتي:

Client(client N°, clientName, merchantman N°)

Merchantman(merchantman N°, merchantman Name)

وضع العلاقة في الشكل النظامي الثاني يبدأ بتحديد جميع الارتباطات الوظيفية بين الواصفات، ومن ثم تجزئة العلاقة إلى مجموعة علاقات (Decomposition) تضمن اعتماد جميع الواصفات التي لا تشكل جزءاً من المفتاح الأساسي بشكل تام على تلك المفتاح.