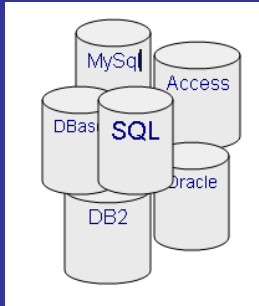




# ניהול נתונים – קבצים מול מסד נתונים



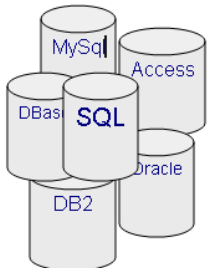
מהו מסד נתונים?

**מסד נתונים (או בסיס נתונים Database)** בקיצור (DB) תוכנה המשמשת לאחסון של נתונים מכל סוג שהוא במחשב, לשם אחזרה ועיבודה. לתוכנה זו יש מודלים תכנותיים קבועים מראש, שמקילים על העבודה עם המידע, כמו מנגנונים פנימיים למיון וחיפוש, מנגנון אבטחה ושפה פנימית לקריאת, הוספת, מחיקת ועידכון נתונים.

# ניהול נתונים – קבצים מול מסד נתונים

## קבצים מול מסד נתונים

- כיום רוב מערכות המידע שומרות מידע במסד נתונים, בעבר המידע נשמר בקבצים.
- מסד נתונים (או בסיס נתונים DataBase בקיצור DB) - תוכנה המשמשת לאחסון וניהול של נתונים, משמשת כמחסן של מידע.
- מסד נתונים מגדיר אוסף של סוגי נתונים שונים.
- מסד נתונים מאפשר שיתוף נתונים בין יישומים השונים.
- במסד נתונים המידע מחולק ליישויות הנקראות טבלאות, בין הטבלאות השונות מתקיימת מערכת עניפה של קשרים.



# ניהול נתונים – קבצים מול מסד נתונים

## קבצים מול מסד נתונים

בעבר ניהול המידע היה בקבצים, במערכות אלו נוצרו בעיות רבות :

- לכל יישום היו קבצים נפרדים, לא היה קשר בין הקבצים.
- תהליך האיתור ושליפת המידע מהקובץ היו מסורבלים, על מנת לפתור בעיה זו נוספו קבצי אינדקס שמטרתם לייעל את תהליך שליפה המידע .
- כפילות מידע – אותו מידע הוגדר במספר מקומות ויצר בעיה בעדכון , כל שינוי דרש עדכון של כל המופעים , אחרת עלול להיווצר מצב שיהיו שתי גרסאות של נתונים כתוצאה מתהליכי עדכון ועיבוד נפרדים.
- לא היה ניהול אבטחה רציני.
- כל התהליך דרש כתיבת קוד רב.

# ניהול נתונים – קבצים מול מסד נתונים

## קבצים מול מסד נתונים

מסד הנתונים הינו מבנה מתקדם יותר מקובץ או מאוסף של קבצים:

- המידע מחולק ליישויות הנקראת טבלאות, בין הטבלאות מתקיימת מערכת קשרים עניפה.
- מגדיר שפה מיוחדת (שפת SQL) המקילה מאוד על ניהול הנתונים.
- המפתח אינו צריך לדאוג לניהול המידע, מתבצע באופן אוטומטי על ידי מסד הנתונים.
- שמירה על תקינות הנתונים באמצעות מערכת של חוקים, אילוצים וקשרים בין ובתוך הטבלאות השונות.
- רמת אבטחה גבוהה.
- עבודה עם מסדי נתונים גורשת כתיבת מעט קוד בהשוואה לקבצים.
- קל יותר לשתף נתונים בין יישומים שונים.

# ניהול נתונים – קבצים מול מסד נתונים

## קבצים מול מסד נתונים

דוגמה: מערכת לניהול הקורסים באוניברסיטה.

- המערכת מנהלת את הנתונים אודות הסטודנטים , המרצים והקורסים .
- בין סוגי נתונים אלה מתקיימים מספר קשרים , לדוגמא:
  - סטודנט לומד בקורס.
  - מרצה מלמד בקורס.
  - לסטודנט בקורס יש ציון.
- מימוש מערכת זו באמצעות קבצים דורשת להוסיף קבצים מיוחדים מעבר לקבצי הנתונים.
- מערכת לניהול קבצים לא מתייחסת לעובדה שקיימים קשרים בין קובצי הנתונים. לדוגמה: במערכת כזו ניתן יהיה לבטל נתוני סטודנט מבלי לבטל את העובדה שהוא לומד בקורס מסוים .
- מסד נתונים לא יאפשר זאת.

# ניהול נתונים – קבצים מול מסד נתונים

- מסד הנתונים מאפשר ריכוז של כל הנתונים במאגר מרכזי אחד . מסד הנתונים מיועד לשרת מספר רב של יישומים ולכן מאוחסנים הנתונים בצורה בלתי תלויה בתוכניות היישום המשתמשות בנתונים.
- מסד נתונים אחד אין פירושו שכל סוגי הנתונים יאוחסנו פיסית בקובץ אחד.
- ההבדל העיקרי בין אוסף קבצים המנוהלים על ידי מערכת קבצים לבין אוסף קבצים המנוהלים על ידי מערכת מסד נתונים , טמון בעובדה שמערכת לניהול מסדי נתונים "רואה" את כל הקבצים כאוסף אינטגרטיבי עם קשרים ומטפלת בניהול הקשרים ואמינות הנתונים.

# ניהול נתונים – קבצים מול מסד נתונים

- מערכת לניהול מסדי נתונים מכונה :  
(System Data Base Management) DBMS  
הינה מערכת תוכנה המאפשרת ליצור ולנהל את מסד הנתונים .
- מערכת לניהול מסדי נתונים מסייעת להגדיר, להקים ולהפעיל בסיסי נתונים לצורך יישומים שונים, והיא גם כוללת שפה לניהול המידע (SQL).
- במערכת לניהול מסדי נתונים יש הפרדה ברורה בין נתונים ותוכניות.
- מערכת לניהול מסדי נתונים הינה מערכת תוכנה המאפשרת גישה למספר רב של יישומים שונים בו זמנית.
- קיימות מערכות מסחריות רבות לניהול מסד נתונים ( Data Base Management System – DBMS ) . חלק גדול מהמערכות בהן משתמשים עבור מערכת מידע מבוסס על המודל הטבלאי ( Relational ) שאר המערכות מבוססות על המודל ההיררכי או המודל הרשתי.
- המודלים הללו מבטאים סוגי קשרים שונים בין הנתונים השונים.



# ניהול נתונים – קבצים מול מסד נתונים

- המודל הנפוץ ביותר היום, הוא המודל הטבלאי שמכונה גם יחסי (Relational).
- במודל זה המידע מחולק לטבלאות, כל טבלה מכילה מידע על ישות מסוימת (לדוגמה, סטודנטים במערכת לניהול אוניברסיטה).
- הטבלה מורכבת משדות (ת"ז, שם פרטי, שם משפחה, כתובת, טלפון וכו').
- לכל רשומה בטבלה יש שדה מפתח שמזהה באופן ייחודי את הרשומה (למשל, ת"ז).
- מוגדרים קשרים בין הטבלאות, לדוגמה: מוגדר קשר בין טבלת סטודנטים לטבלת קורסים, ולכן לא ניתן יהיה לרשום סטודנט לקורס שפרטיו אינם מופיעים בטבלת קורסים.
- שליפת מידע ופעולות עדכון בבסיס נתונים טבלאי נעשות באמצעות שפת שאילתות SQL שפה לטיפול ועיבוד מידע, המהווה ממשק המאפשר גישה לנתונים מבלי להתייחס לאופן שמירתם בבסיס הנתונים.

# ניהול נתונים – קבצים מול מסד נתונים

- החל משנות התשעים רב מערכות המידע החדשות מפותחות ופועלות סביב מערכות ניהול בסיסי נתונים טבלאיים – RDBMS (Relational DBMS) .
- מערכות אלו הפכו לטכנולוגיית ניהול הנתונים המועדפת עבור רב מערכות המידע המודרניות ומחסני נתונים (Data warehouse) ועבור שרתי אינטרנט (Web Servers) .

# ניהול נתונים – קבצים מול מסד נתונים

## מסדי נתונים – יתרונות וחסרונות

### יתרונות

- שיתוף נתונים – מערכות לניהול מסדי נתונים מאפשרת שיתוף מידע בין המחלקות/משתמשים בארגון .
- אמינות נתונים - מערכות לניהול מסדי נתונים מכילה מספר מנגנונים לשמירה על אמינות הנתונים , ניהול הקשרים בין הנתונים על ידי המערכת , יכולת הגדרת בדיקות תקינות באופן מרכזי , יכולת הפעלת שגרות באופן אוטומטי בהתרחש אירוע מסוים , רישום אוטומטי של יומן אירועים והפעלה של מנגנון למחיקת עדכונים כאשר יש תקלה כלשהי.
- זמינות נתונים - מערכות לניהול מסדי נתונים מספקת כלים מתוחכמים לשליפה ועדכון מסד הנתונים. מאפשרת להפוך במהירות ובפשטות נתונים גולמיים למידע
- ריכוזיות – כל הנתונים מוגדרים באופן מרכזי במסד נתונים אחד .

# ניהול נתונים – קבצים מול מסד נתונים

## מסדי נתונים – יתרונות וחסרונות

### יתרונות - המשך

- גמישות לשינויים – קל יותר להתאים את מסד הנתונים ואת תוכניות היישום המשתמשות בו לשינויים המתרחשים באופן שוטף בכל ארגון. העובדה שכל מבני הנתונים מוגדרים באופן חיצוני לתוכניות ותוכניות אלו יכולות לגשת למסד הנתונים מבטיחה רמה גבוהה יחסית של אי תלות לוגית ואי תלות פיזית.
- אבטחת נתונים - מערכות לניהול מסדי נתונים מכילות מספר מנגנונים לאבטחת הנתונים במסד הנתונים. במערכות אלו מאפשרות להגדיר הרשאות גישה למשתמשים.
- המערכת מכילה מנגנון רישום (Log) כל פעולה שנעשתה, מתי נעשתה ועל ידי מי, כך שניתן לשחזר את הפעולות שהתבצעו במסד הנתונים.
- תמיכה בעדכון בזמני - מערכות לניהול מסדי נתונים מכילות מנגנוני נעילת רשומות כך שלא יתבצע עדכון אותה רשומה בזמני על ידי מספר משתמשים.

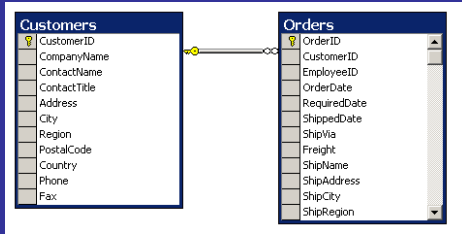
# ניהול נתונים – קבצים מול מסד נתונים

## מסדי נתונים – יתרונות וחסרונות

### חסרונות

- תפעול – מערכת לניהול מסדי נתונים הינה מערכת מורכבת יחסית הדורשת מיומנות רבה בתפעולה. מערכת לניהול מסדי נתונים בארגונים גדולים דורשת בדרך כלל צוות של אנשי מקצוע לתמיכה בפעולות המערכת.
- עלות – עלות מערכת לניהול מסדי נתונים הינה גבוהה יחסית.
- משאבי מערכת - מערכת לניהול מסדי נתונים דורשת יותר משאבי חומרה: זיכרון , CPU ושטח אחסון בדיסק מאשר מערכת לניהול קבצים.
- רגישות לתקלות – כל הנתונים של הארגון מרוכזים במסד נתונים ולכן תקלה יכולה לגרום להשבתה של רוב פעילות הארגון.
- מורכבות שחזור - מערכת לניהול מסדי נתונים מנהלת קשרים בין הנתונים בנוסף לנתונים עצמם, לכן שיקום מסד הנתונים לאחר תקלה בעייתי יותר.

# המודל הטבלאי



- טכניקת עיצוב הנתונים המודרנית והמקובלת מכונה מודל הנתונים היחסי ( Data Relational Model ) או מודל נתונים טבלאי.
- מודל זה החליף מודלים ישנים ומורכבים יותר להבנה שנקראו המודל ההיררכי והמודל הרשתי .
- המודל הוצג לראשונה בשנת 1970 על ידי ד"ר קוד ( Codd ) .
- קוד טען שהיתרון העיקרי של המודל הוא באפשרות הצגה מדויקת ופשוטה של מסד הנתונים .
- מודל הנתונים מתבסס על התורה המתמטית של הקבוצות ומשתמש באופרטורים לטיפול ביחסים ( Relations ) .
- הפשטות מושגת על ידי מימוש היחסים במבנה של טבלה ( Table ) , מבנה פשוט ומוכר לכל ועל ידי טיפול בטבלאות באמצעות מספר קטן של אופרטורים פשוטים .
- המודל הטבלאי מאופיין על ידי הפרדה ברורה בין המבנה הפיסי של הנתונים לבין המבנה הלוגי שלהם , כפי שהוא מוצג למשתמש.

# סוגים של בסיסי נתונים

- **Analytical database software** – מאפשר למשתמש למשוך נתונים ממגוון מאגרי מידע ולבחון אותם לצורך הערכה כמותית של ביצועים (עסקים, מחלקתיים, עובדים וכו').
- **Data warehouse software** – מאפשר למשתמש למשוך נתוני מפתח ממגוון מסדי נתונים ולאחסן אותם במיקום מרכזי לצורך הפקת תובנות קריטיות ודיווח על תובנות אלו. מחסן נתונים זה יישמש את הגורם הרלוונטי במקביל לבסיס נתונים מקורי ולא יפריע לתפקוד שלו.
- **Distributed database software** – בסיס נתונים מבוזר כולל מערכת ניהול מרכזית השולטת במידע המאוחסן במגוון מיקומים (כולל ענן, או שרת רשת). מודל מבוזר נחשב לעתים קרובות הבטוח ביותר בגלל יכולת לשמור עודפי מידע (מידע שמוכפל כמה מפעמים).
- **End user database software** – מסד נתונים של משתמשי הקצה מאחסן מידע המשמש בעיקר אדם אחד, כגון Microsoft Excel וגיליונות אלקטרוניים אחרים.
- **External database software** – מסד נתונים חיצוני חייב להיות נגיש למגוון רחב של משתמשים, לעתים קרובות באמצעות האינטרנט (למשך מסד נתונים בענן).
- **Operational database software** – תפעולי מאפשרת למשתמש לשנות נתונים בזמן אמת (למטרות כגון ניהול פיננסי וניהול קשרי לקוחות).

# המודל הטבלאי

- התכונות העיקריות של המודל הטבלאי הן פשטות תכנון המודל , גמישות ויכולת התמודדות עם שינויים במבנה ובניהול הנתונים ועוצמה בטיפול בנתונים. תכונות אלו הפכו את המודל למודל הנפוץ ביותר כיום ולמקובל ביותר בין אנשי המקצוע.
- המודל הטבלאי הינו מודל פורמלי לייצוג האובייקטים והקשרים ביניהם במערכות ממוחשבות.
- המודל הטבלאי מבוסס על ההנחה שכל הנתונים מיוצגים כיחס (Relation) או הטבלה (Table) .
- הגדרה :  
נתון אוסף של קבוצות –  $S_1, S_2, \dots, S_n$  לא בהכרח זרות . היחס (Relation) מוגדר כאוסף מסודר של איברים  $\{s_1, s_2, \dots, s_n\}$  , כך שכל איבר  $s_i$  מתאים לקבוצה  $D_j$  עבור  $j=1, 2, \dots, n$  . היחס יסומן על ידי  $R(D_1, D_2, \dots, D_n)$  כאשר  $R$  הוא שם היחס.
- ניתן לתאר את היחס כאוסף של שורות ועמודות או בפשטות טבלה . כל עמודה מייצגת תכונה כלשהי של האובייקט וכל שורה מייצגת אובייקט אחד בטבלה. הערך  $s_{ij}$  המופיע בשורה  $i$  ועמודה  $j$  , נלקח מהקבוצה  $S_j$  המייצגת את מרחב הערכים האפשרי של התכונה. עמודה מתאימה לשדה ושורה מתאימה לרשומה וטבלה לקובץ.



# המודל הטבילאי

- אחת התרומות החשובות של Codd היתה בהבחנה שקיים דמיון רב בין טבילה (או קובץ) לבין הקבוצה המתמטית.
- מכיוון שהערכים המופיעים נלקחים מתוך מרחב הערכים האפשרי של התכונה, הרי שנוכל להגדיר קבוצה דמיונית הנוצרת בעקבות המכפלה הקרטזית של מרחבי הערכים השונים. מתוך קבוצה דמיונית זו, נשלוף רק שורות מסוימות, לקבוצה המתקבלת נקרא בשם יחס.

# המודל הטבלי

טבלת Persons

|   | PersonID | LastName | FirstName | Birthdate  |
|---|----------|----------|-----------|------------|
|   | 123      | מתלון    | רון       | 10/02/1991 |
|   | 124      | זורקין   | אלימלך    | 24/02/1945 |
|   | 125      | מרגרינה  | רינה      | 14/06/1985 |
|   | 126      | אגלי     | יפה       | 05/01/1965 |
|   | 127      | סתמשם    | זלדה      | 15/08/1974 |
| ▶ |          |          |           |            |

שדות

רשומות

עמודות  
(Columns)

# המודל הטבלאי

- מספר השדות קבוע.
- סדר השדות אינו חשוב.
- מספר הרשומות אינו מוגבל.
- סדר הרשומות אינו חשוב.
- לכל טבלה חייב להיות מפתח חד ערכי הנקרא מפתח ראשי (primary key) ..
- המפתח יכול להיות תכונה או הרכב של מספר תכונות.


# המודל הטבלאי

- לכל טבלה חייב להיות מפתח חד ערכי הנקרא מפתח ראשי (primary key), תפקידו לזהות את הרשומה באופן חד ערכי:

יתכן שבטבלת Persons ישמרו פרטי מספר אנשים ששמן אלי כהן, כולם גרים בתל אביב, ויתכן גם שכולם או חלקם נולדו באותו היום.

כיצד ניתן יהיה להבדיל ביניהם? , באמצעות מספר הזהות המוגדר כמפתח ראשי ולכן לא יכול להופיע פעמיים או יותר באותה הטבלה:

| Person  |              |
|---|--------------|
|  | PersonID     |
|   | LastName     |
|   | FirstName    |
|   | Birthdate    |
|   | PhoneNum     |
|   | EmailAddress |
|   | WebSite      |

|   | PersonID | LastName | FirstName | Birthdate  |
|---|----------|----------|-----------|------------|
|   | 123      | כהן      | אלי       | 10/10/1950 |
|   | 124      | כהן      | אלי       | 10/10/1950 |
|   | 125      | כהן      | אלי       | 10/10/1950 |
|   | 126      | כהן      | אלי       | 10/10/1950 |
|  | 127      | כהן      | אלי       | 10/10/1950 |
| *   |          |          |           |            |

- המפתח יכול להיות שדה בודד או הרכב של מספר שדות.

# המודל הטבלאי

- **ישות** – פרט ששומרים עליו נתונים במערכת המידע , אובייקט או עצם מהעולם הממשי.  
דוגמא: תלמיד בבית ספר, מורה , כיתה וכד' הם ישויות.
- **קבוצת ישויות** – אוסף של ישויות.
- **שדה** – כל פריט מידע ששומרים על ישות , מאפיין של ישות.  
דוגמא : תעודת זהות , שם , כתובת , מקצועות לימוד הם תכונות של הישות מורה.
- **שדה מורכב** – שדה שניתן לפרק לתכונות פשוטות יותר.  
דוגמא : את השדה כתובת ניתן לפרק לעיר , רחוב , מספר , מיקוד . ההחלטה אם לשמור שדה כמורכב (כל הכתובת כשדה טקסט או לפרק אותה , תלויה בצרכי מערכת המידע).

# המודל הטבלאי

- **שדה מרובה ערכים** – שדה שיכול לקבל בו זמנית קבוצה של ערכים. דוגמא : מקצועות לימוד של מורה. מורה יכול ללמד מספר מקצועות. במודל הטבלאי אין אפשרות להגדיר שדה מרובה ערכים ולכן הפתרון המקובל הוא להתייחס אליו כאל ישות ולייצג אותו באמצעות טבלה.

**דוגמא :**

טבלת מקצועות למורה

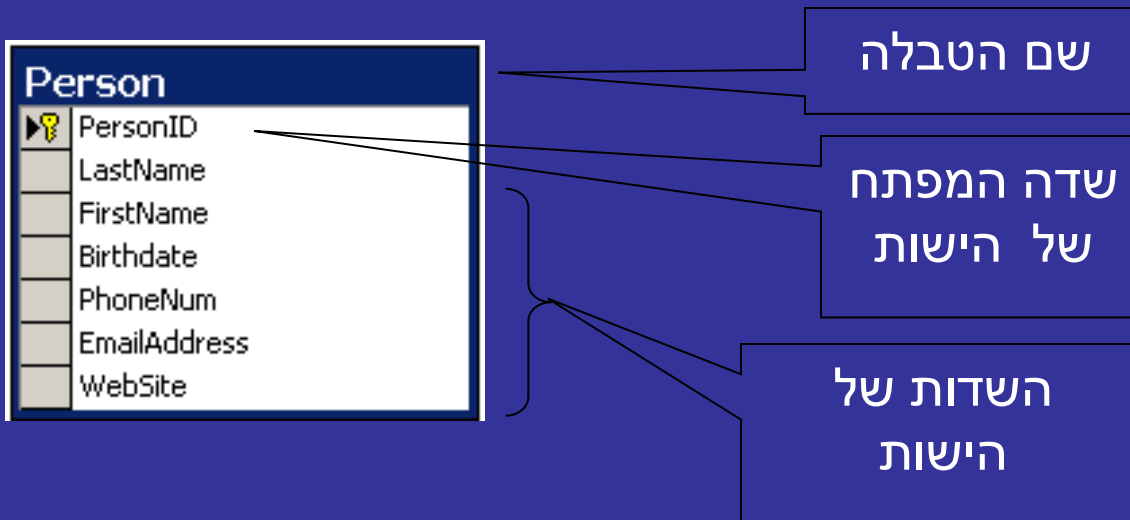
| ..... | מקצוע | תעודת זהות |
|-------|-------|------------|
|       | ספרות | 055874595  |
|       | לשון  | 055874595  |
|       | חיבור | 055874595  |

טבלת מורים

| ..... | שם       | תעודת זהות |
|-------|----------|------------|
|       | רוני לוי | 055874595  |
|       | בר רונן  | 302154684  |
|       | גיא פלד  | 512954454  |

# המודל הטבלי

- תרשים מבנה הנתונים (Data Structure Diagram - DSD) הוא כלי גרפי המתאר את מבנה הנתונים (ואת הקשרים ביניהם) שיש לאחסן במסד הנתונים.
- התיאור של טבלת "סטודנטים" בתרשים DSD.
- טבלה, במקרה הפשוט, מייצגת יישות במציאות.
- במקרים מורכבים יותר יישות במציאות מיוצגת באמצעות מספר טבלאות.

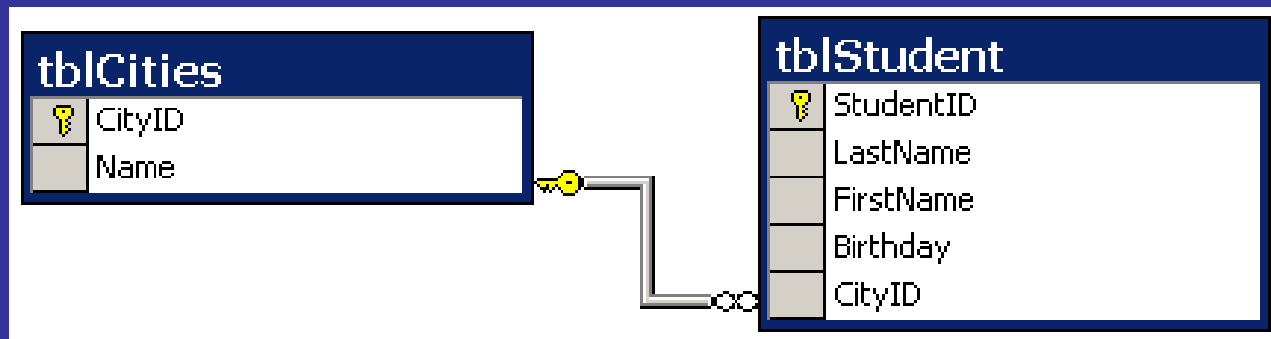


# המודל הטבלאי

- מסד נתונים מורכב מטבלאות וקשרים ביניהם.
- את הקשר בין הישויות מציגים על ידי קו המחבר את השדות המתאימים של הישויות .

דוגמא : בטבלת סטודנטים מתארים כל סטודנט באמצעות מספר סטודנט, שם פרטי, שם משפחה, תאריך לידה ועיר מגורים.

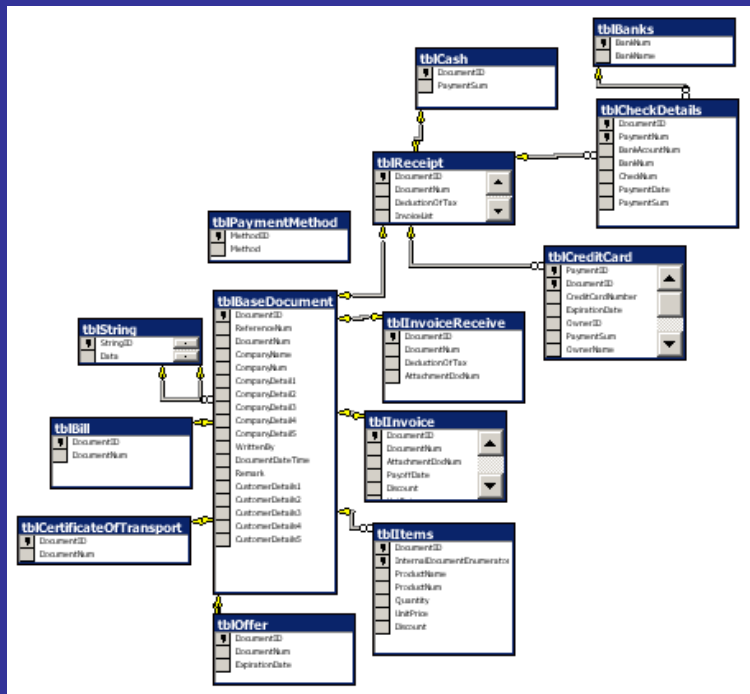
מסיבות שндון בהם מאוחר יותר, לא נגדיר עיר כשדה, אלא נגדיר טבלת ערים, ונקשר אותה לטבלת סטודנטים באמצעות שדה מפתח:





# המודל הטבלאי

1. בניית רשימת הטבלאות – מזהים את הישויות הרלוונטיות למערכת הנבדקת ונותנים שם לכל ישות.
2. הגדרת השדות של כל טבלה (כולל מפתחות).
3. זיהוי הקשרים בין הטבלאות.
4. בדיקת התרשים.



# המודל הטבילאי

- קשר - יחס בין שתי טבלאות או בין טבלה לעצמה.

דוגמא : בין תלמידים לכיתות קיים קשר של לומד ב- . (תלמיד x



בין מורים למקצועות קיים קשר של יכול ללמד (מורה x יכול ללמד



מקצוע y)

- יכול להיות שבין שתי ישויות יש יותר מקשר אחד.

- דוגמא : בין מורים לכיתות קיימים מספר קשרים:

מורה x מלמד ב כיתה y

מורה x מחנך של כיתה y

- יכול להיות שקבוצת ישויות קשורה אל עצמה .

דוגמא : מקצוע x הוא דרישת קדם ל מקצוע y



# המודל הטבילאי

1. יחיד ליחיד (1-1) :

כל רשומה מטבלה A קשורה לכל היותר לרשומה אחת מקבוצה B  
וכל רשומה מטבלה B קשורה לכל היותר לרשומה אחת מטבלה A  
דוגמא: הקשר "מחנך של"

מורה יכול לחנך כיתה אחת בלבד  
לכל כיתה יש מחנך אחד בלבד

tblTeachers



tblClasses

# המודל הטבלאי

2. קשר יחיד לרבים ( $M - 1$ ):

כל רשומה מטבלה A קשורה למספר רשומות מטבלה B ,  
כל רשומה מטבלה B יכולה להיות קשורה לכל היותר לרשומה  
אחת מטבלה A .

דוגמא: הקשר "לומד ב" בין תלמידים לכיתות,

תלמיד לומד בכיתה אחת בלבד.

בכל כיתה לומדים מספר תלמידים

tblStudent



tblClasses

# המודל הטבלאי

3. קשר רבים לרבים ( $M - N$ ) :

כל רשומה מטבלה A קשורה למספר רשומות מטבלה B, וכל רשומה מטבלה B יכולה להיות קשורה למספר רשומות מטבלה A.

דוגמא: הקשר "מלמד" בין מורים ומקצועות

מורה יכול ללמד מספר מקצועות

מקצוע מסוים יכול להילמד על-ידי כמה מורים.

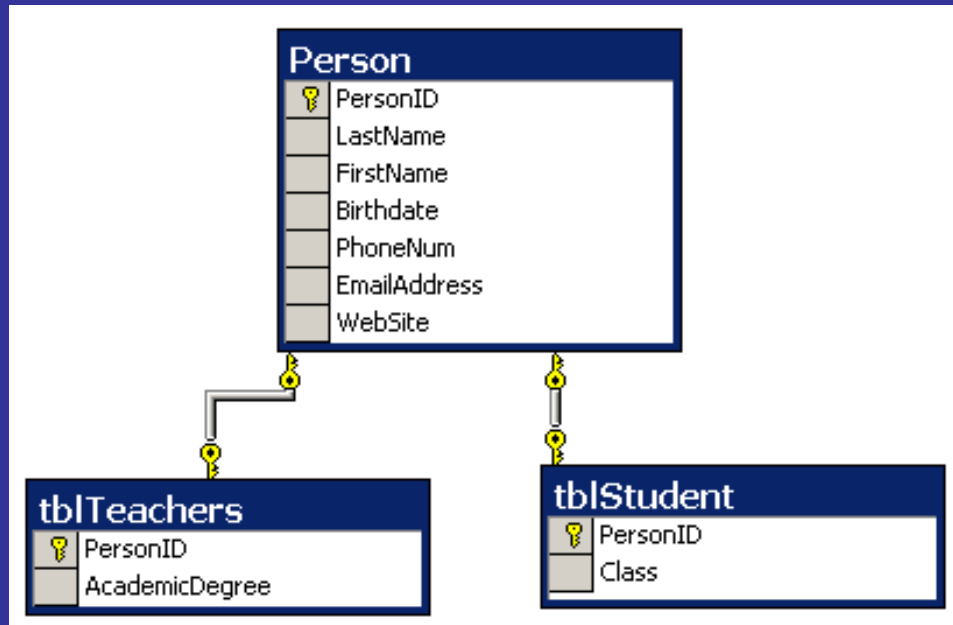
הערה : קשר רבים לרבים ממומש על ידי טבלת קשר המכילה את המפתחות הראשיים של שתי הטבלאות, ועל כך בהמשך.

# המודל הטבלאי

חלק מן הטבלאות במסד הנתונים חולקות לעיתים שדות משותפים. לדוגמא: למרצים ולסטודנטים יש פרטי מידע רבים משותפים : מספר מזהה , שם פרטי , שם משפחה , כתובת ועוד. אולם, בנוסף יש לישויות הללו יש גם תכונות ייחודיות שאינן משותפות , לכן אפשר להתייחס לכל אחת מהן כתת יישות של היישות המשותפת "אנשים" .



# המודל הטבילאי



הערות :

- כל תת ישות תלויה בישות האם שלה ולכולן מפתח זהה.
- כל תת ישות מקבלת את התכונות המשותפות מישות האם שלה.
- כאשר בתרשים ה-DSD יש הרבה ישויות וקשרים, רצוי להציג את הקשרים ההיררכיים בנפרד מהמקשרים האחרים.

# המודל הטבלאי

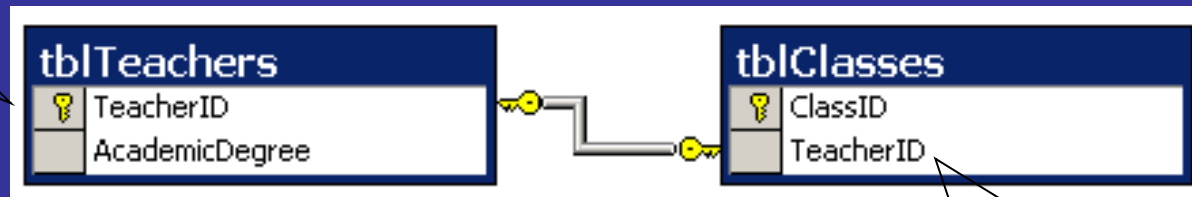
קשר יחיד ליחיד (1-1) מיוצג באמצעות מפתח זר (Foreign Key).

דוגמא:

כדי לייצג את הקשר "מחנך של" בין מורה לכיתה מוסיפים את התכונה מספר זהות של מחנך לטבלת הכיתות.

בטבלת מורים השדה מספר זהות הוא המפתח הראשי (Primary Key), בטבלת הכיתות מספר זהות של המורה הוא מפתח זר (Foreign Key). הקשר בין המפתחות הללו מגדיר את הקשר.

מפתח ראשי



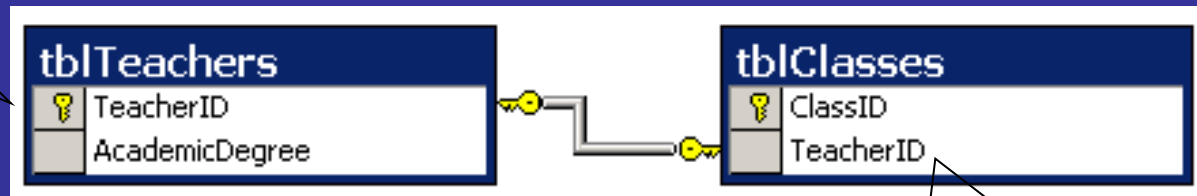
מפתח זר (ייחודי)



# המודל הטבלאי

כדי ליצור את הקשר הוסף את המפתח הראשי של אחת הטבלאות (A) כשדה (או שדות) לטבלה השנייה (B).  
שדה זה (או צירוף שדות) מוגדר כמפתח זר ב – B אשר מיוחס למפתח הראשי של – A.  
המפתח הזר יכול גם להיות מפתח ראשי, או לפחות שדה ייחודי (Unique).  
הטיפוסים של השדות חייבים להיות זהים.

מפתח ראשי

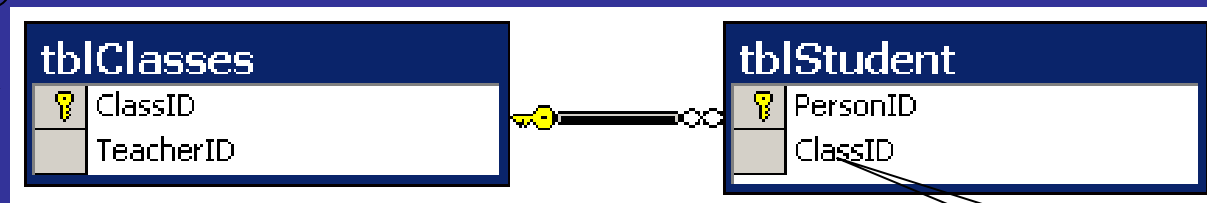


מפתח זר (ייחודי)

# המודל הטבלאי

קשר יחיד לרבים מיוצג גם הוא על ידי קשר יחיד ליחיד (1-1) מיוצג באמצעות מפתח זר (Foreign Key).  
להבדיל מיחיד ליחיד המפתח הזר לא יהיה מפתח ולא שדה ייחודי.  
דוגמא: הקשר תלמיד "לומד ב" כיתה , או , בכיתה "לומדים הרבה" תלמידים.

מפתח ראשי



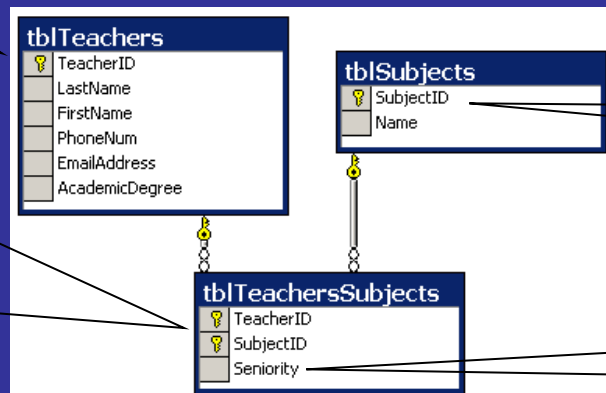
מפתח זר (לא ייחודי)

# המודל הטבלי

קשר רבים לרבים מורכב יותר משני האחרונים, הוא ממומש באמצעות מפתחות אולם בתוספת טבלה שלישית המכונה טבלת קשר. טבלת הקשר תכיל את המפתחות הראשיים של שתי קבוצות הישויות וכן את תכונות הקשר.

**דוגמא : הקשר "מלמד" בין מורה למקצועות, כל מורה יכול ללמד מספר מקצועות.**

## במקרה זה ניצור טבלת קשר טבלת מורים-מקצועות



## מפתח ראשי

המפתחות הזרים  
מרכיבים את  
המפתח של  
טבלת הקשר

## מפתח ראשי

שדה נוסף  
לתיאור הקשר

# NoSQL – בסיסי נתונים לא רלציוניים

- מדובר במסד נתונים בעל נפחים גדולים, השומר מידע בכמויות עצומות, הנכנס במהירות גבוהה מאוד.
- מידע זה אינו מאורגן לפי שיטה כלשהי (**Unstructured**), הוא מגוון מאוד כך שלא ניתן לסדר אותו בטבלאות. דוגמאות לסוגי מידע שכאלה הינם: לייקים ושיתופים ברשתות חברתיות, תגובות של גולשים, תמונות, מידע מסנסורים במכשור טכנולוגי מתקדם ועוד. מידע מסוג כזה הינו מידע שמעמיד כל ארגון בפני אתר **Big Data** מסדי נתונים שבאים כפתרון מכונים בשם משותף. **NoSQL** –
- מסדי נתונים **NoSQL** הופיעו כחלופה פופולרית למסדי נתונים רלציוניים כאשר יישומי אינטרנט הפכו למורכבים יותר ויותר.
- ההבדל הקריטי בין מסדי נתונים של **NoSQL** ומסדי נתונים רלציוניים הוא שתוכניות **RDBMS** מגדירות באופן נוקשה את האופן שבו כל הנתונים שהוכנסו למסד הנתונים חייבים להיות מוקלדים ומורכבים, ואילו מסדי נתונים של **NoSQL** מבוססים סכימת הנתונים בכל צורה שהיא, ומאפשרים לאחסן ולטפל בנתונים בלתי מובנים ומובנים למחצה.
- בסיסי נתונים **NoSQL** מתחלקים לכמה משפחות עיקריות, כאשר כל אחת מאופיינת במטרות וביצועים שונים.

# NoSQL – בסיסי נתונים לא רלציוניים

## • Key-Value Stores

- לדוגמא Redis ו: Amazon DynamoDB - הן מערכות פשוטות לניהול מסדי נתונים שמאחסנות זוגות של ערכי מפתח ומספקות פונקציונליות בסיסית לאחזור הערך המשויך למפתח ידוע.
- הפשטות בשמירת נתונים באופן הזה הופכת את מערכות ניהול מסדי נתונים אלה למותאמות היטב ל- embedded databases, הנטונים המאוחסנים אינם מורכבים במיוחד ומהירותם היא בעלת חשיבות עליונה.

## • Wide Column Stores

- לדוגמא Cassandra, Scylla ו: HBase - הן מערכות המאפשרות למשתמשים לאחסן נתונים במשפחות של עמודות או טבלאות, שורה אחת יכולה להיחשב כרשומה בפני עצמה כך שיש כאן Key-Value Stores רב מימדי. פתרונות אלה מתוכננים במטרה להרחיב את קנה המידה של ניהול הנתונים על פני מספר רב של שרתים במערכת מסיבית ומבוזרת.

## • Document Stores

- לדוגמא MongoDB ו: Couchbase - הן מערכות שלא בנויות על סכימה אלא מאחסנות נתונים בצורה של מסמכים. מסדי נתונים אלה דומים לשניים הקודמים, אך כאן המסמך הוא המפתח והתוכן של המסמך, באשר הוא, הוא הערך.

# **Definition of Data Warehouse**

A data warehouse is constructing by integrating data from multiple heterogeneous sources that support analytical reporting, structured and queries, and decision making.

# Examples of Data stored In Data warehouse

The data stored in the warehouse is uploading from customer information from a company's point-of-sale, information collected for a research paper, Qualitative data, Quantitative data, marketing or sales data, aeronautic Data ,weather data, Law and regulations etc.

Now! we fully understand that leverage the data we are collecting has become more and more apparent.

# Using Data Warehouse Information

- **Production Strategies** – Repositioning and managing the product portfolios by comparing the sales quarterly or yearly.
- **Customer Analysis** – Analyzing the customer's buying preferences, buying time, budget cycles, etc.
- **Operations Analysis** – customer relationship management, making environmental corrections and analyze business operations.



# Tools and Utilities of Data Warehouse

The following are the functions of data warehouse tools and utilities –

- Extraction** – Involves gathering data from multiple heterogeneous sources.
- Cleaning** – Involves finding and correcting the errors in data.
- Transformation** – Involves converting the data from legacy format to warehouse format.

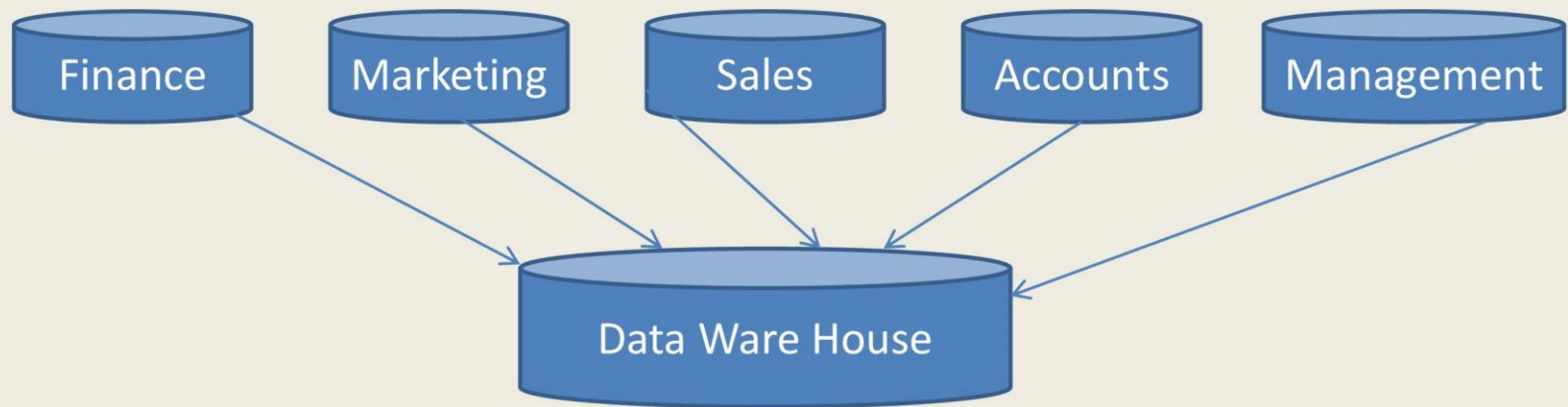
- Loading** – Involves sorting, summarizing, consolidating, checking integrity, and building indices and partitions.

- Refreshing** – Involves updating from data sources to warehouse.

Data cleaning and transformation are important steps in improving the quality of data. All data loaded into the data warehouse would have to be converted to use this standard format is called **Extraction-Transformation-Load (ETL)**.

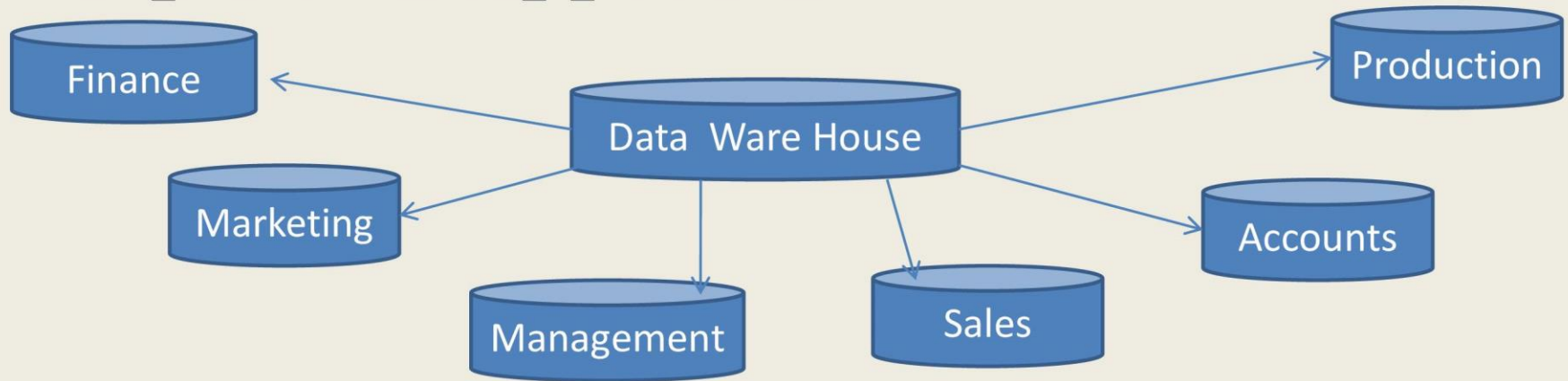
# Designing a data warehouse

## Bottom-Up Approach



The **Bottom-Up Approach** creating small data marts, to solve specific business problems. As these data marts can be combined into a larger data warehouse.

# Top-Down Approach



The **Top-Down Approach** suggests that start by creating an enterprise-wide data warehouse and then, as specific business needs are identified, create smaller data marts.