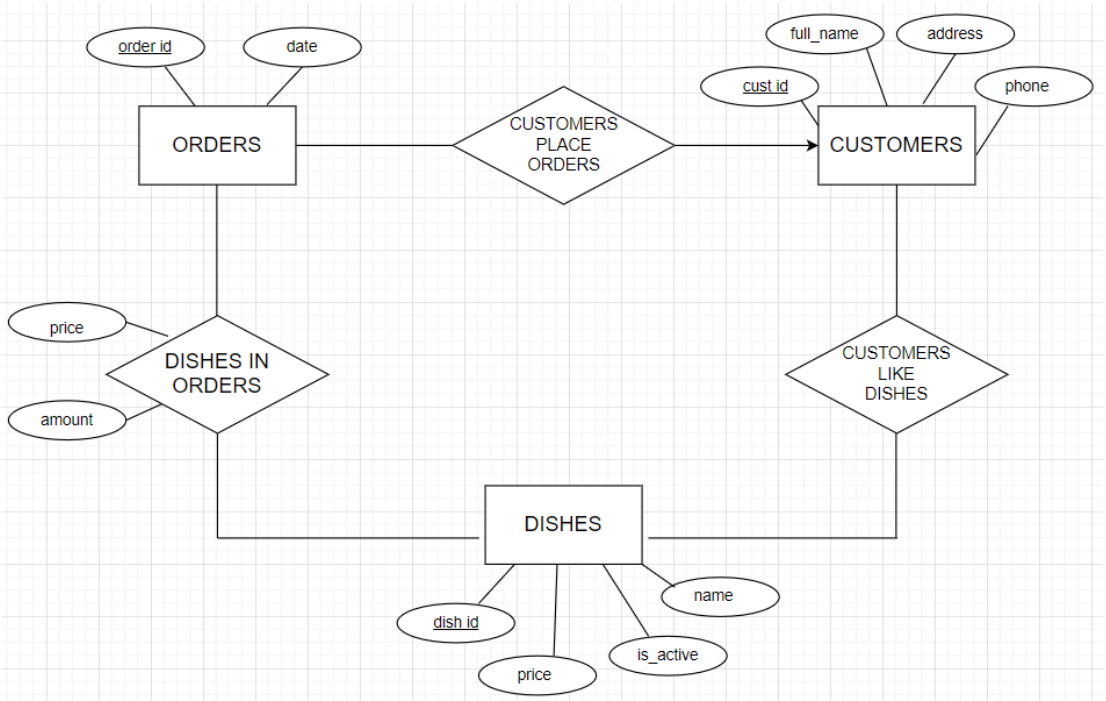
יובל צוברי

אסתר(אתי) חיה רווח

מסדי נתונים מטלה 2 – חלק יבש

ERD:



הסבר:

ב-DB שלנו נשמור על 3 טבלאות ראשיות:

1. לקוחות: ת"ז – primary key, שם, כתובת וטלפון. ביצירת הטבלה נבדק שהת"ז >0 וכן שכל הפרמטרים אינם NULL כמו כן נבדק אם הכתובת לפחות בגודל 3 תווים.
2. הזמנות: ת"ז– primary key, תאריך. ביצירת הטבלה נבדק שהת"ז >0 ואינו NULL.
3. מנות: : ת"ז – primary key, שם, האם פעילה, מחיר עדכני. ביצירת הטבלה נבדק שהת"ז >0 , שהשדות אינם NULL וכן ששם המנה הינו לפחות 3 תווים.

וכן נשמור על 3 טבלאות המייצגות את היחסים ביניהם:

1. מנות בהזמנות (DISHES_IN_ORDERS) – שומר את המידע על המנות שהוכנסו עם ההזמנות עם המחיר שלהם בזמן ההזמנה וכן כמות מהמנה. מנה לא תופיע בהזמנה יותר מפעם אחת ולכן גם ת"ז הזמנה וגם ת"ז מנה (foreign key) ישמשו יחד כ-primary key. מחיר המנה נשמר נכון לזמן ההזמנה.
2. הזמנות של לקוחות (CUSTOMERS_PLACE_ORDERS) – מקשר בין הלקוח להזמנה שהזמין, לקוח יכול להזמין הרבה הזמנות, אך הזמנה תופיע ביחס רק עם לקוח יחיד – many to one. ת"ז לקוח ות"ז מנה הם foreign key ות"ז מנה משמש כ-primary key.
3. לקוחות שאהבו הזמנות (CUSTOMERS_LIKE_DISHES) – ישמור על המצב הנוכחי של אילו מנות לקוחות אהבו. רישום כזה יופיע רק פעם אחת ולכן ת"ז לקוח ות"ז מנה (שהם foreign key) ישמשו יחד כ-primary key.

- בכל היחסים יכנס מידע על בסיס המידע הקיים כבר ב-DB – כלומר לדוגמא לא תתווסף הזמנה לאחד היחסים עם הטבלה אם היא אינה מופיע בטבלה המקורית.

VIEWS:

1. ACTIVE_DISHES_VIEW: מראה את המנות שהן פעילות כרגע, מסתמך על טבלת DISHES.
2. CUSTOMERS_ORDERS_TOTAL_PRICE_VIEW: לצד ת"ז הזמנה ות"ז לקוח, מראה את הסכום הכולל של מנה, נבצע JOIN בין DISHES_IN_ORDERS ו-CUSTOMERS_PLACE_ORDERS, חישוב המחיר הכולל נעשה באמצעות הכפלת כמות במחיר מנה וכן ביצוע GROUP BY על ההזמנה ב"מנות בהזמנות" וכן על מזהה הלקוח ב"הזמנות של לקוחות". אם לא נוספו מנות להזמנה, נקבל ערך דיפולטי 0.
3. MOST_LIKED_DISHES_RANKING_VIEW: מכיל את כל ה- dish_id של המנות הקיימות (הנמצאות בטבלה DISHES) יחד עם סך ה-likes שהם קיבלו מלקוחות. המידע על כמות ה-likes התקבל מהטבלה CUSTOMERS_LIKE_DISHES , פונקציית COUNT על כמות לקוחות ו-GROUP BY על ת"ז המנה (כאשר מתייחסים למנה שלא נמצאת ב- CUSTOMERS_LIKE_DISHES כמנה שקיבלה 0 likes). ה-view נועד לצורך מציאת 5 המנות האהובות ביותר.
4. MOST_LIKED_DISH_VIEW: view שכל רשומה בו מכילה dish_id ו-amount_likes כאשר מידע זה מתקבל מ MOST_LIKED_DISHES_RANKING_VIEW אבל עובר סינון ומשאיר רק רשומות בהם כמות ה-likes גדולה מ-0, ה-view נועד למציאת המנה האהובה ביותר מהמנות שלקוחות אהבו.
5. MOST_PURCHASED_DISH_VIEW: view שכל רשומה בו מכילה dish_id וכמות מנות שנרכשו (purchased_amount) שמחושבת בעזרת פונקציית SUM על amount ו-GROUP BY לפי dish_id, מתבסס על מידע שנמצא בטבלה DISHES_IN_ORDERS ונועד כדי למצוא את המנה שנרכשה הכי הרבה.
6. PROFIT_PER_MONTH_VIEW: view שכל רשומה בו מכילה year, month , profit. השנה והחודש נלקחים מהטבלה ORDERS משדה date והרווח מחושב לפי מידע שנמצא ב-DISHES_IN_ORDERS (כמות המנה שנרכשה כפול מחיר המנה בעת יצירת ההזמנה, מקובץ בעזרת GROUP BY לפי השנה והחודש כדי לקבל את סך הרווח מההזמנות שהתבצעו בזמן זה, כאשר בהזמנה שלא רכשה מנות הרווח הדיפולטיבי זה 0).
7. MONTHS_VIEW: view שמכיל את חודשי השנה, לצורך הפונקציה get_total_profit_per_month כדי שחודשים שלא היו בהם הזמנות יילקחו בחשבון ויקבלו רווח אפס.
8. SIMILAR_CUSTOMERS_VIEW: view שכל רשומה בו מכילה שני cust_id שמהווים לקוחות דומים. כל לקוח שה-id שלו נמצא ב-CUSTOMERS_LIKE_DISH נמצא בטבלה כ-customer וכל הלקוחות שיש להם לפחות 3 מנות שונות שהם אהבו וגם לקוח זה אהב, נמצאים איתו כ-similar_customer ברשומות שונות, זה נעשה בעזרת יצירת כל הזוגות האפשריים (כאשר זה יוצר רשומות של 2 id של לקוחות ו2 id של מנות- לפי המנה שהייתה איתם ברשומות שאחודו מ-CUSTOMERS_LIKE_DISHES) וקיבוץ הזוגות לפי ה-id-ים כאשר שומרים רק רשומות בהם כמות המנות השונות שנמצאו עם זוגות אלו זה לפחות 3.
9. ORDERED_DISHES_PROFIT_VIEW: מראה את ה-average_profit כפי שתואר במטלה, של כל מנה לכל מחיר שנכנסה למנה איתו. מסתמך על טבלת DISHES_IN_ORDERS
10. ACTIVE_ORDERED_DISHES_CURRENT_PROFIT_VIEW: מראה את ה-average _profit כפי שתואר במטלה, של כל מנה פעילה על פי המחיר הנוכחי שלה.

פונקציות ושאילתות:

עבור הפונקציות הבאות תקינות הקלט נבדקת בבניית הטבלה כפי שצויין קודם. כאשר השאילתא תקבל חריגות DB ישלחו ערכי חזרה מתאימים:

- CHECK_VIOLATION, NOT_NULL_VIOLATION

ReturnValue.BAD_PARAMS בהכנסה,

ReturnValue.NOT_EXISTS בשליפה או מחיקה

ReturnValue.ALREADY_EXISTS - UNIQUE_VIOLATION

1. **add_customer:** הוספת לקוח לטבלת CUSTOMERS , בעזרת INSERT INTO יכניס את הלקוח המתקבל לטבלה תוך בדיקת תקינות קלט בדרך שצוינה קודם.
2. **get_customer:** החזרת הלקוח אליו שייך ה-ID המתקבל, תוך בדיקת קלט שמתבצעת בטבלה כפי שצוין קודם. בעזרת SELECT נבחר את מאפייני הלקוח לפי ID בעזרת FROM מטבלת ה-customers ונשתמש בWHERE לבחירת הלקוח הנכון. נבנה את אובייקט הלקוח ונחזירו.
3. **delete_customer:** מחיקת לקוח מטבלת CUSTOMERS בעזרת DELETE FROM כאשר נשתמש ב-WHERE על מנת למצוא את הרשומה עם הלקוח המתאים ולמחוק אותה אם קיימת תוך בדיקת תקינות קלט כפי שצוין קודם.
4. **add_order:** הוספת הזמנה לטבלת ORDERS, בעזרת INSERT INTO יכניס את ההזמנה המתקבלת לטבלה תוך בדיקת תקינות קלט כפי שצוין קודם.
5. **get_order:** החזרת ההזמנה אליה שייך ה-ID המתקבל, תוך בדיקת קלט שמתבצע בטבלה כפי שצוין קודם. בעזרת SELECT נבחר את מאפייני ההזמנה לפי ID בעזרת FROM מטבלת ה-ORDERS ונשתמש בWHERE לבחירת ההזמנה הנכונה. נבנה את אובייקט ההזמנה ונחזירו.
6. **delete_order:** מחיקת הזמנה מטבלת orders בעזרת DELETE FROM כאשר נשתמש ב-WHERE על מנת למצוא את הרשומה עם ההזמנה המתאימה ולמחוק אותה אם קיימת תוך בדיקת תקינות קלט שצוינה קודם.
7. **add_dish:** הוספת מנה לטבלת DISHES, בעזרת INSERT INTO יכניס את המנה המתקבלת לטבלה תוך בדיקת תקינות קלט שצוינה קודם.
8. **get_dish:** החזרת המנה אליה שייך ה-ID המתקבל, תוך בדיקת קלט שמתבצעת בטבלה כפי שצוינה קודם. בעזרת SELECT נבחר את מאפייני המנה לפי ID בעזרת FROM מטבלת ה-DISHES ונשתמש בWHERE לבחירת המנה הנכונה. נבנה את אובייקט המנה ונחזירו.
9. **update_dish_active_status:** מעדכן בעזרת UPDATE בטבלת DISHES את הרשומה בעלת ה-ID שתתקבל תוך שימוש בWHERE על ה-id להיות עם מצב is_active שניתן לפונקציה.
10. **update_dish_price:** מעדכן בעזרת UPDATE את מחיר המנה ברשומה בטבלה DISHES בעלת ה-ID המתאים שתתקבל תוך שימוש ב WHERE על ה-ID, תוך בדיקה בשאילתא אם המחיר שהתקבל גדול מ0 וכן שהמנה פעילה.
11. **customer_placed_order:** הוספת התאמה בין הזמנה ללקוח לטבלת CUSTOMERS_PLACE_ORDERS, בעזרת INSERT INTO יכניס את ההזמנה המתקבלת לטבלה תוך בדיקת תקינות קלט שצוינה קודם וכן בדיקת קיימות הלקוח וההזמנה לפי איך הטבלה מושכת את המפתחות מטבלאות "לקוחות" והזמנות" – (foreign key).

12. **get_customer_that_placed_order:** נבצע SELECT על הטבלה "הזמנות של לקוחות" ונשתמש ב-WHERE על מנת למצוא את הרשומה עם מזהה הלקוח, ובאמצעות מזהה זה נבצעי SELECT נוסף, הפעם מטבלת "לקוחות" לבחירת המידע השלם על הלקוח לפי המזהה שלו. באי מציאה של בקשת השליפה ישלח אובייקט מסוג BadCustomer, במציאה נבנה אובייקט לקוח ונחזירו.
13. **order_contains_dish:** הוספת מנה להזמנה ותיעוד לכך בטבלת DISHES_IN_ORDERS. בעזרת ACTIVE_DISHES_VIEW נבדוק אם המנה קיימת ופעילה ורק אם כן נקבל ערכים מה-SELECT שנכניס ל-DISHES_IN_ORDERS. כמו כן עקב בניית הטבלה על בסיס טבלאות "הזמנות" ו"מנות" לא יתאפשר להכניס מנה או הזמנה שאינם קיימות כבר ב-DB.
14. **order_does_not_contain_dish:** מסירה מנה מהזמנה באמצעות שימוש ב- DELETE FROM DISHES_IN_ORDERS. המנה מוסרת רק אם קיימת בהזמנה כפי שמתועד בטבלה "מנות בהזמנות". נמצא את הרשומה המתאימה לפי שני המפתחות, ת"ז המנה ות"ז ההזמנה.
15. **get_all_order_items**: נשתמש ב-SELECT על טבלת "מנות בהזמנות" לבחירת המידע הנדרש על המנות הנמצאות יחד עם מזהה ההזמנה. אם אין מנות כאלו או הזמנה כזו תוחזר רשימה ריקה. בעזרת ORDER BY dish_id ASC נקבל את מזהי המנות בסדר עולה.
16. **customer_likes_dish:** נוסיף לטבלת CUSTOMERS_LIKE_DISHES את הצימוד המתקבל של מזהה מנה ומזהה לקוח, בעזרת INSERT תוך בדיקת תקינות קלט שצוינה קודם המתבססת על קיום המנה והלקוח בטבלאות DISHES, CUSTOMERS.
17. **customer_dislike_dish:** נשתמש ב-DELETE על מנת למחוק רשומה מטבלת CUSTOMERS_LIKE_DISHES אם קיימת אחת עם מזהה הלקוח ומזהה המנה המתקבלים כקלט.
18. **get_all_customer_likes:** נשתמש ב-SELECT על ה-JOIN בין טבלת CUSTOMERS_LIKE_DISHES לבין טבלת DISHES. JOIN זה נעשה על מנת לקבל את המידע המלא על מנה כפי שמופיעה ב-DISHES עבור מנות שהלקוח אהב בעזרת WHERE על התאמה לID של הלקוח. נשתמש בORDER BY כדי להחזיר את מזהי המנות בסדר עולה, לבסוף עם התוצאות שקיבלנו מהשאילתא נבנה רשימה של אובייקטי מנה, אם אין מנות כאלו או הזמנה כזו תוחזר רשימה ריקה.
19. **get_order_total_price:** נשתמש ב-SELECT על מנת לבחור את הסכום הכולל מ-CUSTOMERS_ORDERS_TOTAL_PRICE_VIEW המראה לנו את הסכום הכולל של הזמנה של לקוח כרגע ונבצע שימוש ב-WHERE על מנת למצוא את הרשומה המתאימה למזהה ההזמנה.
20. **get_max_amount_of_money_cust_spent:** נשתמש ב-VIEW CUSTOMERS_ORDERS_TOTAL_PRICE_VIEW וב-WHERE על מנת לקבל את המידע של הסכומים הכוללים של כל הזמנות הלקוח, וכן נבצע SELECT על MAX סכום כולל של הזמנה לקבלת הסכום המקסימלי שלקוח הוציא על הזמנה.
21. **get_most_expensive_anonymous_order**: בעזרת LEFT OUTER JOIN בין ORDERS ל-DISHES_IN_ORDERS נקבל total_price עבור כל מנה (בעזרת פונקציית SUM על מחיר מנה כפול כמות שנקנתה בהזמנה, יחד עם GROUP BY על מזהה ההזמנה (והתאריך כי הוא מופיע בSELECT) כאשר הערך הדיפולטיבי הוא 0 כשאין מנות בהזמנה), בעזרת WHERE נסתכל רק על רשומות של מנות אנונימיות (כאלו שהמזהה שלהם NOT IN בטבלה CUSTOMERS_PLACE_ORDERS) ובעזרת SELECT נקבל את המידע: מזהה מנה, תאריך ההזנה ומחירה הכולל, כאשר את מידע זה נקבל ממוין לפי המחיר בסדר יורד, כאשר בשוויון יש מיון משני לפי מזהה מנה בסדר עולה, לבסוף נבנה מנה מהערכים ברשומה העליונה ונחזיר אובייקט הזמנה (מניחים שיש לפחות הזמנה אחת אנונימית בDB)

22. **is_most_liked_dish_equal_to_most_purchased**: נבחר את מזהה המנה שנרכשה הכי הרבה מתוך טבלה שנקבל מביצוע INNER JOIN בין טבלה שמכילה את המנה שנרכשה הכי הרבה – מידע שנקבל ממיון MOST_PURCHASED_DISH_VIEW והגבלת גודל הטבלה ל-1 בעזרת LIMIT, וטבלה שמכילה את המנה הכי אהובה – מידע שנקבל באופן דומה מ-MOST_LIKED_DISH_VIEW כאשר הJOIN מתבצע לפי התאמה בין מזהי המנות, כך שרק במצב של שוויון תתקבל טבלה לא ריקה וה-SELECT יחזיר ערך לא ריק, במקרה זה נחזיר TRUE ואחרת FALSE.

23. **get_customers_ordered_top_5_dishes**: נבחר ID של לקוחות בעזרת SELECT ממוינים בסדר עולה בעזרת ORDER BY על מזהי הלקוחות, מתוך טבלה שמתקבלת מINNER JOIN בין DISHES_IN_ORDERS ו-CUSTOMERS_PLACE_ORDERS לפי התאמת מזהי המנות, כאשר בעזרת WHERE נבחר רק רשומות בהן מזהה המנה נמצא ברשימת 5 המנות האהובות ביותר, כלומר ב-MOST_LIKED_DISHES_RANKING_VIEW, בעזרת GROUP BY על מזהי הלקוחות נקבל לכל לקוח את המנות שהוא הזמין מתוך 5 האהובות ביותר ובעזרת HAVING נבחר רק לקוחות שיש להם בדיוק 5 מנות שונות, כלומר הזמינו את כל 5 המנות האהובות ביותר. אם לא הוחזר ערך מהשאילתה אין לקוחות שהזמינו את כל 5 המנות האהובות ביותר ותוחזר רשימה ריקה, אחרת תוחזר רשימת ID של הלקוחות המתאימים, בסדר עולה.

24. **get_non_worth_price_increase**: נבצע JOIN בין ORDERED_DISHES_PROFIT_VIEW לבין ACTIVE_ORDERED_DISHES_CURRENT_PROFIT_VIEW. האחרון מציין את הרווח הממוצע של מנה במחירה העדכני והראשון מציג את הרווח הממוצע של מנה לפי כל מחיר שהייתה בו. את הJOIN נתנה ב-WHERE על מנת שיראה רק את הרשומות בהן מחיר עבר זול יותר מהמחיר העדכני וגם שהרווח הממוצע של מחיר עבר גבוה יותר מזה של מחיר הווה. כמו כן נבצע עליהן ORDER BY כדי שמזהי המנות יופיעו בסדר עולה. נבחר את מזהי המנות שב-JOIN זה באמצעות SELECT. לבסוף נכניס את כל מזהי המנות שקיבלנו לרשימה ונחזירה. אם אין מנות כאלו או שהתקבלה שגיאה כלשהי מהשאילתא עקב מזהה מנה לא חוקי או חוסר במנה כזו ב-DB נחזיר רשימה ריקה.

25. **get_total_profit_per_month**: בעזרת SELECT נקבל כל חודש עם הרווח שלו (שיהיה 0 אם לא היו מנות שנרכשו בחודש זה), נבחר את מידע זה מתוך טבלה שתתקבל מLEFT OUTER JOIN בין MONTH_VIEW (כדי להבטיח שנקבל מידע על כל חודש) ו-PROFIT_PER_MONTH_VIEW שמכיל את הרווח לכל חודש בו היה רווח גדול מ-0, כאשר עבור חודש שלא קיים ב-PROFIT_PER_MONTH_VIEW ניתן ערך דיפולטיבי 0 באיחוד, לבסוף נחזיר את המידע ממוין בסדר יורד לפי החודש (נחזיר רשימה של tuples שכל tuple מכיל חודש ורווח חודשי).

26. **get_potential_dish_recommendations**: נבחר מזהי מנה ייחודים בעזרת SELECT מתוך הטבלה CUSTOMERS_LIKE_DISHES כאשר בעזרת WHERE נבחר רק מתוך רשומות בהן מזהה הלקוח מהווה "לקוח דומה" ללקוח בעל הID שקיבלנו בפונקציה והמנות אינן מנות שהלקוח עם המזהה שקיבלנו בפונקציה כבר אהב, לשם כך נשתמש ב-SIMILAR_CUSTOMERS_VIEW כאשר נקבל מVIEW זה רק רשומות בהן הID של הלקוח שניתן בפונקציה נמצא בעמודה customer ככה שבWHERE העליון נבחר מזהי לקוחות שINה הרשומות הנבחרות בview בהן הID הם בצד similar_customer עם הלקוח שניתן בפונקציה, כאשר בבניית הVIEW דאגנו שלקוח לא יכול להופיע עם עצמו בטבלה ולהיבחר כsimilar_customer. כדי לדאוג להציע רק מנות שהלקוח לא אהב, בבחירת הרשומות מ CUSTOMERS_LIKE_DISHES דאגנו לבחור רק רשומות WHERE ה-dish_id הוא NOT IN בטבלה CUSTOMERS_LIKE_DISHES ברשומות שנמצאות עם מזהה הלקוח שקיבלנו בפונקציה. לבסוף

בעזרת ORDER BY החזרנו את מזהי המנות המתאימות כהמלצות בסדר עולה. אם אין המלצות מתאימות תוחזר רשימה ריקה, אחרת תוחזר רשימת ID של מנות שלקוחות שדומים ללקוח אהבו אבל הוא לא אהב אותם בעצמו.