

# דוח מיני פרויקט בסיסי נתונים - מחלקה המטבח במאפייה

מגישות: מיכל חיימוב וחן אלקיים.

## הקדמה

מטרת פרויקט זה היא לייצג בסיס נתונים עבור **מחלקה המטבח והיצור במאפייה**, הפרויקט מכיל 10 טבלאות שונות, המאפשרות ניהול מרכיבי המטבח והסקת מסקנות עסקיות ותפעוליות באמצעות שאלות ודווחות.

## טבלאות

להלן פירוט של כל אחת מהטבלאות בסיס הנתונים של פרויקט מחלקה המטבח במאפייה, המבנה שלהן, ומטרתן:

### 1. טבלת עובדים (Employee)

טבלה זו מייצגת את כלל העובדי המאפייה, המנהל ועד האופים והאורזים. מטרתה היא לאפשר שיבוץ עובדים למשמרות ועמדות עבודה, ומעקב אחר ביצועי העבודה שלהם לפי תפקיד.

שם השדה בטבלה	סוג השדה	הסבר קצר על השדה
employee_id	SERIAL PRIMARY KEY	מזהה ייחודי של העובד. מפתח ראשי (PK).
first_name	(VARCHAR(60)	שם פרטי של העובד.
last_name	(VARCHAR(60)	שם משפחה של העובד.
role	(VARCHAR(30)	תפקיד העובד במאפייה (למשל, Baker, Manager).

להלן קוד יצירה הטעלה בSQL:

```
-- =====
-- 1) Employee
-- =====
CREATE TABLE IF NOT EXISTS employee (
    employee_id    SERIAL PRIMARY KEY,
    first_name     VARCHAR(60),
    last_name      VARCHAR(60),
    role           VARCHAR(30),
    CONSTRAINT employee_role_check
        CHECK (role IN ('Baker', 'Prep', 'Manager', 'Packaging'))
);
```

## 2. טבלת חומרי גלם (Ingredient)

טבלה זו מכילה את כל המידע על חומרי הגלם המשמשים במתכונים (כגון קמח, סוכר, שוקולד). מטרתה היא לתעד את שמות החומרים, ייחידות המידה שלהם, עלות היחידה, ומיעד על אלרגנים.

הסבר קצר על השדה	סוג השדה	שם השדה בטבלה
מזהה ייחודי של חומר הגלם. מפתח ראשי (PK).	SERIAL PRIMARY KEY	ingredient_id
שם חומר הגלם (למשל, 'קמח לבן', 'שמרים').	VARCHAR(120) NOT NULL	name
.(kg, g, L, pcs) ייחידת המידה לכימות	(VARCHAR(20	unit
סימון האם חומר הגלם מכיל אלרגן (YES/NO)	(VARCHAR(3	allergen_flag
עלות חומר הגלם ליחידת מידת.	(NUMERIC(10,2	cost_per_unit

להלן קוד יצירת הטבלה בSQL:

```
-- =====
-- 2) Ingredient
-- =====
CREATE TABLE IF NOT EXISTS ingredient (
    ingredient_id      SERIAL PRIMARY KEY,
    name               VARCHAR(120) NOT NULL,
    unit               VARCHAR(20),
    allergen_flag     VARCHAR(3),
    cost_per_unit     NUMERIC(10,2),
    CONSTRAINT ingredient_unit_check
        CHECK (unit IN ('kg','g','L','mL','pcs')),
    CONSTRAINT ingredient_allergen_flag_check
        CHECK (allergen_flag IN ('YES','NO')),
    CONSTRAINT ingredient_cost_per_unit_check
        CHECK (cost_per_unit > 0)
);
```

### 3. טבלת מוצרים (Product)

טבלה זו מייצגת את כל המוצרים המוגמרים שהמאפייה מוכרת ללקחות. מטרתה היא לנחל את קטלוג המוצרים, כולל מחיר המכירה שלהם וקטgoriyת השיווק (כגון לחמים, עוגות).

הסבר קצר על השדה	סוג השדה	שם השדה בטבלה
מזהה ייחודי של המוצר המוגמר. מפתח ראשי (PK).	SERIAL PRIMARY KEY	product_id
שם המוצר (למשל, 'עוגת גבינה', 'bagel كلאו').	VARCHAR(120) NOT NULL	name
הקטגוריה שאליה משתייך המוצר (למשל, (Breads, Cakes).	(VARCHAR(30	category
מחיר המכירה של יחידה אחת מה מוצר.	(NUMERIC(10,2	price

להלן קוד יצירת הטבלה בSQL:

```
-- =====
-- 3) Product
-- =====
CREATE TABLE IF NOT EXISTS product (
    product_id    SERIAL PRIMARY KEY,
    name          VARCHAR(120) NOT NULL,
    category      VARCHAR(30),
    price         NUMERIC(10,2),
    CONSTRAINT product_category_check
    | CHECK (category IN ('Breads','Cakes','Savory')),
    CONSTRAINT product_price_check
    | CHECK (price > 0)
);
```

#### 4. טבלת עמדות עבודה (Station)

טבלה זו מתעדת את כל עמדות העבודה והצדד הזמינים במטבח המאפייה. מטרתה היא לנוהל את משאבי הייצור ולאפשר שיבוץ הפקות וצורות לעמדות ספציפיות.

שם השדה בטבלה	סוג השדה	הסבר קצר על השדה
station_id	SERIAL PRIMARY KEY	מזהה ייחודי של עמדת העבודה/צדיך. מפתח ראשי (PK).
name	VARCHAR(60) NOT NULL	שם העמדה (למשל, 'תנור ראשי', 'מיكسر').
type	(VARCHAR(30	סוג העמדה (Oven, Mixer, Prep Station)

להלן קוד יצירת הטבלה בSQL:

```
-- =====
-- 4) Station
-- =====
CREATE TABLE IF NOT EXISTS station (
    station_id    SERIAL PRIMARY KEY,
    name          VARCHAR(120),
    type          VARCHAR(30),
    CONSTRAINT station_type_check
        CHECK (type IN ('Oven','Prep','Mixer','Packaging'))
);
```

## 5. טבלת משמרות (Shift)

טבלה זו מתעדת את כל משמרות העבודה המתווכנות והמתקימות במאפייה.  
מטרתה היא לתחום את זמני העבודה של העובדים, כולל תאריך, שעות התחלת/סיום וסוג המשמרת.

הסבר קצר על השדה	סוג השדה	שם השדה בטבלה
מזהה ייחודי של המשמרת. מפתח ראשי .(PK)	SERIAL PRIMARY KEY	shift_id
התאריך שבו מתקימת המשמרת.	DATE NOT NULL	shift_date
שעת התחלת המשמרת (ערך שלם, מ-0 עד .(23	INTEGER	start_hour
שעת סיום המשמרת (ערך שלם, מ-0 עד .(23	INTEGER	end_hour
.(Morning, Evening, Night) סוג המשמרת	(VARCHAR(30	shift_type

להלן קוד יצירת הטבלה בSQL:

```
-- =====
-- 5) shift
-- =====
CREATE TABLE IF NOT EXISTS shift [
    shift_id      SERIAL PRIMARY KEY,
    shift_date    DATE,
    start_hour   NUMERIC(2,0),
    end_hour     NUMERIC(2,0),
    shift_type   VARCHAR(10),
    CONSTRAINT shift_start_hour_check
        CHECK (start_hour >= 0 AND start_hour <= 23),
    CONSTRAINT shift_end_hour_check
        CHECK (end_hour  >= 0 AND end_hour   <= 23),
    CONSTRAINT shift_shift_type_check
        CHECK (shift_type IN ('Morning','Evening','Night')),
    CONSTRAINT shift_shift_date_check
        CHECK (shift_date <= CURRENT_DATE)
];
```

## 6. טבלת מתכונים (Recipe)

טבלה זו מכילה את המתכונים ליצירת המוצרים המוגמרים.  
מטרתה היא לתעד את גרסאות המתכונים, מתי נוצרו, וכמה יחידות תוצר סופי כל מתכון אמור ליצור.

שם השדה בטבלה	סוג השדה	הסבר קצר על השדה
recipe_id	SERIAL PRIMARY KEY	מזהה ייחודי של המתכון. מפתח ראשי (PK).
product_id	INTEGER	מפתח זר (FK) לטבלה Product. מקשר את המתכון למוצר.
version_no	(NUMERIC(6,2)	מספר גרסה המתכון (לצורך מעקב שינויים).
created_date	DATE	התאריך בו נוצר/עדכן המתכון.
yield_units	INTEGER	כמות יחידות המוצר הסופית שמאפייק המתכון.

להלן קוד יצירת הטבלה בSQL:

```
-- =====
-- 6) Recipe
-- =====
CREATE TABLE IF NOT EXISTS recipe (
    recipe_id      SERIAL PRIMARY KEY,
    product_id     INTEGER,
    version_no     NUMERIC(6,2),
    created_date   DATE,
    notes          VARCHAR(200),
    yield_units    INTEGER,
    CONSTRAINT uq_recipe_product_version
        UNIQUE (product_id, version_no),
    CONSTRAINT recipe_product_id_fkey
        FOREIGN KEY (product_id)
        REFERENCES product (product_id)
        ON UPDATE NO ACTION
        ON DELETE NO ACTION,
    CONSTRAINT recipe_version_no_check
        CHECK (version_no > 0),
    CONSTRAINT recipe_created_date_check
        CHECK (created_date <= CURRENT_DATE),
    CONSTRAINT recipe_yield_units_check
        CHECK (yield_units > 0)
);
```

## 7. טבלת אצאות מלאי (Batch)

טבלה זו עוקבת אחר מלאי חומר הגלם הפיזיים לפי ארגזים קבלה ספציפיים. מטרתה היא לאפשר ניהול מדויק של תאריכי התפוגה, כמות נכנסת, והכמות הנוכחית שנותרת במלאי מכל חומר גלם.

שם השדה בטבלה	סוג השדה	הסבר קצר על השדה
batch_id	SERIAL PRIMARY KEY	מזהה ייחודי של אצואה חומר הגלם. מפתח ראשי (PK).
ingredient_id	INTEGER	מפתח זר (FK) לטבלה Ingredient.
received_date	DATE	תאריך קבלת האצואה למלאי.
expiry_date	DATE	תאריך התפוגה של חומר הגלם באצואה זו.
quantity_current	(NUMERIC(12,3	הכמות שנותרת במלאי מהאצואה.
location	(VARCHAR(60	מקום אחסון האצואה.

להלן קוד ייצירת הטבלה בSQL:

```
-- =====
-- 7) Batch
-- =====
CREATE TABLE IF NOT EXISTS batch (
    batch_id      SERIAL PRIMARY KEY,
    ingredient_id INTEGER,
    received_date DATE,
    expiry_date   DATE,
    quantity_current NUMERIC(12,3),
    location      VARCHAR(60),
    CONSTRAINT batch_ingredient_id_fkey
        FOREIGN KEY (ingredient_id)
            REFERENCES ingredient (ingredient_id)
            ON UPDATE NO ACTION
            ON DELETE NO ACTION,
    CONSTRAINT batch_check
        CHECK (expiry_date >= received_date),
    CONSTRAINT batch_quantity_current_check
        CHECK (quantity_current >= 0),
    CONSTRAINT batch_received_date_check
        CHECK (received_date <= CURRENT_DATE)
);
```

## 8. טבלת פריטי מתכון (RecipeItem)

טבלה זו היא טבלת קישור (Many-to-Many) בין Recipe לבין Ingredient. מטרתה היא לפרט אילו חומרי גלם נדרשים לכל מתכון, ובאיזה כמות מדויקת.

שם השדה בטבלה	סוג השדה	הסבר קצר על השדה
recipe_id	INTEGER	מפתח זר (FK) לטבלת Recipe. חלק מהמפתח הראשי.
ingredient_id	INTEGER	מפתח זר (FK) לטבלת Ingredient. חלק מהמפתח הראשי.
quantity	(NUMERIC(12,3)	הכמות הנדרשת מחומר הגלם עבור המתכון.
unit	(VARCHAR(20)	יחידת המידה של הכמות הנדרשת (kg, g, L, pcs).

להלן קוד יצירת הטבלה בSQL:

```
-- =====
-- 8) RecipeItem
-- =====
CREATE TABLE IF NOT EXISTS recipeitem (
    recipe_id      INTEGER NOT NULL,
    ingredient_id  INTEGER NOT NULL,
    quantity        NUMERIC(12,3),
    unit            VARCHAR(20),
    PRIMARY KEY (recipe_id, ingredient_id),
    CONSTRAINT recipeitem_recipe_id_fkey
        FOREIGN KEY (recipe_id)
        REFERENCES recipe (recipe_id)
        ON UPDATE NO ACTION
        ON DELETE NO ACTION,
    CONSTRAINT recipeitem_ingredient_id_fkey
        FOREIGN KEY (ingredient_id)
        REFERENCES ingredient (ingredient_id)
        ON UPDATE NO ACTION
        ON DELETE NO ACTION,
    CONSTRAINT recipeitem_quantity_check
        CHECK (quantity > 0),
    CONSTRAINT recipeitem_unit_check
        CHECK (unit IN ('kg','g','L','mL','pcs'))
);
```

## 9. טבלת הפקות (Production)

טבלה זו מתעדת כל אירוע ייצור שהתרחש בפועל במטבח. מטרתה היא לתעד איזה מוצר נוצר, באיזה מטבח, היכן בוצע, על ידי מי הוביל, ובכמה יחידות תוצר סופי.

שם השדה בטבלה	סוג השדה	הסבר קצר על השדה
production_id	SERIAL PRIMARY KEY	מזהה ייחודי של אירוע ההפקה. מפתח ראשי (PK).
product_id	INTEGER	מפתח זר (FK) לטבלה Product.
recipe_id	INTEGER	מפתח זר (FK) לטבלה Recipe.
station_id	INTEGER	מפתח זר (FK) לטבלה Station.
leader_employee_id	INTEGER	מפתח זר (FK) לטבלה Employee. העובד המוביל את ההפקה.
bake_date	DATE	התאריך שבו בוצעה ההפקה.
quantity_output	(NUMERIC(12,3)	כמות יחידות המוצר שהופקו בפועל.
shift_id	INTEGER	מפתח זר (FK) לטבלה Shift.

להלן קוד ייצור הטבלה בSQL:

```
-- 9) Production
-- =====
CREATE TABLE IF NOT EXISTS production (
    production_id      SERIAL PRIMARY KEY,
    product_id         INTEGER,
    recipe_id          INTEGER,
    station_id         INTEGER,
    leader_employee_id INTEGER,
    bake_date          DATE,
    quantity_output    NUMERIC(12,3),
    shift_id           INTEGER,
    CONSTRAINT production_product_id_fkey
        FOREIGN KEY (product_id)
        REFERENCES product (product_id)
        ON UPDATE NO ACTION
        ON DELETE NO ACTION,
    CONSTRAINT production_recipe_id_fkey
        FOREIGN KEY (recipe_id)
        REFERENCES recipe (recipe_id)
        ON UPDATE NO ACTION
        ON DELETE NO ACTION,
    CONSTRAINT production_station_id_fkey
        FOREIGN KEY (station_id)
        REFERENCES station (station_id)
        ON UPDATE NO ACTION
        ON DELETE NO ACTION,
    CONSTRAINT production_leader_employee_id_fkey
        FOREIGN KEY (leader_employee_id)
        REFERENCES employee (employee_id)
        ON UPDATE NO ACTION
        ON DELETE NO ACTION,
    CONSTRAINT fk_production_shift
        FOREIGN KEY (shift_id)
        REFERENCES shift (shift_id)
        ON UPDATE NO ACTION
        ON DELETE NO ACTION,
    CONSTRAINT production_bake_date_check
        CHECK (bake_date <= CURRENT_DATE),
    CONSTRAINT production_quantity_output_check
        CHECK (quantity_output >= 0)
);
```

## 10. טבלה שיבוץ (Assignment)

טבלה זו משבצת עובד ספציפי למשמרת ולעמדת עבודה מסוימת. מטרתה היא לתעד את התכונן התפעולי: מי עבד, מתי, היכן, ומה הייתה המשימה שהוטלה עליו.

שם השדה בטבלה	סוג השדה	הסבר קצר על השדה
assignment_id	SERIAL PRIMARY KEY	מזהה ייחודי של אירוע השיבוץ. מפתח ראשי (PK).
employee_id	INTEGER	מפתח זר (FK) לatable Employee.
shift_id	INTEGER	מפתח זר (FK) לatable Shift.
station_id	INTEGER	מפתח זר (FK) לatable Station.
task_name	(VARCHAR(30)	המשימה הספציפית שבוצעה (למשל, 'Baking', 'Packaging').

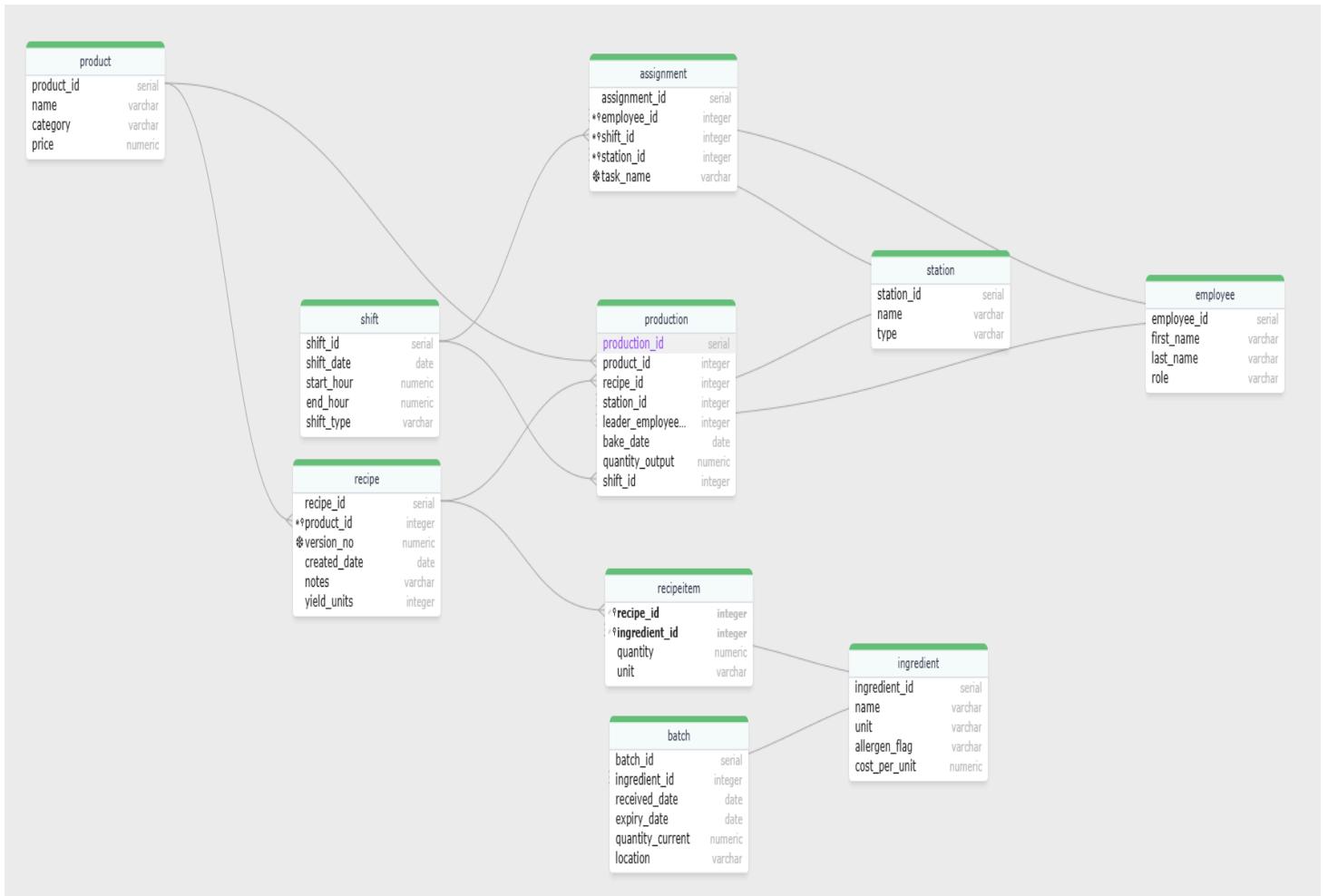
להלן קוד יצירת הטבלה בSQL:

```
-- =====
-- 10) Assignment
-- =====
CREATE TABLE IF NOT EXISTS assignment (
    assignment_id SERIAL PRIMARY KEY,
    employee_id   INTEGER,
    shift_id      INTEGER,
    station_id   INTEGER,
    task_name     VARCHAR(30),
    CONSTRAINT uq_assignment_unique
    UNIQUE (employee_id, shift_id, station_id, task_name),
    CONSTRAINT assignment_employee_id_fkey
    FOREIGN KEY (employee_id)
    REFERENCES employee (employee_id)
    ON UPDATE NO ACTION
    ON DELETE NO ACTION,
    CONSTRAINT assignment_shift_id_fkey
    FOREIGN KEY (shift_id)
    REFERENCES shift (shift_id)
    ON UPDATE NO ACTION
    ON DELETE NO ACTION,
    CONSTRAINT assignment_station_id_fkey
    FOREIGN KEY (station_id)
    REFERENCES station (station_id)
    ON UPDATE NO ACTION
    ON DELETE NO ACTION,
    CONSTRAINT assignment_task_name_check
    CHECK (task_name IN ('Prep','Baking','Cleaning','Packaging','Mixing'))
);
```

## 1. תרשימים ERD

התרשים מייצג את הישויות המרכזיות (הטבלאות) ואת הקשרים ביניהן. הקשרים בטבלה ERD שלנו:

- **קשר אחד-לרבים (1:N)** קיים בין **Product** ( מוצר) ל**Recipe** ( מטבחון), כאשר למוצר אחד יכולם להיות מספר גרסאות מתכוון.
- **קשר רבים-רבים (M:N)** קיים בין **Ingredient** (redient) ל**Recipe** ( מטבחון) באמצעות טבלת הקישור **RecipItem**.
- **קשר אחד-רבים (N:1)** קיים בין **Production** (עמדת הפקה) ל**Station** (הפקה), שכן עמדה אחת יכולה לשמש להפקות רבות.



## שאלות ודוחות



השאלות שפותחו נועדו לתת מענה לצורכי הניהול, הבקרה והתפעול של מחלקת המטבח במאפייה, ולאפשר הסקת מסקנות עסקיות מורכבות.

#### א. שאלות מורכבות

1. שאלתה לזייה עובדים המעורבים בייצור מעלה הממוצע

פיארָן

שאילתה זו נועדה לזהות אילו עובדים היו שותפים להפקות שבון כמוות הייצור הייתה גבוהה מהתמוצע הכללי במאפייה.

השאילתה מחשבת תחילת את ממוצע התוצריים בכל פעולות הייצור, ולאחר מכן אמרת רק אם הפקות שרכו כמות הייצור גורובה ממנה.

בסוף מתבצע חיבור (JOIN) בין פעולות אלו לבין השיבוצים של העובדים, וכך ניתן לקבל רשימה של העובדים שבהן מושרכות ביחסות מושגניות:

מושלים למחנה

השאילתת אפשרות להזיהות עובדים מצטיינים, להבין מי תורם יותר לתפקידו, ולתכנן משמרות  
רונומטיות בהתאם

## 2. דוח שימוש מלאי (Inventory Usage Report)

תיאור:

שאילתת זו מחשבת את הכמות האמיתית של חומרי הגלם שנצרכו בכל פעולות הייצור, ומשווה זאת מול כמות המלאי הנוכחי מכל הארגזים. החישוב מבוצע על בסיס הכמות הנדרשת לכל מרכיב בתוכן כפול מספר היחידות שיוצרו בייצור בפועל. בנוסף מוצג גם כמה נשאר מהמלאי עבור כל חומר גלם.

תועלת למנהל:

השאילתת מס'ית בתכנון הזמן, מניעת חוסרים, ניתוח בΖבוזים ויזיהו רכיבים הנמצאים בשימוש אינטנסיבי.

```
-- 2) Inventory usage report
SELECT
    IU.ingredient_id,
    IU.ingredient_name,
    IU.used_amount,
    ISK.total_quantity,
    (ISK.total_quantity - IU.used_amount) AS remaining_quantity
FROM (
    -- שימוש אמיתי בחומר גלם לפי כל החקפות--
    SELECT
        ri.ingredient_id,
        i.name AS ingredient_name,
        SUM(ri.quantity * p.quantity_output) AS used_amount
    FROM Production p
    JOIN RecipeItem ri
    ON p.recipe_id = ri.recipe_id
    JOIN Ingredient i
    ON ri.ingredient_id = i.ingredient_id
    GROUP BY ri.ingredient_id, i.name
) AS IU
LEFT JOIN (
    -- כמה יש מלאי מתוך באז'
    SELECT
        ingredient_id,
        SUM(quantity_current) AS total_quantity
    FROM Batch
    GROUP BY ingredient_id
) AS ISK
ON IU.ingredient_id = ISK.ingredient_id
ORDER BY IU.used_amount DESC;
```

### 3. דוח הכנסות לפי עמדת עבודה (Station Revenue Report)

תיאור:

שאילתת זו מציגה את סך הכנסות שכל עמדה במטבח יוצרה.  
הчисלוב מבוצע לפי: כמות המוצרים שייצרו × מחיר המוצר.  
השאילתת משתמשת ב-NJOIN LEFT כדי להציג גם עמדות שבהן לא בוצע כלל ייצור.

תועלת למנהל:

מאפשר להבין אילו עמדות מניבות יותר, לזרות צוואר בקבוק, ולתכנן חלוקה חדשה של כוח אדם וצדוק.

```
-- 3) station revenue report
SELECT
    s.station_id, s.name AS station_name,
    COALESCE(SUM(pr.quantity_output * p.price), 0) AS total_revenue
FROM Station s
LEFT JOIN Production pr
    ON pr.station_id = s.station_id
LEFT JOIN Product p
    ON pr.product_id = p.product_id
GROUP BY s.station_id, s.name
ORDER BY total_revenue DESC;
```

### 4. דוח רווחות למוצר (Profit per Product Report)

תיאור:

שאילתת זו מחשבת את הרווח לייחידת מוצר על בסיס: עלות חומרי הגלם הדרושים לייצור יחידה אחת (ע"י חלוקה של עלות המרכיבים ב-yield של המתכוון).  
לאחר מכן מחושב הרווח לייחידה ואחוז הרווח ביחס למחיר המוצר.

תועלת למנהל:

מאפשרת לראות אילו מוצרים רווחיים יותר, לזרות מוצרים בעיתויים, ובוצע אופטימיזציה למחיר או למתקון.

```
--4)profit report per product
SELECT
    p.product_id,
    p.name AS product_name,
    p.price AS sell_price,
    SUM(ri.quantity * i.cost_per_unit) / r.yield_units AS cost_per_unit,
    (p.price - SUM(ri.quantity * i.cost_per_unit) / r.yield_units) AS profit_per_unit,
    ROUND(((p.price - SUM(ri.quantity * i.cost_per_unit) / r.yield_units) / p.price) * 100,2)
        AS profit_margin_percent

FROM Product p
JOIN Recipe r
    ON r.product_id = p.product_id
JOIN RecipeItem ri
    ON ri.recipe_id = r.recipe_id
JOIN Ingredient i
    ON i.ingredient_id = ri.ingredient_id

GROUP BY p.product_id, p.name, p.price
ORDER BY profit_per_unit DESC;
```

## 5. מוצרים שנייתן לייצר מחומר גלם שפג תוקף בקרוב (Batch Expiration Forecast)

תיאור:

שאילתה זו בודקת אילו חומרי גלם אמורים לפג תוקף ב-30 הימים הקרובים, ומצאה אילו מוצרים ניתן להכין מהם כדי למנוע מצב זה.

היא משתמשת ב-WIPE שמצויג את הגרסה الأخيرة של כל מתכוון, ומשם מחשבת כמה ייחידות מוצר ניתן לייצר מכל כמות קיימת.

תועלת למנהל:

סיווע בקבלת החלטות על סדרי ייצור, מניעת הפסדים, ושימוש חכם במלאי לפני פגיעה.

```
--5) Products that can be made from ingredients nearing expiration in the next 30 days

CREATE VIEW lastVersionRecipeProduct AS
SELECT DISTINCT ON (product_id)
    product_id, recipe_id, version_no,yield_units
FROM Recipe
ORDER BY product_id,version_no DESC;

SELECT p.product_id as product_id,
       p.name as product_name,
       ing.name as ingredient_name,
       MIN(b.expiry_date) AS earliest_expiry,
       MIN(b.expiry_date) - CURRENT_DATE AS soonest_expiration_days,
       FLOOR(SUM(ROUND(b.quantity_current / ri.quantity)*r.yield_units)) AS estimated_product_units_to_save
FROM Batch b
JOIN Ingredient ing
ON b.ingredient_id=ing.ingredient_id
JOIN RecipeItem ri
ON ri.ingredient_id=ing.ingredient_id
JOIN lastVersionRecipeProduct r
ON r.recipe_id=ri.recipe_id
JOIN Product p
ON r.product_id=p.product_id
WHERE b.expiry_date<=CURRENT_DATE+30 and b.expiry_date>CURRENT_DATE and b.quantity_current>0
GROUP BY p.product_id,p.name,ing.name
ORDER BY
    soonest_expiration_days ASC,
    estimated_product_units_to_save DESC;
```

## ב. שאלות ברמה בינונית (Intermediate Level Queries)

### 1. עשרת העמדות העמוסות ביותר (Top 10 Busy Stations)

**תיאור:**

שאילתת זו מציגה אילו עמדות ביצעו את כמות הייצור הגבוהה ביותר. החישוב מתבצע באמצעות סכימת כמות התוצרים שיוצרו בכל עמדה.

**תועלת למנהל:**

ניתן לזהות עמדות בעלות מיוחדת, להבין היכן יש צורך בהרחבת ציוד או כוח אדם, ולשפר היעילות של תהליך הייצור.

```
-- 1) Top 10 busy stations
SELECT
    s.station_id,
    s.name,
    SUM(p.quantity_output) AS total_amount_produced
FROM Station s
JOIN Production p
    ON s.station_id = p.station_id
GROUP BY s.station_id, s.name
ORDER BY total_amount_produced DESC
Limit 10;
```

### 2. תפוקה ממוצעת לשעה לפי עמדה (Average Hourly Output per Station)

**תיאור:**

שאילתת זו מחשבת את התפוקה הממוצעת לשעה של כל עמדה, על בסיס: כמות הייצור

הכוללית / משך המשמרת בשעות.

החיבור בין הנתונים נעשה באמצעות JOIN על טבלת המשמרות.

**תועלת למנהל:**

מאפשרת לראות אילו עמדות עובדות מהר יותר, ואילו דורותות תחקור או שיפור בתהליכי העבודה.

```
--2)The average hourly output for each station id in each shift id
SELECT
    st.station_id,
    SUM(p.quantity_output)/(ABS(s.start_hour-s.end_hour)) AS avg_output_per_hour
FROM production p
JOIN Shift s
ON p.shift_id = s.shift_id
JOIN Station st
ON p.station_id = st.station_id
GROUP BY st.station_id
```

### 3. פיצול ייצור לפי תפקיד העובד המוביל (Production Output by Leader Role) (Tutor)

תיאור:

שאילתה זו מציגה את סך התוצרים שיוצרים בכל עמדת, מפוצלים לפי תפקיד העובד שהיה אחראי על התהליך (Baker, Prep, וכו').  
באמצעות JOIN לטבלת העובדים ניתן לראות האם תפקידים מסוימים מייצרים יותר.

תועלת למנהל:

מאפשר להבין את תרומת כל סוג תפקיד, לראות מי משפייע על התפקוכה, ולדעת היכן נדרש תגבור או הדרכה נוספת.

```
-- 3) The total amount of output produced at each station,  
-- broken down by the role of the worker who led the production batch.  
SELECT  
    s.name AS station_name,  
    e.role AS leader_role,  
    SUM(p.quantity_output) AS total_output_units  
FROM  
    production p  
JOIN  
    station s ON p.station_id = s.station_id  
JOIN  
    employee e ON p.leader_employee_id = e.employee_id  
GROUP BY  
    s.name,  
    e.role  
ORDER BY  
    s.name,  
    total_output_units DESC;
```

## **שלב 3- אינטגרציה**

בשלב האינטגרציה התקבל בסיס נתונים נוסף מזוג אחר, אשר עוסק בניהול משאבי אנוש בארגון וכלל טבלאות כגון עובדים, מחלקות, תפקידים, משכורות, נוכחות וחופשות. מטרת שלב זה הייתה לשלב בין שני בסיסי הנתונים – בסיס הנתונים של מערכת הייצור והמאפייה ובבסיס הנתונים של מערכת משאבי האנוש – כך שייתקבל בסיס נתונים אחד מאוחד, עיקבי ושלם, המאפשר הרצת כל השאלות משנה הפרויקטים ללא שגיאות.

### **זיהוי טבלאות חופפות וكونפליקטים**

במהלך ניתוח מבני הסכמות של שני בסיסי הנתונים, זוהתה טבלה בעלת משמעות דומה בשני הפרויקטים – טבלת העובדים. בבסיס הנתונים של הפרויקט שלנו הופיעה הטבלה employees, אשר שימשה לניהול עובדים לצורכי תהליכי ייצור (שיוך למשמרות, תחנות ותפקידים), בעוד שבבסיס הנתונים של הזוג השני הופיעה הטבלה hr\_employees, אשר כללה מידע נרחב יותר על העובדים בהיבטי משאבי אנוש (פרטים אישיים, מחלקות, תפקידים, סטטוס תעסוקתי וכו'). הוחלט לאחד את הטבלאות לטבלת ליבה אחת בשם employees, המשמשת כמקור אמת לכל המידע על עובדים במערכת.

לצורך שמירה על עקביות הנתונים, הוגדרו טבלאות מייפוי אשר אפשרו קישור בין מידעidos יפנים למזהה עובדים חדשים. לאחר מכן עודכנו בהדרגה המפתחות הזרים בכל הטבלאות התלוויות בעובדים, כך שייפנו לטבלה המאוחדת employees. בנוסף, כל השאלות הקשורות לעובדים נוערכו כך שייעברו ישירות מול הטבלה המאוחדת, ובכך הושלם המעבר המלא לסכימת הנתונים החדשה.

תיאור השאלה (SQL Query)	תיאור השאלה
<pre> WITH above_prod AS (   SELECT shift_id, station_id   FROM public.production   WHERE quantity_output &gt; (SELECT AVG(quantity_output) FROM public.production) )  SELECT DISTINCT e.first_name, e.last_name, e.role FROM above_prod p JOIN public.assignment a ON a.shift_id = p.shift_id AND a.station_id = p.station_id JOIN public.employees e ON e.employee_id = a.employee_id; </pre>	<p>שאילתת זו מתרת הפקות שבן כמות הייצור גבוהה מהממוצע הכללי, ומחזירה את רישימת העובדים ששובצו לשמשרות ולתchnות של אותן הפקות. השאלה מאפשרת זיהוי עובדים המעורבים בBITS גבויים.</p>
<pre> SELECT d.department_name, EXTRACT(YEAR FROM s.pay_date) AS pay_year, EXTRACT(MONTH FROM s.pay_date) AS pay_month, COUNT(DISTINCT e.employee_id) AS num_employees_paid, SUM(s.total_salary) AS total_payroll, AVG(s.total_salary) AS avg_salary FROM salaries s JOIN employees e ON s.employee_id = e.employee_id JOIN departments d ON e.department_id = d.department_id GROUP BY d.department_name, pay_year, pay_month ORDER BY pay_year DESC, pay_month DESC, total_payroll DESC; </pre>	<p>שאילתת זו מציגה דוח שכר חודשי לכל מחלקה בארגון, כולל את מספר העובדים שקיבלו שכר, סך השכר ששולם והשכר המוצע, בחלוקת לפי שנה וחודש.</p>

שאלת מbasis הנתונים השני.	תאור השאלה
<pre data-bbox="219 368 966 714"> SELECT e.employee_id, CONCAT(e.first_name, ' ', e.last_name) AS full_name, EXTRACT(YEAR FROM a.work_date) AS work_year, EXTRACT(MONTH FROM a.work_date) AS work_month, SUM(a.hours_worked) AS total_hours FROM employees e JOIN attendance a ON e.employee_id = a.employee_id GROUP BY e.employee_id, full_name, work_year, work_month HAVING SUM(a.hours_worked) &gt; 8 ORDER BY total_hours DESC; </pre>	<p>שאלת זו מחשבת את סך שעות העבודה של כל עובד לפי חודש ונה, ומהזירה עובדים שביצעו מעל 8 שעות עבודה מוצברות באותו חודש. מטרתה ליזהות עומס עבודה ושעות נוספת.</p>
<pre data-bbox="219 846 982 1343"> WITH max_dates AS (   SELECT MAX(end_date) AS max_end   FROM public.leaves   WHERE status = 'Approved' ) SELECT e.employee_id, e.first_name, e.last_name, l.leave_type, l.start_date, l.end_date, l.status FROM public.leaves l JOIN public.employees e ON l.employee_id = e.employee_id CROSS JOIN max_dates WHERE l.status = 'Approved' AND l.end_date &gt;= max_dates.max_end - INTERVAL '30 days' ORDER BY l.end_date DESC; </pre>	<p>שאלת זו מציגה חופשות מאושרו מהטוח הרלוונטי ביותר בנתונים (30 ים מהתאריך האחרון), כדי לזרות עובדים הנמצאים או שהו לאחרונה בחופשה</p>
<pre data-bbox="219 1430 1030 1927"> SELECT p.product_id, p.name AS product_name, p.price AS sell_price, ROUND(SUM(ri.quantity * i.cost_per_unit) / r.yield_units, 2) AS cost_per_unit, ROUND(p.price - SUM(ri.quantity * i.cost_per_unit) / r.yield_units, 2) AS profit_per_unit, ROUND(((p.price - SUM(ri.quantity * i.cost_per_unit) / r.yield_units) / p.price) * 100, 2) AS profit_margin_percent FROM public.product p JOIN public.recipe r ON r.product_id = p.product_id JOIN public.recipeitem ri ON ri.recipe_id = r.recipe_id JOIN public.ingredient i ON i.ingredient_id = ri.ingredient_id GROUP BY p.product_id, p.name, p.price, r.yield_units ORDER BY profit_per_unit DESC; </pre>	<p>שאלת זו מציגה דוח רווחות לפי מוצר, הכלול עלות ייצור יחידה, רווח יחידה ואחוז רווחיות על בסיס עלות רכיבי המתכוון.</p>

```
SELECT e.employee_id,
CONCAT(e.first_name, ' ', e.last_name) AS full_name,
SUM(s.bonus) AS total_bonus
FROM employees e
JOIN salaries s ON e.employee_id = s.employee_id
WHERE EXTRACT(YEAR FROM s.pay_date) =  EXTRACT(YEAR
FROM CURRENT_DATE)
GROUP BY e.employee_id, full_name
ORDER BY total_bonus DESC
LIMIT 3;
```

שאילתה זו ממחישה את  
שלושת העובדים בעלי  
הbonos המציגים הגבוהה  
ב尤ר בשנה הנוכחית.