**SQL-Portfolio**

**SQL Oracle – PLSQL**

Querry Number 1 – Check Correct Emails:

CREATE TABLE Check\_Correct\_email AS

WITH "CTE" AS

(

SELECT DISTINCT

       email\_table.id,

       email\_table.email,

*-- בודק אם המייל מכיל את אחד הסימנים האלה*

      CASE  WHEN REGEXP\_LIKE(email\_table.email,'[][}{#%^\*+=!?+<>$"&\_]')

*-- בודק אם המייל מכיל מספרים בלבד*

            OR REGEXP\_LIKE( email\_table.email, '^\d{0,6}\.{0,1}\d{0,6}$')

*-- בודק אם המייל מכיל תווים בעברית*

            OR REGEXP\_LIKE(email\_table.email,'[א-ת]')

*-- בודק אם המייל מסתיים או מתחיל באחד מהתווים הבאים*

            OR email\_table.email LIKE '@%'

            OR email\_table.email LIKE '%@@%'

            OR email\_table.email LIKE '%con'

            OR email\_table.email LIKE '%i'

            OR email\_table.email LIKE '%[ci.il](http://ci.il" \t "_blank)'

            OR email\_table.email LIKE '%.'

            OR email\_table.email LIKE '%-'

            OR email\_table.email LIKE '%;%'

            OR email\_table.email LIKE '%[fnx.co.il](http://fnx.co.il" \t "_blank)%'

            OR email\_table.email NOT LIKE '%@%'

            OR email\_table.email NOT LIKE '%.%'

            OR email\_table.email LIKE '%ill'

*--בודק אם האות האחרונה במייל מסתיימת במספר כלשהו*

            OR SUBSTR(email\_table.email, LENGTH(email\_table.email)) IN ('0','1','2','3','4','5','6','7','8','9')

            then 'Wrong Email'

            else 'Correct Email' END AS CheckEmails

FROM EmailTable email\_table

WHERE 1=1

      AND email\_table.email IS NOT NULL

)

SELECT  DISTINCT

 cte.id,

 COUNT(cte.email) OVER (PARTITION BY cte.id AS CountEmails,

 LISTAGG(cte.email, ';') WITHIN GROUP (ORDER BY cte.email)

OVER(PARTITION BY cte.id) AS email

FROM "CTE" cte

WHERE 1=1

     AND CheckEmails = 'Correct Email'

;

 Querry Number 2 – Make Pivot Table By Years:

SELECT \*

FROM

(

SELECT salary.date, salary.id, salary.ID\_Name, salary.Company\_Name, salary.Salary\_Year, salary.Sum\_Salary

FROM Salary\_Table salary

) salary\_table\_2

PIVOT (

      SUM(Sum\_Salary) FOR Salary\_Year

      IN

      (2026,2025,2024,2023,2022,2021,2020,2019,2018,2017,2016,2015,2014,

       2013,2012,2011,2010,2009,2008,2007,2006,2005,2004,2003,

       2002,2001,2000,1999,1998,1997,1996,1995,1994,1993,1992,1991,1990)

      )

WHERE 1=1

      AND date =

      (

      SELECT MAX(salary\_table\_3.date )

      FROM Salary\_Table salary\_table\_3

      )

      ;

 Querry Number 3 – Checking the salary for the past three months:

*--------------- שימוש בפונקציית CTE*

WITH "Ranked\_Dates" AS

(

SELECT

*-------- המרה של תאריך נכונות*

to\_char(salary.date ,'yyyymmdd') AS date,

salary.Company\_Name,

salary.id,

salary.SumSalary,

*-------- שימוש בפונקציית חלון כדי להביא את השכר של החודש הקודם*

LAG(salary. SumSalary) OVER (PARTITION BY salary.id,salary.Company\_Name

ORDER BY salary.date ) AS Prev\_Month\_Salary,

*-------- שימוש בפונקציית חלון כדי להביא השכר של החודשיים הקודמים*

LAG(salary.Salary,2) OVER (PARTITION BY salary.Company\_Name,salary.id

ORDER BY salary.date ) AS Prev\_2\_Months\_Salary,

*-------- שימוש בראנק כדי לדרג את התאריכים*

RANK() OVER (PARTITION BY salary.id, salary.Company\_Name

ORDER BY salary.date ) AS rnk

FROM Salary\_Table salary

WHERE 1=1

      AND salary.Company\_Name = 'XBR'

)

SELECT \*

FROM  "Ranked\_Dates"

*--------- להביא את תאריך הנכונות האחרון*

WHERE date = (SELECT MAX(Ranked.date) FROM "Ranked\_Dates" Ranked)

ORDER BY id

;

**SQL Server**

Querry Number 1:

SELECT \*

FROM PortfolioProject..CovidDeaths

WHERE continent is not NULL

ORDER BY 3,4

SELECT \*

FROM PortfolioProject..CovidVaccinations

ORDER BY 3,4

-- Select Data that we are going to be using

SELECT location, date, total\_cases, new\_cases, total\_deaths, population

FROM PortfolioProject..CovidDeaths

WHERE continent is not NULL

ORDER BY 1,2

-- Looking at Total Cases VS Total Deaths

-- Shows likelihood dying if you contract covid in your country

SELECT location, date, total\_cases, total\_deaths,(total\_deaths/total\_cases)\*100 as DeathPercentage

FROM PortfolioProject..CovidDeaths

WHERE location like '%state%' AND continent is not NULL

ORDER BY 1,2

-- Looking at Total Cases VS Population

-- Shows what percentage of population got Covid

SELECT location, date, population, total\_cases,(total\_cases/population)\*100 as PercentPopulationInfected

FROM PortfolioProject..CovidDeaths

WHERE continent is not NULL

ORDER BY 1,2

-- Looking at Countries with Highest Rate compared to Population

SELECT location, population, MAX(total\_cases) as HighestInfectionCount, MAX((total\_cases/population))\*100 as PercentPopulationInfected

FROM PortfolioProject..CovidDeaths

WHERE continent is not NULL

GROUP BY location, population

ORDER BY PercentPopulationInfected DESC

--Showing Countries with Highest Death Count per population

SELECT location, MAX(cast(total\_deaths as int)) AS TotalDeathCount

FROM PortfolioProject..CovidDeaths

WHERE continent is not NULL

GROUP BY location

ORDER BY TotalDeathCount DESC

-- LET'S BREAK THINGS DOWN BY CONTINENT

-- Showing continents with the highest death count per population

SELECT continent, MAX(cast(total\_deaths as int)) AS TotalDeathCount

FROM PortfolioProject..CovidDeaths

WHERE continent is not NULL

GROUP BY continent

ORDER BY TotalDeathCount DESC

-- GLOBAL NUMBERS

SELECT SUM(new\_cases) AS total\_cases, SUM(cast(new\_deaths as int)) AS total\_deaths,

SUM(cast(new\_deaths as int))/SUM(New\_cases) \* 100 AS DeathPercentage

FROM PortfolioProject..CovidDeaths

WHERE continent is not NULL

--GROUP BY date

ORDER BY 1,2

--Looking at Total Population VS Vaccinations

SELECT dea.continent, dea.location, dea.date, dea.population, vac.new\_vaccinations,

SUM( cast(vac.new\_vaccinations as int))

OVER (PARTITION BY dea.location ORDER BY dea.location, dea.date) AS RollingPeopleVaccinated

-- (RollingPeopleVaccinated/population)\*100

FROM PortfolioProject..CovidDeaths dea JOIN

PortfolioProject..CovidVaccinations vac ON

dea.location = vac.location AND dea.date = vac.date

WHERE dea.continent is not NULL

ORDER BY 2,3

-- USE CTE

WITH PopVsVac (Continent, Location, Date, Population, New\_Vaccinations, RollingPeopleVaccinated)

AS

(

SELECT dea.continent, dea.location, dea.date, dea.population, vac.new\_vaccinations,

SUM(cast(vac.new\_vaccinations as int))

OVER (PARTITION BY dea.location ORDER BY dea.location, dea.date) AS RollingPeopleVaccinated

FROM PortfolioProject..CovidDeaths dea JOIN

PortfolioProject..CovidVaccinations vac ON

dea.location = vac.location AND dea.date = vac.date

WHERE dea.continent is not NULL

)

SELECT \*, (RollingPeopleVaccinated/population)\*100

FROM PopVsVac

-- TEMP TABLE

CREATE TABLE #PercentPopulationVaccinated

(

Continent nvarchar(255),

Location nvarchar(255),

Date datetime,

Population numeric,

New\_vaccinations numeric,

RollingPeopleVaccinated numeric

)

INSERT INTO #PercentPopulationVaccinated

SELECT dea.continent, dea.location, dea.date, dea.population, vac.new\_vaccinations,

SUM(cast(vac.new\_vaccinations as bigint))

OVER (PARTITION BY dea.location ORDER BY dea.location, dea.date) AS RollingPeopleVaccinated

FROM PortfolioProject..CovidDeaths dea JOIN

PortfolioProject..CovidVaccinations vac ON

dea.location = vac.location AND dea.date = vac.date

WHERE dea.continent is not NULL

SELECT \*, (RollingPeopleVaccinated/population)\*100

FROM #PercentPopulationVaccinated

-- Creating View to store data for later visualizations

Create View PercentPopulationVaccinated as

SELECT dea.continent, dea.location, dea.date, dea.population, vac.new\_vaccinations,

SUM(cast(vac.new\_vaccinations as bigint))

OVER (PARTITION BY dea.location ORDER BY dea.location, dea.date) AS RollingPeopleVaccinated

FROM PortfolioProject..CovidDeaths dea JOIN

PortfolioProject..CovidVaccinations vac ON

dea.location = vac.location AND dea.date = vac.date

WHERE dea.continent is not NULL

SELECT \*

FROM PercentPopulationVaccinated

Querry Number 2:

--A table that ranks the 10 best customers to date (without customers who only bought in 2015)--

WITH "CTE\_Q1" AS

(

SELECT customerID, (CASE WHEN YEAR([Date]) ='2015' THEN 0 ELSE 1 END) AS Y2015, Income

FROM Sheet

)

SELECT TOP 10 SUM(Y2015) AS YearYear, customerID, SUM(Income) AS TotalIncome

FROM "CTE\_Q1"

GROUP BY customerID

HAVING SUM(Y2015)!=0

ORDER BY TotalIncome DESC;

-----------------------------------------------------------

--A table that ranks the best 3 months--

WITH "CTE\_Q2" AS

(

SELECT SUM(Income) AS TotalIncome, MONTH([Date]) AS Months , YEAR([Date]) AS Years

FROM Sheet

GROUP BY MONTH([Date]), YEAR([Date])

)

SELECT TOP 3 TotalIncome, Months, Years

FROM "CTE\_Q2"

ORDER BY TotalIncome DESC;

------------------------------------------------

--Table at the monthly level of 2016--

SELECT DATEPART(MONTH, [Date]) AS MonthDate, COUNT(DISTINCT customerID) AS NumOfCustomers,

COUNT(DISTINCT InvoiceID) AS NumOfInvoices, COUNT(ProductID) AS NumOfProducts, SUM(Income) AS Total,

SUM(CASE WHEN ProductID ='A' THEN Income ELSE 0 END) AS TotalA

FROM Sheet

WHERE [Date] BETWEEN '01/01/2016' AND '12/31/2016'

GROUP BY DATEPART(MONTH, [Date]);

-----------------------------------------------

--A customer-level table that shows

--Customer ID, revenue in 2015, revenue in 2016, has he ever purchased product A (yes / no),

--has his total revenue to date been higher than 1,000 (yes / no), how many invoices does he have to date--

SELECT customerID, SUM(CASE WHEN YEAR([Date])='2015' THEN Income ELSE 0 END) AS Income\_2015,

SUM(CASE WHEN YEAR([Date])='2016' THEN Income ELSE 0 END) AS Income\_2016,

IIF(MAX(CASE WHEN ProductID ='A' THEN 1 ELSE 0 END)=1, 'Yes', 'No') AS PurchasedProductA,

( IIF(SUM(Income)<1000, 'No', 'Yes')) AS TotalIncomeAbove1000,

COUNT(invoiceID) AS TotalInvoiced

FROM Sheet

GROUP BY customerID;

----------------------------------------------

--Table of number of customers by number of invoices and shows number of invoices per customer and how many customers in total--

WITH "CTE\_Q5" AS

(

SELECT customerID, COUNT(InvoiceID) AS NumOfInvoices

FROM Sheet

GROUP BY customerID

)

SELECT COUNT(customerID) AS Customers, NumOfInvoices

FROM "CTE\_Q5"

GROUP BY NumOfInvoices

ORDER BY Customers DESC

-----------------------------------------------

--Customer-level table showing customer ID, last purchase date, penultimate purchase date--

WITH "CTE\_Q6" AS

(

SELECT customerID, [Date], DENSE\_RANK () OVER (PARTITION BY customerID ORDER BY [Date] DESC) RankDate

FROM Sheet

)

SELECT DISTINCT customerID, [DATE], RankDate

FROM "CTE\_Q6"

WHERE RankDate<3

ORDER BY customerID, [Date] DESC