**Part 1 – C# Code**

**1.**

The line error - case enforcedOption.All its should be RoomAvailableOption.All.

public class RoomSearchFilter

{

private readonly RoomAvailableOption enforcedOption;

public RoomSearchFilter(RoomAvailableOption enforcedOption)

{

this.enforcedOption = enforcedOption;

}

public bool IncludeInFinalResults(Room room)

{

bool isAvailable = CheckIfRoomAvailable(room);

return enforcedOption switch

{

RoomAvailableOption.All => true,

RoomAvailableOption.AvailableOnly => isAvailable,

RoomAvailableOption.NotAvailableOnly => !isAvailable,

\_ => throw new ArgumentOutOfRangeException($"Error option: {enforcedOption}")

};

}

}

**2.**

Option 1:

public User GetUserFromDB(int userId)

{

DataRow dr = DataRepository.GetUserById(userId);

User user = new User { UserId = userId };

foreach (var prop in typeof(User).GetProperties())

{

string columnName = prop.Name;

prop.SetValue(user, GetValueOrDefault(dr, columnName));

}

return user;

}

private string GetValueOrDefault(DataRow dr, string columnName)

{

var value = dr[columnName];

return value != DBNull.Value && value != null ? value.ToString() : string.Empty;

}

If its string only can do: return dr[columnName] as string ?? string.Empty;

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

Option 2 if I use **Entity Framework:**

public User GetUserFromDB(int userId)

{

using (var context = new YourDbContext())

{

return context.Users

.Where(u => u.UserId == userId)

.Select(u => new User

{

UserId = u.UserId,

FirstName = u.FirstName ?? string.Empty, // ערך ברירת מחדל אם null

LastName = u.LastName ?? string.Empty,

Address = u.Address ?? string.Empty,

CityName = u.CityName ?? string.Empty,

CountryName = u.CountryName ?? string.Empty,

Email = u.Email ?? string.Empty

})

.FirstOrDefault();

}

}

**Part 3 – SQL**

**1.**

SELECT

a.BANK\_ID,

a.BRANCH\_ID,

a.ACCOUNT\_NUM,

a.CURRENCY,

m.AMOUNT

FROM

ACCOUNTS a

INNER JOIN

AMOUNTS m

ON

a.BANK\_ID = m.ACCOUNT\_REC AND

a.BRANCH\_ID = m.BRANCH\_ID AND

a.ACCOUNT\_NUM = m.ACCOUNT\_NUM

ORDER BY

a.BANK\_ID, a.BRANCH\_ID, a.ACCOUNT\_NUM;

**2.**

CREATE OR REPLACE FUNCTION CALC\_AMT (

P\_AMT\_TYPE VARCHAR2,

P\_ID NUMBER,

P\_BS\_DATE DATE

) RETURN NUMBER IS

V\_AMT NUMBER;

BEGIN

SELECT MAX(a.AMT)

INTO V\_AMT

FROM (

SELECT AMT, BUSINESS\_DATE, ID FROM MR\_SMT WHERE P\_AMT\_TYPE = 'C'

UNION ALL

SELECT AMT, BUSINESS\_DATE, ID FROM MR\_OP WHERE P\_AMT\_TYPE <> 'C'

) a

WHERE a.BUSINESS\_DATE BETWEEN P\_BS\_DATE - 7 AND P\_BS\_DATE

AND a.ID = P\_ID;

RETURN V\_AMT;

END;

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

SELECT

t.BRANCH,

t.ACCOUNT,

t.BUSINESS\_DATE,

t.ID,

CASE

WHEN T.AMT\_TYPE = 'C' THEN MAX(s.AMT)

ELSE MAX(o.AMT)

END AS CALC\_AMT,

t.FILENAME,

t.POPULATION\_DATE

FROM

TMP\_TAB t

LEFT JOIN MR\_SMT s

ON t.AMT\_TYPE = 'C'

AND s.BUSINESS\_DATE BETWEEN t.BUSINESS\_DATE - 7 AND t.BUSINESS\_DATE

AND s.ID = t.ID

LEFT JOIN MR\_OP o

ON t.AMT\_TYPE <> 'C'

AND o.BUSINESS\_DATE BETWEEN t.BUSINESS\_DATE - 7 AND t.BUSINESS\_DATE

AND o.ID = t.ID

GROUP BY

t.BRANCH, t.ACCOUNT, t.BUSINESS\_DATE, t.ID, t.FILENAME, t.POPULATION\_DATE, t.AMT\_TYPE;

also add index:

CREATE INDEX IDX\_MR\_SMT\_BUSINESS\_DATE\_ID ON MR\_SMT (BUSINESS\_DATE, ID); // MR\_SMT  
CREATE INDEX IDX\_MR\_OP\_BUSINESS\_DATE\_ID ON MR\_OP (BUSINESS\_DATE, ID); // MR\_OP

Also maybe on:  
CREATE INDEX IDX\_MR\_SMT\_BUSINESS\_DATE\_ID\_AMT ON MR\_SMT (BUSINESS\_DATE, ID, AMT);

CREATE INDEX IDX\_TMP\_TAB\_BUSINESS\_DATE\_ID ON TMP\_TAB (BUSINESS\_DATE, ID, AMT\_TYPE);