ZAJĘCIA II

ZAPYTANIA PROSTE

Zad. 1
Podać nazwiska i zarobki każdego pracownika, dane posortować malejąco wg daty zatrudnienia.

Zad. 2
Wyznaczyć dla każdego pracownika jego nazwisko oraz jego zarobki (włącznie z premią), nadać etykietę zarobkom.

Zad. 3
Sprawdzić jaki stopień wynagrodzenia posiada każdy zatrudniony. Wyświetlić nazwisko i stopień zarobków.

Zad. 4
Wypisać nazwiska tych pracowników, którzy pracują w Białymstoku lub Olsztynie.

Zad. 5
Podać nazwiska tych pracowników, którzy nie mają kierowników.

Zad. 6
Podać listę pracowników, którzy pracują na stanowisku pracy, którego nazwa rozpoczyna się od litery A.

Zad. 7
Wyznaczyć listę pracowników, których nazwiska rozpoczynają się od litery N lub M i którzy nie otrzymują premii.

Zad. 8
Podać nazwiska i zarobki tych pracowników, których stopień wynagrodzenia jest równy 1 lub 2.

Zad. 9
Podać listę pracowników wraz z nazwami projektów z którymi byli/są związani.

Zad. 10
W jakich departamentach prowadzone są prace nad poszczególnymi projektami. Lista powinna zawierać odpowiednio nazwy departamentów i projektów.

Zad. 11
Podać nazwiska osób pracujących na stanowisku ‘sprzedawca’ z departamentu ‘Departament 1’.

POMOC

SELECT nazwy kolumn (z nadanymi etykietami, lub bez) po przecinkach
FROM nazwy tabel (z nadanymi etykietami, lub bez) po przecinkach
[
WHERE warunki
GROUP BY kolumny, po których odbywa się grupowanie (nazwy kolumn po przecinkach)
HAVING warunki dla nowo powstałych grup
ORDER BY kolumny (nazwy kolumn po przecinkach), według których odbywać się będzie porządkowanie wyników (rosnąco ASC- domyślne, malejąco DESC)
];

Funkcja nvl(war1,war2)

gdy odczytana wartość kolumny o nazwie war1 jest NULL, to zamieniana jest ta wartość na war2.

Słowo kluczowe DISTINCT

służy do tego, aby wypisywanie wierszy odbywało się bez powtórzeń

```
SELECT DISTINCT nazwisko
FROM pracownik;
```

wypisane zostaną wszystkie różne nazwiska (jeśli jest pięciu Kowalskich, to nazwisko Kowalski zostanie wypisane jeden raz, bez słowa DISTINCT wypisanych by zostało pięć wierszy z nazwiskiem Kowalski).

Wynikiem zapytania:

```
SELECT DISTINCT nazwisko, pensja
FROM pracownik;
```

będą wiersze, w których nie powtórzy się kombinacja wartości np. 'N1', 1000 (tzn. że jeśli byłoby parę osób o takim samym nazwisku zarabiających tyle samo, to wypisana zastałaby tylko jedna kombinacja tych wartości).

Klauzula WHERE

1. Warunki są łączone spójnikami AND lub OR

   ```
   WHERE pensja>300 AND nr_departamentu=30
   ```

2. BETWEEN war1 AND war2

   ```
   WHERE pensja BETWEEN 300 AND 500 to samo, co:
   WHERE pensja>=300 AND pensja<=500
   ```

3. IS NULL (IS NOT NULL)

   WHERE kolumna IS NULL (wybierane są wiersze, w których wartość danej kolumny jest NULL, tzn. komórka jest pusta)

4. LIKE

   WHERE kolumna LIKE 'A%' (tzn., że wybierane są wiersze, w których wartość danej kolumny rozpoczyna się literą A, po literze A może być wiele znaków)

   % wiele znaków
   * jeden znak

Wybieranie danych z większej niż jeden liczby tabel

Do każdej kolumny można odwoływać się w następujący sposób:

```
tabela.kolumna
```

Nie jest to jednak konieczne. Konieczność taka pojawia się w sytuacji, gdy wybieramy dane z większej ilości tabel i nazwa danej kolumny występuje w co najmniej dwóch tabelach. Załóżmy, że w tabelach `tabela1` i `tabela2` jest kolumna o nazwie **`k`**. Wtedy odwołanie postaci:

```
SELECT k
FROM tabela1, tabela2;
```

jest niejednoznaczne i należy zastąpić **`k`** jednym z odwołań: `tabela1.k` lub `tabela2.k`.

Jeśli wybierane są dane z większej ilości tabel i jeśli istnieje jakiś schemat według którego te wiersze mają być wybierane to w klauzuli `WHERE` należy ten schemat podać.

Przykład:

```
WHERE tab1.kolumna1=tab2.kolumna2
```

tzn., że wartości w wymienionych kolumnach mają sobie odpowiadać.

Bez podania warunku łączenia wynikiem zapytania:

```
SELECT kolumna1, kolumna3
FROM tabela1, tabela2
```

będzie iloczyn kartezjański wierszy z obu tabel.

Łączenie tabel

`LEFT OUTER JOIN`

W tym przypadku zwracane są wszystkie wiersze tabeli po lewej stronie, nawet gdy nie ma odpowiedników w tabeli po prawej stronie (w klauzuli `WHERE`).

```
SELECT kol_A, kol_B
FROM tab_A, tab_B
WHERE tab_A.łącze=tab_B.łącze (+);
```

Gdzie `łącze` to nazwa kolumny.

`RIGHT OUTER JOIN`

W tym przypadku zwracane są wszystkie wiersze tabeli po prawej stronie, nawet gdy nie ma odpowiedników w tabeli po lewej stronie (w klauzuli `WHERE`).

```
SELECT kol_A, kol_B
FROM tab_A, tab_B
WHERE tab_A.łącze(+)=tab_B.łącze;
```