Bases de datos

Unidade 5: Consultas resumo con agrupamentos

Índice

1. A03. Consultas resumo con agrupamentos

1.1 Introdución

1.1.1 Obxectivos

Os obxectivos desta actividade son:

- Realizar consultas agrupando filas do conxunto de resultados e facer cálculos coas función de agrupamento.
- Realizar consultas establecendo condicións para os grupos.

1.1.2 Software

Utilizarase a plataforma WAMP (Windows-Apache-MySQL-PHP) WampServer 2.5 (última versión estable en outubro 2015), que inclúe MySQL Community Edition 5.6.17 como SXBDR (Sistema Xestor de Bases de Datos Relacional). As razóns de utilización deste software son que:

- É software libre, polo que o alumnado poderá descargalo de forma gratuíta e utilizalo legalmente na súa casa.
- É unha forma sinxela de facer a instalación do software necesario para desenvolver aplicacións web.

Páxina oficial de WampServer: http://www.wampserver.com

Páxina oficial de MySQL: https://www.mysql.com/

Utilizarase MySQL Workbench 6.3 como ferramenta cliente gráfica xa que é a recomendada por MySQL en outubro de 2015, aínda que tamén poderían utilizarse outras como phpMyAdmin, EMS MyManager, ou MySQL Query Browser.

Normalmente, para a proba das consultas realizadas nesta actividade, mostrarase a zona de manipulación de datos de Workbench, coas filas que forman a táboa de resultados. Para completar a anterior información ou cando a consulta non pode mostrarse enteira xa que devolve moitas filas, mostrarase ademais a zona de saída (output) coa información do estado da execución da consulta e o número de filas que devolve

En https://www.mysql.com/products/workbench/ pode obterse información detallada sobre a ferramenta MySQL Workbench e descargar o software.

En http://dev.mysql.com/doc/index-gui.html pode descargarse o manual de MySQL Workbench.

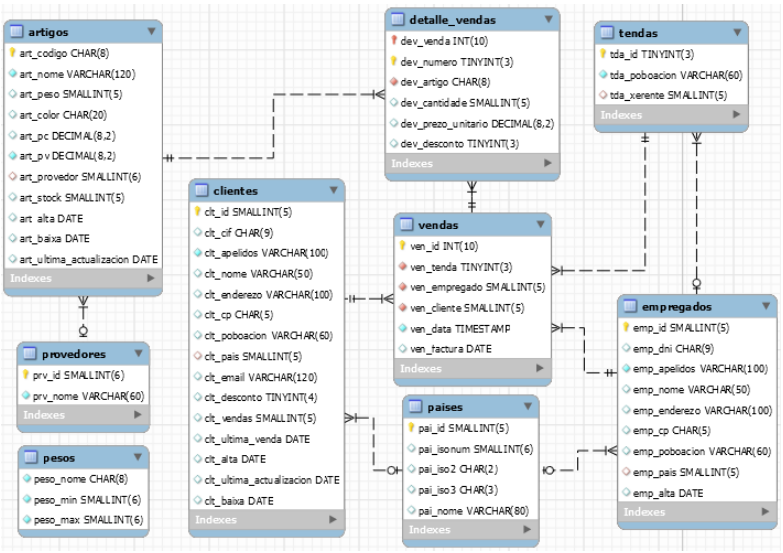
Anexiónase a esta actividade unha guía básica de MySQL Workbench 6.3.

1.1.3 Bases de datos de traballo

As bases de datos *tendaBD*, *traballadores* e *practicas5* utilizaranse para os exemplos e tarefas desta actividade. Antes de empezar a probar os exemplos ou realizar as tarefas, hai que executar os scripts de creación no servidor e poñer en uso a base de datos correspondente. Os scripts atópanse no cartafol anexo a esta actividade descrito no apartado '3.3 Material auxiliar'.

1.1.3.1 Base de datos tendaBD

A base de datos *tendaBD* serve para controlar as vendas dunha cadea de tendas. Gárdanse nela os datos das vendas que se realizan, das tendas nas que se fan as vendas, dos artigos vendidos, e dos clientes. As táboas desta base de datos que se van a utilizar nesta actividade móstranse no seguinte diagrama entidade relación deseñado con Workbench e descríbense a continuación.



- Táboa *empregados*

Nome columna	Tipo	Null	Clave	Observacións
emp_id	smallint unsigned	Non	Primaria	Identificador do empregado. Numéranse de 1 en adiante de forma automática.
emp_dni	char(9)			DNI do empregado.
emp_apelidos	varchar(100)	Non	Índice	Apelidos do empregado.
emp_nome	varchar(50)			Nome do empregado.
emp_enderezo	varchar(100)			Enderezo do empregado.
emp_cp	char(5)			Código postal do empregado.
emp_poboacion	varchar(60)			Poboación do empregado.
emp_pais	smallint unsigned		Foránea	Código do país segundo a táboa de países.
emp_alta	date			Data na que se deu de alta o empregado.

- Táboa *pesos*

Nome columna	Tipo	Null	Clave	Observacións
peso_nome	char(8)	Non		Nome que describe o tipo de peso.
peso_min	smallint	Non		Peso mínimo para ese nome.
peso_max	smallint	Non		Peso máximo para ese nome.

- Táboa *clientes*

Nome columna	Tipo	Null	Clave	Observacións
clt_id	smallint unsigned	Non	Primaria	Identificador do cliente. Numeraranse de 1 en adiante de forma automática.
clt_cif	char(9)		Única	
clt_apelidos	varchar(100)	Non	Índice	Apelidos ou razón social do cliente.
clt_nome	varchar(50)			Nome ou tipo de sociedade (SL, SA, ...) do cliente.
clt_enderezo	varchar(100)			
clt_cp	char(5)			Código postal do cliente.
clt_poboacion	varchar(60)			
clt_pais	smallint unsigned		Foránea	Código do país segundo a táboa de países.
clt_email	varchar(120)			
clt_desconto	tinyint			Porcentaxe de desconto aplicable ao cliente.
clt_vendas	smallint unsigned			Número de vendas feitas ao cliente.
clt_ultima_venda	date			Data da última venda feita ao cliente.
clt_alta	date	Non		Data na que se deu de alta ao cliente.
clt_ultima_actualizacion	date			Data da última vez que se fixeron cambios nos datos do cliente.
clt_baixa	date			Data na que se deu de baixa ao cliente.

- Táboa *artigos*

Nome columna	Tipo	Null	Clave	Observacións
art_codigo	char(8)	Non	Primaria	Toma valores entre 1 e 200.000.
art_nome	varchar(120)	Non	Índice	Nome ou descrición do artigo.
art_peso	smallint unsigned			Peso en gramos. Valor numérico enteiro.
art_color	char(20)			Cor do artigo
art_pc	decimal(8,2)			Prezo de compra do artigo.
art_pv	decimal(8,2)	Non		Prezo de venda do artigo.
art_provedor	smallint		Foránea	Identificador do provedor.
art_stock	smallint unsigned			Número de unidades do artigo dispoñibles no almacén.
art_alta	date	Non		Data na que se deu de alta o artigo.
art_baixa	date			Data na que se deu de baixa o artigo.
art_ultima_actualizacion	date			Data da última vez que se fixeron cambios nos datos do artigo.

- Táboa *paises*

Nome columna	Tipo	Null	Clave	Observacións
pai_id	smallint unsigned	Non	Primaria	Identificador do país. Numeraranse de 1 en adiante de forma automática.
pai_isonum	smallint			Número de país segundo a norma ISO 3166-1:2013.[1]
pai_iso2	char(2)			Código de país de 2 carácteres segundo a norma ISO 3166-1:2013.
pai_iso3	char(3)			Código de país de 3 carácteres segundo a norma ISO 3166-1:2013.
pai_nome	varchar(80)			Nome do país.

[1] Máis información sobre a norma ISO 3166-1:2013 en https://es.wikipedia.org/wiki/ISO_3166-1

- Táboa *provedores*

Nome columna	Tipo	Null	Clave	Observacións
prv_id	smallint	Non	Primaria	Identificador do provedor.
prv_nome	varchar(60)	Non		Nome do provedor.

- Táboa *tendas*

Nome columna	Tipo	Null	Clave	Observacións
tda_id	tinyint unsigned	Non	Primaria	Identificador da tenda. Numéranse do 1 en adiante de forma automática.
tda_poboacion	varchar(60)	Non		Poboación na que está situada a tenda.
tda_xerente	smallint unsigned		Foránea	Identificador do empregado que é xerente da tenda.

- Táboa *vendas*

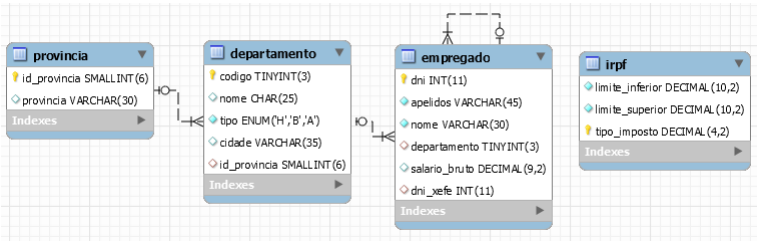
Nome columna	Tipo	Null	Clave	Observacións
ven_id	int unsigned	Non	Primaria	Identificador da venda. Numeraranse de 1 en adiante de forma automática.
ven_tenda	tinyint unsigned	Non	Foránea	Identificador da tenda na que se fixo a venda.
ven_empregado	smallint unsigned	Non	Foránea	Identificador do empregado que fixo a venda.
ven_cliente	smallint unsigned	Non	Foránea	Identificador do cliente ao que se fixo a venda.
ven_data	date	Non		Data e hora na que se fixo a venda.
ven_factura	date			Data da factura na que se inclúe esta venda.

- Táboa *detalle_vendas*

Nome columna	Tipo	Null	Clave		Observacións
dev_venda	int unsigned	Non	Primaria	Foránea	Identificador da venda á que corresponde a liña de detalle.
dev_numero	tinyint unsigned	Non			Número da liña de detalle dentro da venda.
dev_artigo	char(8)	Non	Foránea		Identificador do artigo vendido.
dev_cantidade	smallint unsigned	Non			Número de unidades vendidas.
dev_prezo_unitario	decimal(8,2) unsigned	Non			Prezo por cada unidade vendida.
dev_desconto	tinyint unsigned	Non			Porcentaxe de desconto aplicado.

1.1.3.2 Base de datos practicas5

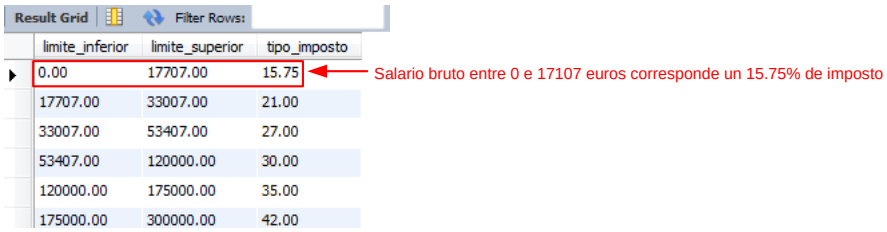
A base de datos *practicas5* está creada con fins didácticos para realizar os exemplos de consultas nesta unidade. Está formada por un grupo de táboas, relacionadas entre si, tal e como se mostra no seguinte diagrama entidade relación deseñado con Workbench e se describe a continuación.



- Táboa *empregado*. A columna *departamento* é unha clave foránea que contén o código do departamento no que traballa o empregado, e fai referencia á columna *codigo* da táboa *departamento*. Os valores que toma a columna *departamento* teñen que coincidir cos que toma a columna *codigo* da táboa *departamento*, ou ser NULL no caso que o

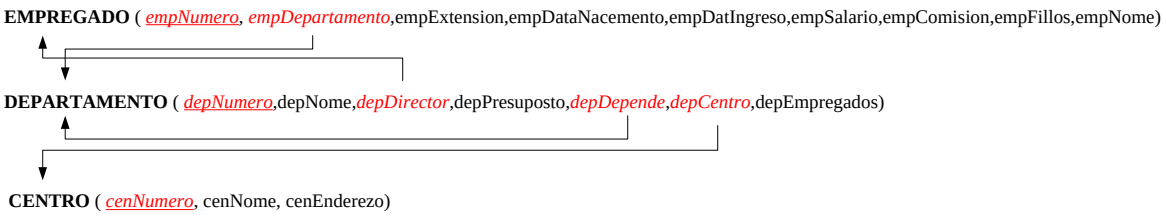
empregado non teña asignado ningún departamento. A columna *dni_xefe* é outra clave foránea que contén o dni doutro empregado que sería o seu xefe, ou o valor NULL no caso que non tivera xefe.

- Táboa *departamento*. A columna *id_provincia* é unha clave foránea que fai referencia á columna *id_provincia* da táboa *provincia*.
- Táboa *irpf*. Contén a porcentaxe de imposto que hai que aplicarlle a cada empregado, en función do seu salario bruto, dependendo dos límites entre os que se atope. Esta táboa podería conter unha información similar a esta:



1.1.3.3 Base de datos traballadores

A base de datos *traballadores* serve para levar control dos empregados, departamentos e centros dunha empresa. Está formada por un grupo de táboas, relacionadas entre si, tal e como se mostra no seguinte grafo relacional e se describe a continuación. As táboas son MyIsam (non transaccionais) e por tanto non teñen definidas claves foráneas.



- Táboa centro

Nome columna	Tipo	Null	Clave	Observacións
cenNumero	int	Non	Primaria	Número co que se identifica.
cenNome	char(30)		Índice	Nome.
cenEnderezo	char(30)			Enderezo.

- Táboa empregado

Nome columna	Tipo	Null	Clave	Observacións
empNumero	int	Non	Primaria	Número co que se identifica.
empDepartamento	int	Non	Índice	Número do departamento no que traballa.
empExtension	smallint	Non		Extensión telefónica para o empregado. Pode compartirse entre empregados de diferentes departamentos.
empDataNacemento	date			Data de nacemento.
empDataIngreso	date			Data de ingreso na empresa.
empSalario	decimal(6,2)			Salario mensual en euros.
empComision	decimal(6,2)			Comisión mensual.
empFillos	smallint			Número de fillos.
empNome	char(20)	Non	Índice	Nome do empregado coa forma: primeiro apelido, nome.

- Táboa departamento

Nome columna	Tipo	Null	Clave	Observacións
depNumero	int	Non	Primaria	Número co que se identifica.
depNome	char(20)		Índice	Nome.
depDirector	int	Non	Índice	Número do empregado director do departamento.
deptipoDirector	char(1)			Tipo de directo: P (en propiedade, é dicir, titular), F (en funcións).
depPresuposto	decimal(9,2)			Cantidade en euros de presuposto anual.
depDepende	int		Índice	Número do departamento do que depende.
depCentro	int		Índice	Número do centro ao que pertence.
depEmpregados	smallint unsigned			Número de empregados que traballan no departamento.

1.2 Actividade

1.2.1 Consultas con agrupamento de filas

Chámase grupo á información correspondente a un conxunto de filas que teñen o mesmo valor nunha ou varias columnas dunha táboa denominadas columnas de agrupamento.

Existe un tipo de consultas con agrupamento de filas que dá como resultado unha táboa resumo que ten como columnas, as columnas de agrupamento e os cálculos, e ten unha fila por cada valor distinto que toman as columnas de agrupamento. Os cálculos realízanse coas funcións de agrupamento ou agregado que actúan sobre as filas de cada grupo en lugar de actuar sobre todas as filas seleccionadas como se fixo ata este momento.

A cláusula GROUP BY utilízase para facer consultas con agrupamento de filas.

1.2.2 Cláusula GROUP BY

Permite especificar as columnas de agrupamento. Sintaxe:

```
[GROUP BY {nome_columna | expresión | posición} [ASC|DESC], ... [WITH ROLLUP]]
```

- As columnas de agrupamento pódense representar por unha lista dun ou máis nomes de columnas, expresións, ou ben o número de orde que ocupa a columna dentro da lista de selección, igual que na cláusula ORDER BY. Non é recomendable utilizar o número de orde da columna porque no caso de cambiar a orde das columnas na lista de selección, ou engadir ou eliminar algunha columna, pódense producir resultados inesperados se non se cambia a cláusula GROUP BY.
- O modificador WITH ROLLUP, permite engadir filas na táboa resumo. Estas filas corresponden a cálculos feitos creando un grupo novo que contén todas as filas, e creando grupos a varios niveis no caso de agrupar por máis dunha columna.

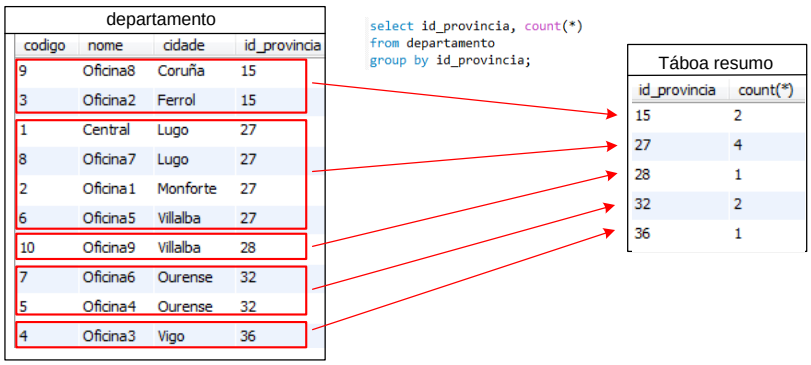
En MySQL, cando se utiliza esta cláusula, ordénanse as filas polas columnas de agrupamento como se fose unha cláusula ORDER BY, antes de formar os grupos. As opcións [ASC | DESC] son unha extensión de MySQL ao estándar, e indican a orde na que se ordenan as filas antes de formar os grupos (ASC = ascendente, DESC = descendente); se non se indica nada, tómase ASC como valor por defecto.

Exemplo: calcular o número de departamentos que hai en cada provincia tendo en conta os datos contidos na táboa *departamento* da base de datos *practicas5*.

```
select id_provincia, count(*)
from departamento
group by id_provincia;
```

O SXBDR selecciona todas as filas da táboa *departamento* e as ordena pola columna *id_provincia*, que é a columna de agrupamento, para que estean xuntos todos os departamentos que pertencen á mesma provincia; despois vai collendo fila a fila e cada vez que cambia o valor da columna *id_provincia* prodúcese un ruptura e fórmase un novo grupo. Esta explicación do funcionamento ten unha finalidade didáctica, pero a forma en que se executa internamente a sentenza a decide o optimizador de consultas, e pode variar dun xestor a outro.

Representación gráfica do resultado da execución:



Na parte esquerda da imaxe móstranse as filas e columnas que forman a táboa *departamento*, remarcando as filas que teñen o mesmo valor na columna *id_provincia*. Cada un dos marcos representa un grupo formado por todas as filas que teñen o mesmo valor na columna de agrupamento.

Na parte dereita da imaxe, móstrase a táboa resumo resultante que ten unha fila por cada grupo. As columnas que ten a táboa resumo son, en primeiro lugar, a columna de agrupamento (*id_provincia*) para mostrar o valor que identifica ás filas de cada grupo, e en segundo lugar, os cálculos que se queren facer coas filas do grupo, utilizando neste caso, a función de agregado COUNT(*) que conta o número de filas do grupo. Pódese observar que non ten sentido poñer calquera outra columna da táboa *departamento* na táboa resumo porque pode que teña valores distintos en distintas filas do grupo e o valor que se mostraría na columna sería inconsistente.

Algunhas consideracións a ter en conta cando se utiliza a cláusula GROUP BY:

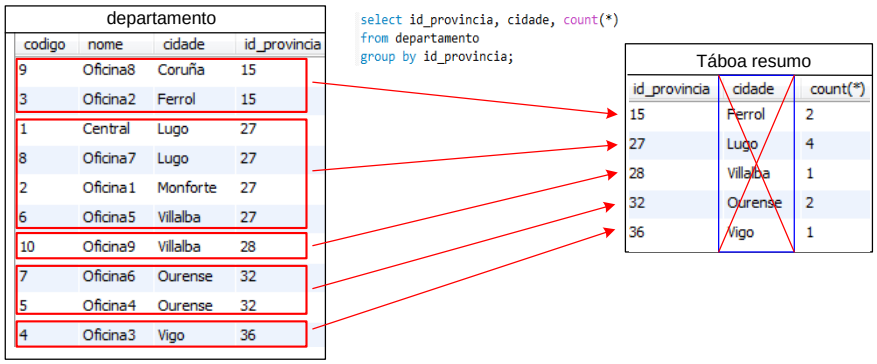
- Para indicar as columnas de agrupamento, pódense utilizar os alias de columna.
- Pode haber diferenzas no tratamento dos valores NULL á hora de formar grupos. Algúns SXBDR consideran que non se agrupan as filas que conteñan o valor NULL na columna de agrupamento e outros consideran que todas as filas que teñan o valor NULL na columna de agrupamento forman un grupo. MySQL utiliza a segunda opción.
- O estándar SQL establece que na lista de selección da cláusula SELECT só se pode facer referencia ás columnas polas que se agrupa, funcións de agregado, constantes, ou expresións que combinen ás anteriores. Na maioría dos SXBDR, no caso de incluír calquera outra columna daría lugar a un erro. MySQL estende o uso de GROUP BY permitindo facer referencia na lista de selección de columnas distintas das columnas de agrupamento, e deixa en mans do usuario controlar que as columnas tomen un valor único para cada grupo. Pódese cambiar este comportamento, e desactivar as extensións de MySQL para GROUP BY, habilitando o modo SQL 'ONLY_FULL_GROUP_BY'.
- Para cada grupo (cada valor distinto que tomen as columnas de agrupamento) créase unha fila na táboa resumo. Todos os elementos da lista de selección deben ter un valor único para cada grupo.

Os seguintes exemplos mostran dúas situacións diferentes producidas ao incluír na selección columnas distintas das columnas de agrupamento.

- Exemplo incluíndo a columna *cidade*.

```
select id_provincia, cidade , count(*)
from departamento
group by id_provincia;
```

Representación gráfica do resultado da execución:



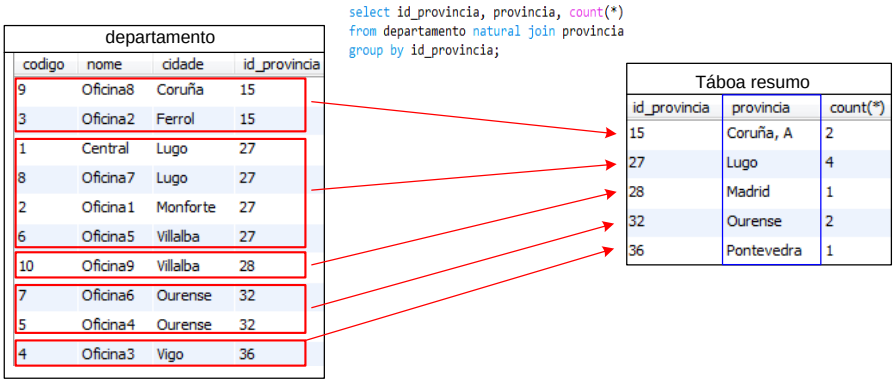
O contido da columna *cidade* na táboa *resumo* é inconsistente. A columna pode tomar máis dun valor para cada grupo. Na táboa resumo móstrase na columna un valor correspondente a unha fila do grupo, pero non representa unha información resumo do grupo para os casos de grupos que teñen máis dunha fila. Por exemplo, a primeira fila da táboa resumo parece que nos indica que na cidade de Ferrol hai dous departamentos, cando na cidade só hai un departamento, o 2 corresponde ao número de departamentos que hai na provincia que ten o *id_provincia* (columna de agrupamento) 15.

MySQL non mostra ningunha mensaxe de erro no momento de executar a consulta pero a sentenza non é correcta e os datos son inconsistentes. A maioría dos SXBDR, ao executar esta sentenza mostrarían un erro indicando que na lista de selección hai unha columna que non está en GROUP BY.

- Exemplo incluíndo a columna *provincia* que contén o nome da provincia.

```
select id_provincia, provincia, count(*)
from departamento natural join provincia
group by id_provincia;
```

Representación gráfica do resultado da execución:



A columna *provincia* da táboa resumo toma valores consistentes, porque para cada valor de *id_provincia* (columna de agrupamento) só hai un valor posible para a columna *provincia*. Neste caso para cada grupo a columna *provincia* só toma un valor independentemente do número de filas que teña o grupo, polo que a sentenza é correcta.

1.2.2.1 Agrupamento por máis dunha columna

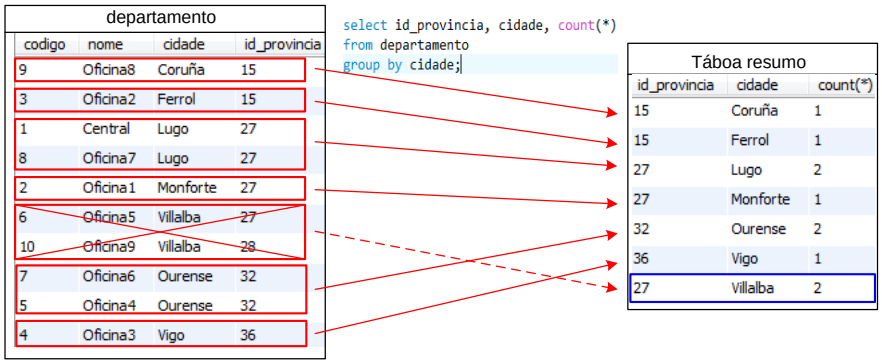
É posible agrupar por máis dunha columna, escribindo a lista das columnas polas que se quere agrupar, separadas por comas na cláusula GROUP BY.

Exemplo: contar o número de departamentos que hai en cada cidade.

- Solución 1 agrupando só pola columna *cidade*:

```
select id_provincia, cidade, count(*)
from departamento
group by cidade;
```

Representación gráfica do resultado da execución:

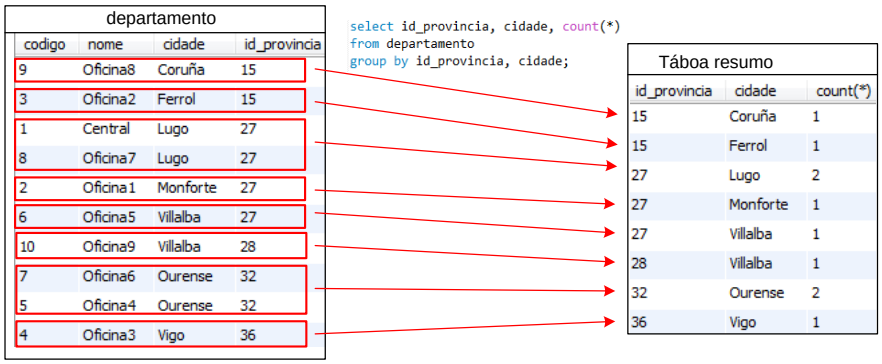


Créase unha única fila para a cidade 'Villalba' na táboa resumo, sen ter en conta que o mesmo nome de cidade se repite en dúas provincias distintas (27 e 28). Esta solución non sería correcta porque mostra que na cidade 'Villalba' da provincia 27 hai dous departamentos cando só hai un; non ten en conta que se trata de cidades diferentes.

- Solución 2 agrupando polas columnas *id_provincia* e *cidade*.

```
select id_provincia, cidade, count(*)
from departamento
group by id_provincia, cidade;
```

Representación gráfica do resultado da execución:



Esta solución sería a correcta porque fai primeiro o agrupamento por provincias, e dentro de cada provincia, o agrupamento por cidade. Neste caso, a cidade 'Villalba' móstrase dúas veces, unha asociada ao id_provincia 27 e outra ao id_provincia 28.

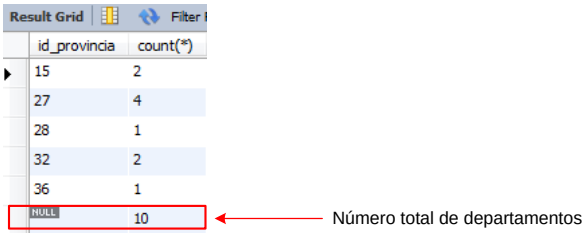
1.2.2.2 Modificador WITH ROLLUP

Permite mostrar novas filas na táboa de resumo. Estas filas representan operacións de resumo para distintos niveis de análises. Resumo do comportamento do modificador nun servidor MySQL:

- No caso de agrupar só por unha columna, o modificador engade unha nova fila na táboa resumo na que se mostran os cálculos para un novo grupo formado por todas filas seleccionadas. Nesta fila móstrase o valor NULL na columna de agrupamento.

  Exemplo: contar o número de departamentos que hai en cada provincia, e o número total de departamentos da empresa, tendo en conta os datos contidos na táboa *departamento* da base de datos *practicas5*.

```
/* agrupar por unha columna co modificador WITH ROLLUP*/
select id_provincia, count(*)
from departamento
group by id_provincia with rollup;
```



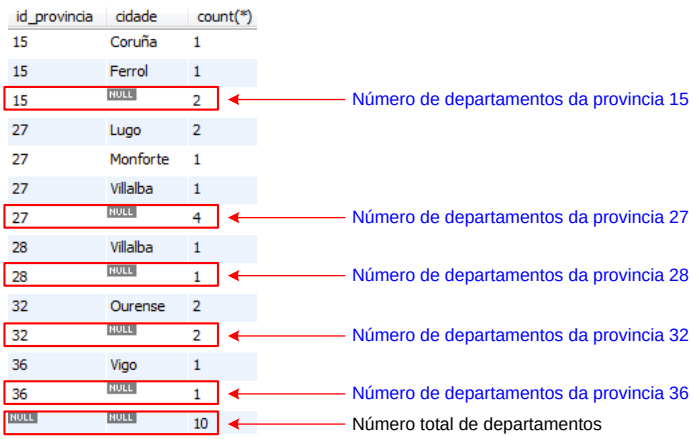
  No resultado da execución pódese ver que ademais de mostrar unha fila por cada grupo, se engade unha fila máis co cálculo feito considerando un novo grupo formado por todas as filas da táboa *departamento*.

- No caso de agrupar por máis dunha columna, o modificador engade unha nova fila na táboa resumo cada vez que hai un cambio de valor nalgunha das columnas de agrupamento, ademais da fila correspondente ao cálculo para todas as filas.

  Exemplo: contar o número de departamentos que hai en cada cidade, os que hai por cada provincia e o número total de departamentos da empresa.

```
/* agrupar por máis dunha columna co modificador WITH ROLLUP*/
select id_provincia, cidade, count(*)
from departamento
group by id_provincia, cidade with rollup;
```

  Resultado a execución:



  No resultado da execución pódese ver que ademais de mostrar unha fila por cada grupo, se engade unha fila máis por cada provincia, que é a primeira columna da lista de columnas de agrupamento, e ao final engade unha fila máis co cálculo feito considerando un novo grupo formado por todas as filas da táboa *departamento*.

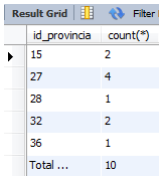
Algunhas consideracións sobre o uso de ROLLUP en MySQL:

- Cando se usa este modificador non se pode utilizar a cláusula ORDER BY.
- Utilizar a cláusula LIMIT co modificador pode producir resultados difíciles de interpretar, porque a cláusula aplícase despois do modificador así que no límite cóntanse as filas extra engadidas polo modificador.

O resultado obtido cando se agrupa por unha soa columna e se utiliza ROLLUP é o mesmo que utilizando unha UNION de consultas.

```
/* Agrupar por unha columna co modificador WITH ROLLUP*/
select id_provincia, count(*)
from departamento
group by id_provincia with rollup;
/* Alternativa utilizando union*/
select id_provincia, count(*)
from departamento
group by id_provincia
union
select 'Total ...', count(*)
from departamento;
```

Ambas opcións obteñen o mesmo resultado:



id_provincia	count(*)
15	2
27	4
28	1
32	2
36	1
Total ...	10

Tarefa 1. Realizar consultas agrupando as filas en táboas resumo.

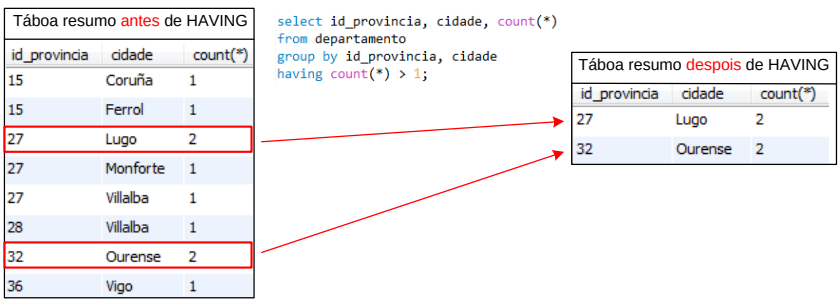
1.2.3 Cláusula HAVING

Esta cláusula está asociada coa cláusula GROUP BY. Permite establecer condicións para descartar aqueles grupos que non cumpran esas condicións. É similar á cláusula WHERE, pero a diferenza entre elas é que mentres WHERE analiza a condición nas filas das táboas de orixe da consulta (as especificadas na cláusula FROM), a cláusula HAVING o fai sobre a táboa resumo, é dicir, sobre os grupos que se formaron despois de agrupar.

Exemplo: contar os departamentos que hai en cada cidade, e mostrar só as cidades que teñen máis dun departamento.

```
select id_provincia, cidade, count(*)
from departamento
group by id_provincia, cidade
having count(*) > 1;
```

Representación gráfica do resultado da execución:



Ao aplicar a condición establecida ha cláusula HAVING, elimináronse da táboa resumo os grupos que non cumpren a condición.

Algunhas consideracións a ter en conta cando se utiliza a cláusula HAVING:

- O estándar ANSI SQL establece que as condicións desta cláusula só poden referirse ás columnas de agrupamento (as que figuran en GROUP BY), ou ás funcións de agregado. A partir da versión 5.0.2 MySQL estende este comportamento, e admite utilizar nas condicións do HAVING outras columnas diferentes das relacionadas en GROUP BY, e subconsultas. Pódese cambiar este comportamento, e desactivar estas extensións de MySQL para GROUP BY, habilitando o modo SQL 'ONLY_FULL_GROUP_BY'.
- Non se debe utilizar HAVING para condicións que deban estar na cláusula WHERE.
- Pódense utilizar os alias para escribir as condicións.

Tarefa 2. Realizar consultas establecendo condicións para os grupos da táboa resumo.

1.3 Tarefas

As tarefas propostas son as seguintes:

- Tarefa 1. Realizar consultas agrupando as filas en táboas resumo.
- Tarefa 2. Realizar consultas establecendo condicións para os grupos da táboa resumo.

1.3.1 Tarefa 1. Realizar consultas agrupando as filas en táboas resumo

A tarefa consiste en realizar as seguintes consultas agrupando filas.

Sobre a base de datos tendaBD

- Tarefa 1.1. Seleccionar da táboa artigos as cores e o prezo medio de venda dos artigos de cada cor (con dous decimais).
- Tarefa 1.2. Seleccionar da táboa artigos as cores e o prezo medio de venda dos artigos de cada cor (con dous decimais), excluíndo aos artigos que teñan un prezo de compra superior a 50 euros.
- Tarefa 1.3. Mostrar as cidades nas que existen clientes e o número de clientes que hai en cada unha de elas. Clasificar a saída en orden decrecente polo número de clientes.
- Tarefa 1.4. Mostrar código, nome, suma dos importes das vendas sen aplicar o desconto, suma dos importes das vendas despois de aplicar o desconto, e desconto efectuado, para os artigos vendidos entre 1-1-2015 e 25-5-2015.
- Tarefa 1.5. Mostrar as estatísticas de vendas do ano 2015 por tenda. A información a mostrar é: identificador da tenda, número de vendas, número de artigos vendidos, suma de unidades vendidas e a media dos prezos unitarios dos artigos vendidos.

Sobre a base de datos traballadores

- Tarefa 1.6. Mostrar o número de empregados que utiliza cada extensión telefónica.
- Tarefa 1.7. Mostrar o número de empregados que teñen 0, 1, 2, 3, ... fillos.
- Tarefa 1.8. Mostrar, para cada departamento, o número de empregados que teñen 0, 1, 2, ... fillos.

- Tarefa 1.9. Mostrar o número de departamentos que dependen de cada centro.
- Tarefa 1.10. Seleccionar, para cada departamento, o maior salario, o menor salario e a diferenza que hai entre o salario máis alto e o salario máis baixo.
- Tarefa 1.11. Mostrar, para cada departamento, a cantidade que queda do presuposto despois de restar o importe dos salarios e comisións a pagar aos empregados.

Solución

- Tarefa 1.1

```
/*********************************************************************************
Seleccionar da táboa artigos as cores e o prezo medio de venda dos artigos de cada cor
(con dous decimais).
 *********************************************************************************/
select art_color,
   round(avg(art_pv),2)
from artigos
where art_color is not null
group by art_color;
```

- Tarefa 1.2

```
/*********************************************************************************
Seleccionar da táboa artigos as cores e o prezo medio de venda dos artigos de cada cor
(con dous decimais), excluíndo aos artigos que teñan un prezo de compra superior a 50
euros
*********************************************************************************/
select art_color as Color,
   avg(art_pv) as Prezo_medio
from artigos
where art_color is not null and art_pc <= 50
group by art_color;
```

- Tarefa 1.3

```
/*********************************************************************************
Mostrar as cidades nas que existen clientes, e o número de clientes que hai en cada
unha de elas. Clasificar a saída en orden decrecente polo número de clientes
*********************************************************************************/
select clt_poboacion as Poblacion,
   count(*) as Numero_clientes
from clientes
group by clt_poboacion
order by Numero_clientes desc;
```

- Tarefa 1.4

```
/*********************************************************************************
Mostrar código, nome, suma dos importes das vendas sen aplicar o desconto, suma dos
importes das vendas despois de aplicar o desconto, e desconto efectuado, para os
artigos vendidos entre 1-1-2015 e 25-5-2015.
*********************************************************************************/
select ar.art_codigo as Codigo,
   ar.art_nome as Nome,
   sum(dv.dev_prezo_unitario*dv.dev_cantidade) as Total,
   round(sum(dv.dev_prezo_unitario*dv.dev_cantidade*(1-dv.dev_desconto/100)),2)
          as Cobrado,
   round(sum(dv.dev_prezo_unitario*dv.dev_cantidade*(dv.dev_desconto/100)),2)
          as Desconto
from detalle_vendas as dv
   join artigos as ar on dv.dev_artigo=ar.art_codigo
   join vendas as ve on dv.dev_venda = ve.ven_id
where date(ve.ven_data) between "2015-01-01" and "2015-05-25"
group by ar.art_codigo, ar.art_nome;
```

- Tarefa 1.5

```
/*********************************************************************************
```

```
Mostrar as estatísticas de vendas do ano 2015 por tenda. A información a mostrar é:
identificador da tenda, número de vendas, número de artigos vendidos, suma de unidades
vendidas e a media dos prezos unitarios dos artigos vendidos
*******************************************************************************/
select ve.ven_tenda as Tenda,
   count(distinct ve.ven_id) as Numero_vendas,
   count(distinct dv.dev_artigo) as Numero_artigos,
   sum(dv.dev_cantidade) as Suma_unidades,
   round(avg(dv.dev_prezo_unitario),2) as Media_prezo
from vendas as ve join detalle_vendas as dv on dv.dev_venda = ve.ven_id
where year(ve.ven_data) = 2015
group by ve.ven_tenda;
```

- Tarefa 1.6

```
/*Mostrar o número de empregados que utiliza cada extensión telefónica*/
select empExtension as Extension,
   count(*) as Empregados
from empregado
group by empExtension;
```

- Tarefa 1.7

```
/*Mostrar o número de empregados que teñen 0, 1, 2, 3, ... fillos*/
select empFillos as Fillos,
   count(*) as Empregados
from  empregado
group by empFillos;
```

- Tarefa 1.8

```
/*Mostrar para cada departamento, o número de empregados que teñen 0, 1, 2, ... fillos,
ordenados por número de departamento e número de fillos.*/
select de.depNumero as Departamento,
   em.empFillos as Fillos,
   count(*) as Empregados
from empregado as em join departamento as de on em.empDepartamento=de.depNumero
group by de.depNumero,em.empFillos;
```

- Tarefa 1.9

```
/*Mostrar o número de departamentos que dependen de cada centro.*/
select depCentro as Centro,
   count(*) as Departamentos
from departamento
group by depCentro;
# Solución mostrando o nome do centro
select ce.cenNome as Centro,
   count(*) as Departamentos
from departamento as de join centro as ce on de.depCentro=ce.cenNumero
group by de.depCentro, ce.cenNome;
```

- Tarefa 1.10

```
/* Seleccionar, para cada departamento, o maior salario, o menor salario e a diferenza
que hai entre o salario máis alto e o salario máis baixo.*/
select    de.depNome as Departamento,
          max(em.empSalario)- min(em.empSalario) as Diferenza,
          max(em.empSalario) as Maior,
          min(em.empSalario) as Menor
   from empregado as em join departamento as de on em.empDepartamento=de.depNumero
   group by de.depNome;
```

- Tarefa 1.11

```
/*Mostrar por departamento, a cantidade que queda do presuposto despois de restar o
importe dos salarios e comisións a pagar aos empregados.*/
select de.depNumero as Numero,
   de.depNome as Departamento,
   de.depPresuposto-sum(em.empSalario)-ifnull(sum(em.empComision),0) as Dispoñible
```

```
from empregado as em
   join departamento as de on em.empdepartamento=de.depNumero
group by de.depNumero;
```

1.3.2 Tarefa 2. Realizar consultas establecendo condicións para os grupos da táboa resumo

A tarefa consiste en realizar as seguintes consultas agrupando filas e establecendo condicións para os grupos.

Sobre a base de datos tendaBD

- Tarefa 2.1. Seleccionar da táboa artigos as cores e o prezo medio de venda dos artigos de cada cor, para as cores que teñan o prezo medio maior que 100 euros.
- Tarefa 2.2. Mostrar as tendas que fixeron máis de 2 vendas no mes de maio de 2015. Para cada tenda débese mostrar: numero de tenda, número de vendas, número de artigos diferentes vendidos e a suma de unidades vendidas nese período de tempo.
- Tarefa 2.3Mostrar o identificador do cliente, data de venda, a cantidade de artigos vendidos, a suma dos importes das vendas na data e o desconto practicado nesas vendas, para os clientes aos que se vendeu máis de 1200 euros nun só día.
- Tarefa 2.4. Mostrar identificador de cliente, apelidos e nome na mesma columna separados por coma, e data e hora da venda, para os clientes que só teñen unha venda.

Sobre a base de datos traballadores

- Tarefa 2.5. Mostrar número e nome dos departamentos que teñan 5 empregados.
- Tarefa 2.6. Para as extensións telefónicas que son utilizadas por máis dun empregado, mostrar o número de empregados que a comparten.

Solución

- Tarefa 2.1

```
/*********************************************************************************
Seleccionar da táboa artigos as cores e o prezo medio de venda dos artigos de cada cor,
para as cores que teñan o prezo medio maior que 100 euros
*********************************************************************************/
select art_color,
   round(avg(art_pv),2)
from artigos
where art_color is not null
group by art_color
having avg(art_pv)>100;
/* utilizando os alias en having */
select art_color as Color,
   round(avg(art_pv),2) as Prezo_medio
from artigos
where art_color is not null
group by art_color
having Prezo_medio>100;
```

- Tarefa 2.2

```
/*********************************************************************************
Mostrar as tendas que fixeron máis de 2 vendas no mes de maio de 2015.
Para cada tenda débese mostrar: numero de tenda, número de vendas, número de
artigos vendidos diferentes e a suma de unidades vendidas nese período de tempo.
*********************************************************************************/
select ve.ven_tenda as Tenda,
   count(*) as Numero_vendas,
```

```
   count(distinct dv.dev_artigo) as Numero_artigos,
   sum(dv.dev_cantidade) as Numero_unidades_vendidas
from vendas as ve join detalle_vendas as dv on dv.dev_venda = ve.ven_id
where month(date(ve.ven_data)) = 5 and year(date(ve.ven_data)) = 2015
group by ve.ven_tenda
having Numero_vendas>2;
```

- Tarefa 2.3

```
/*********************************************************************************
Mostrar o identificador do cliente, data de venda, a cantidade de artigos vendidos,
a suma dos importes das vendas na data e o desconto practicado nesas vendas, para os
clientes aos que se vendeu máis de 1200 euros nun só día.
*********************************************************************************/
select ve.ven_cliente as Cliente,
   ve.ven_data as Data,
   sum(dv.dev_cantidade) as Cantidade,
   round(sum((dv.dev_cantidade*dv.dev_prezo_unitario)*(1-dv.dev_desconto/100)),2)
          as Importe,
   round(sum(dv.dev_cantidade * dv.dev_prezo_unitario)*(dv.dev_desconto/100),2)
          as Desconto
from vendas as ve join detalle_vendas as dv on ve.ven_id = dv.dev_venda
group by ve.ven_cliente, ve.ven_data
having sum((dv.dev_cantidade*dv.dev_prezo_unitario)*(1-dv.dev_desconto/100))>=1200;
```

- Tarefa 2.4

```
/*********************************************************************************
Mostrar o identificador de cliente, apelidos e nome na mesma columna separados por
coma, e data e hora da venda, para os clientes que só teñen unha venda.
*********************************************************************************/
select ve.ven_cliente as Cliente,
   concat(cl.clt_apelidos,', ',cl.clt_nome) as 'Apelidos e nome',
   ve.ven_data as Data
from clientes as cl join vendas as ve on cl.clt_id=ve.ven_cliente
group by cl.clt_id
having count(*)=1;
```

- Tarefa 2.5

```
/*Mostrar número e nome dos departamentos que teña 5 empregados.*/
select de.depNumero as Numero,
   de.depNome as Nome
from departamento as de join empregado as em on de.depNumero = em.empDepartamento
group by de.depNumero
having count(*) = 5;
```

- Tarefa 2.6

```
/* Para as extensións telefónicas que son utilizadas por máis dun empregado, mostrar o
número de empregados que a comparten.*/
select empExtension as Extension,
   count(*) as Comparten
from empregado
group by empExtension
having count(*) > 1;
```